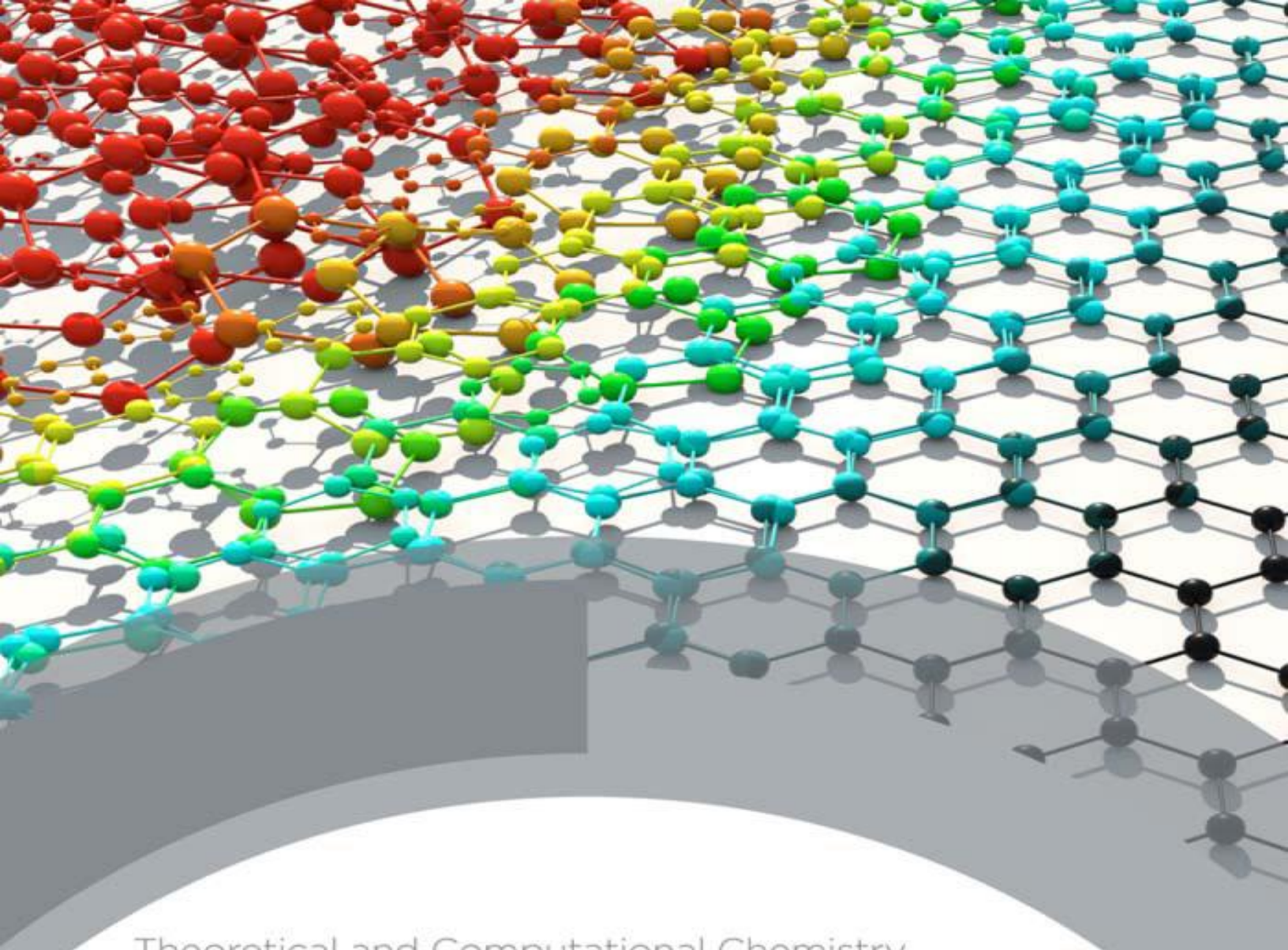


Theoretical and Computational Chemistry

Machine Learning in Chemistry

The Impact of Artificial Intelligence

Edited by Hugh M. Cartwright

ROYAL SOCIETY OF CHEMISTRY

Machine Learning in Chemistry
The Impact of Artificial Intelligence

Theoretical and Computational Chemistry Series

Editor-in-chief:
Jonathan Hirst, *University of Nottingham, Nottingham, UK*

Advisory board:
Dongqing Wei, *Shanghai Jiao Tong University, China*
Jeremy Smith, *Oakridge National Laboratory, USA*

Titles in the series:

1: Knowledge-based Expert Systems in Chemistry: Not Counting on Computers
2: Non-Covalent Interactions: Theory and Experiment
3: Single-Ion Solvation: Experimental and Theoretical Approaches to Elusive Thermodynamic Quantities
4: Computational Nanoscience
5: Computational Quantum Chemistry: Molecular Structure and Properties *in Silico*
6: Reaction Rate Constant Computations: Theories and Applications
7: Theory of Molecular Collisions
8: *In Silico* Medicinal Chemistry: Computational Methods to Support Drug Design
9: Simulating Enzyme Reactivity: Computational Methods in Enzyme Catalysis
10: Computational Biophysics of Membrane Proteins
11: Cold Chemistry: Molecular Scattering and Reactivity Near Absolute Zero
12: Theoretical Chemistry for Electronic Excited States
13: Attosecond Molecular Dynamics
14: Self-organized Motion: Physicochemical Design based on Nonlinear Dynamics
15: Knowledge-based Expert Systems in Chemistry: Artificial Intelligence in Decision Making
16: London Dispersion Forces in Molecules, Solids, and Nano-structures: An Introduction to Physical Models and Computational Methods
17: Machine Learning in Chemistry: The Impact of Artificial Intelligence

How to obtain future titles on publication:
A standing order plan is available for this series. A standing order will bring delivery of each new volume immediately on publication.

For further information, please contact:
Book Sales Department, Royal Society of Chemistry, Thomas Graham House, Science Park, Milton Road, Cambridge, CB4 0WF, UK
Telephone: +44 (0)1223 420066, Fax: +44 (0)1223 420247, Email: booksales@rsc.org
Visit our website at www.rsc.org/books

Machine Learning in Chemistry

The Impact of Artificial Intelligence

Edited by

Hugh M. Cartwright
Oxford University, UK
Email: hugh.cartwright@chem.ox.ac.uk

ROYAL SOCIETY OF CHEMISTRY

Theoretical and Computational Chemistry Series No. 17

Print ISBN: 978-1-78801-789-3
PDF ISBN: 978-1-83916-023-3
EPUB ISBN: 978-1-83916-024-0
Print ISSN: 2041-3181
Electronic ISSN: 2041-319X

A catalogue record for this book is available from the British Library

© The Royal Society of Chemistry 2020

All rights reserved

Apart from fair dealing for the purposes of research for non-commercial purposes or for private study, criticism or review, as permitted under the Copyright, Designs and Patents Act 1988 and the Copyright and Related Rights Regulations 2003, this publication may not be reproduced, stored or transmitted, in any form or by any means, without the prior permission in writing of The Royal Society of Chemistry or the copyright owner, or in the case of reproduction in accordance with the terms of licences issued by the Copyright Licensing Agency in the UK, or in accordance with the terms of the licences issued by the appropriate Reproduction Rights Organization outside the UK. Enquiries concerning reproduction outside the terms stated here should be sent to The Royal Society of Chemistry at the address printed on this page.

Whilst this material has been produced with all due care, The Royal Society of Chemistry cannot be held responsible or liable for its accuracy and completeness, nor for any consequences arising from any errors or the use of the information contained in this publication. The publication of advertisements does not constitute any endorsement by The Royal Society of Chemistry or Authors of any products advertised. The views and opinions advanced by contributors do not necessarily reflect those of The Royal Society of Chemistry which shall not be liable for any resulting loss or damage arising as a result of reliance upon this material.

The Royal Society of Chemistry is a charity, registered in England and Wales, Number 207890, and a company incorporated in England by Royal Charter (Registered No. RC000524), registered office: Burlington House, Piccadilly, London W1J 0BA, UK, Telephone: +44 (0) 20 7437 8656.

For further information see our web site at www.rsc.org

Printed in the United Kingdom by CPI Group (UK) Ltd, Croydon, CR0 4YY, UK

Preface

Progress in science can be lumpy. Ground-breaking advances, such as the emergence of quantum theory, the development of spectroscopic techniques including IR and NMR, or the widespread adoption of computers in chemistry, catalyse research, and open new fields. Artificial intelligence (AI) is the latest significant advance.

Perhaps, this assessment of AI seems too generous. Can a computational technique really be transformational? After all, in every university chemistry department, there are computers and spectrometers, but in only some is AI used. However, there are signs that AI may have as great an impact in science as it promises to do in society more broadly; this book explores that impact.

We begin with an introduction to some of the core concepts in AI and machine learning, at a level suitable for those who are new to the field (Allen, Chapters 1 and 2). The tutorial that follows outlines some of the most widely used AI methods at the interface of machine learning and medicine (Lawrence, Chapter 3). Subsequent chapters deal with a wide range of topics, including fundamental studies in catalysis (Liu, Chapter 19) and prediction of the properties of materials (Winkler, Chapter 9; Jelfs, Chapter 12; Brgoch, Chapter 13); studies of synthesis design (Stukenbroeker, Chapter 6; Hirst, Chapter 7; Brgoch, Chapter 13); drug design (Hudson, Chapter 11; Speck-Planche, Chapter 16); industrial applications (Curteanu, Chapter 10; Clough, Chapter 14); theoretical areas of chemistry (Marquetand, Chapter 4; Mizoguchi, Chapter 17) and some more technical aspects of the use of AI (Staker, Chapter 15); autonomous chemistry (Stukenbroeker, Chapter 6; Simpson, Chapter 18); and chemical

astronomy (Viti, Chapter 8). Several chapters (among them Stukenbroeker, Chapter 6; Hirst, Chapter 7; Brgoch, Chapter 13; Simpson, Chapter 18, and Shankar, Chapter 20) touch upon some of the difficulties that may complicate the use of AI in science and consider how one might circumvent them, while one further chapter (Cartwright, Chapter 5), which also discusses challenges and solutions when using AI in science, is aimed principally at newcomers to the field.

Chemistry research that takes advantage of AI is burgeoning: from a modest number of published papers in 2000, the publication rate had, by the end of 2019, risen 100-fold. Several factors have contributed to this growth:

- access to increasingly large volumes of data;
- a continuing rise in computer speed;
- improvements in the efficiency of AI software;
- the availability of innovative specialised chips, such as custom neural network chips;
- the re-purposing of graphics chips as fast surrogate CPUs for AI applications;
- the refinement of existing AI methods and the development of new techniques;
- recognition that some types of problems can be more successfully tackled using AI than with more conventional methods of analysis; and
- the availability of upgraded tools for automatic data extraction from sources of non-standard structure, such as journals, Powerpoint presentations, or laboratory notebooks.

To that list, we add (cautiously) the future availability of quantum computers. The current view is that such computers will be particularly well-suited to optimisation problems, which are widespread in chemistry (What does the lowest energy conformer look like? Which potential drug binds most strongly to the protein? Which synthetic route provides the highest yield?). Chemists can expect to be among the greatest beneficiaries of this new technology.

Chapters within this book, written by leading groups in the field, provide intriguing examples of how AI can be applied in chemistry. They provide a fascinating glimpse into the future of AI in science.

Hugh Cartwright
Victoria, Canada

Contents

Chapter 1 Computers as Scientists
Timothy E. H. Allen

1.1 What Is Computational Science?
1.2 What Is Artificial Intelligence?
1.3 What Is Machine Learning?
References

Chapter 2 How Do Machines Learn?
Timothy E. H. Allen

2.1 Framing the Question
2.2 Gathering Data
2.3 Setting up the Algorithm
2.4 The Training Procedure
2.5 Overcoming Flaws
2.6 Deploying the Algorithms
Note on References
References

Chapter 3 MedChemInformatics: An Introduction to Machine Learning for Drug Discovery
Matthew G. Roberts and Rae Lawrence

3.1 Introduction

3.1.1 Artificial Intelligence *vs.* Machine Learning
3.1.2 Supervised *vs.* Unsupervised Learning
3.2 Forewarned Is Forearmed
3.2.1 Dataset Assembly and Curation
3.2.2 Model Building
3.2.3 Principal Component Analysis
3.2.4 When Machines Learn in 3D, Alignment IS EVERYTHING
3.2.5 Activity Range and Distribution of Activities for QSAR Modelling
3.2.6 Outliers
3.3 Deep Learning and Neural Networks
3.4 Underlying Mathematics
3.4.1 "Observables" and "Features"
3.4.2 Comparing Molecules Based on Their Descriptors
3.4.3 Model Quality and Statistics
3.4.4 Overtraining and Characteristics of Good Descriptors
3.5 Machine Learning Methods
3.5.1 *k*-Nearest Neighbours
3.5.2 Linear Regression
3.5.3 Decision Trees and Random Forests
3.5.4 Support Vector Machines
3.6 Summary
Acknowledgements
References

Chapter 4 Machine Learning for Nonadiabatic Molecular Dynamics
Julia Westermayr and Philipp Marquetand

4.1 Introduction
4.2 Methods
4.2.1 Machine Learning (ML) Models
4.2.2 Training Set Generation

4.2.3 Wave Function Phases
4.2.4 Phase Correction Algorithm
4.2.5 Surface Hopping Dynamics
4.3 Example: Methylenimmonium Cation
4.3.1 ML Surface Hopping Dynamics
4.3.2 Energy Conservation
4.3.3 Further Tools of ML Models
4.4 Conclusion and Outlook
References

Chapter 5 Machine Learning in Science – A Role for Mechanical Sympathy?
Hugh M. Cartwright

5.1 Introduction
5.1.1 A Little History
5.1.2 Challenges
5.2 Problems and Solutions
5.2.1 How Many Samples do I Need to Train an AI?
5.2.2 We Are Combining Databases; Do Difficulties Await?
5.2.3 We Use a Chemical Reactions Database to Predict Synthetic Routes, but We Are Making Little Progress. What Might Be Going Wrong?
5.2.4 Our Database Is so Large That Training Will Take Forever. What Can We Do?
5.2.5 How Do I Set the Parameters?
5.2.6 How Has My AI Learnt Something That Is Not There?
5.2.7 My Trained AI Tested Beautifully; Now It Produces Rubbish
5.2.8 My AI Solved the Challenges in Box 5.4; How Did It Manage It?
5.2.9 My AI Thinks that Basketball Players Cause Global Warming

5.2.10 Results from my AI Suggest that my Data Contain only Noise, but I Think there Might be Something Real present. How can I Tell?
5.2.11 My AI Was Fine Until I Found Some Extra Data to Give It; Why Did the Data Make It Worse, Not Better?
5.2.12 My AI Struggles Even Though It Has Plenty of Data
5.2.13 The Problem: My AI Is Just Odd
5.2.14 It Is Going Pear-shaped. What Can I Do?
5.3 Conclusion
References

Chapter 6 A Prediction of Future States: AI-powered Chemical Innovation for Defense Applications
Tyler Stukenbroeker and Jonathan Clausen

6.1 Introduction
6.2 The Synthesis Search Engine: Automating Synthetic Planning
6.2.1 Optimizing Known Routes
6.2.2 Predicting Reaction Outcomes
6.2.3 Performing Retrosynthesis
6.2.4 Assessment
6.2.5 Adoption
6.3 Data Challenges for Statistical Learning in Chemistry
6.3.1 Data are Biased Toward Successful Experiments
6.3.2 Existing Data are Unreliable and Non-standardized
6.3.3 Data are Relevant to a Narrow Collection of Tasks
6.3.4 There Is No Standard Representation of Chemistry Data for Machine Learning Models
6.4 Realizable Solutions to Data Challenges

Automation and Standardization of
6.4.1 Experimental Data Collection
6.4.2 Sharing Data Across Laboratories
6.4.3 Leveraging Experiments Described in Scientific Literature
6.4.4 Generating Data Through Simulation
6.4.5 Closed-loop Integration of Data Generation and Predictive Models
6.5 Initial Demonstrations of the Self-driving Laboratory
6.5.1 Implications of Autonomous Research
6.6 Conclusion
Additional Reading
Acknowledgements
References

Chapter 7 Machine Learning for Chemical Synthesis
Alexe L. Haywood, Joseph Redshaw, Thomas Gaertner, Adam Taylor, Andy M. Mason and Jonathan D. Hirst

7.1 Introduction
7.2 The Nature of the Chemical Data
7.2.1 Data Sources
7.2.2 Molecular Descriptors
7.3 Machine Learning Methods
7.4 Synthetic Route Design
7.4.1 Retrosynthesis
7.4.2 Reaction Prediction
7.4.3 Optimising Reaction Conditions
7.5 Real-life Applications
7.6 Conclusions
Acknowledgements
References

Chapter 8 Constraining Chemical Networks in Astrochemistry
Serena Viti and Jonathan Holdship

8.1 Introduction
8.2 Completeness and Reliability of Chemical Reactions Databases
8.2.1 Gas-phase Networks
8.2.2 Grain-surface Networks
8.3 A Bayesian Approach
8.4 Machine Learning Techniques for Complex Networks
8.5 Conclusion
References

Chapter 9 Machine Learning at the (Nano)materials-biology Interface
David A. Winkler

9.1 Machine Learning Overview, Challenges, and Opportunities
9.2 Scale and Importance of Problem/Special Issues with Complex Materials
9.3 Machine Learning Examples in Materials
9.4 Machine Learning Examples in Nanomaterials
9.5 Machine Learning Examples in Biomaterials and Regenerative Medicine
9.6 Materials for Cell Therapies, Bioreactors, and Implantable Cells
9.7 ML and Evolutionary Methods
9.8 The Path Forward
References

Chapter 10 Machine Learning Techniques Applied to a Complex Polymerization Process
Silvia Curteanu

10.1 Difficulties in Chemical Process Modeling
10.2 Phenomenological Modeling of the Free-radical Polymerization Process

10.3 Artificial Neural Networks Applied to Polymerization Processes
10.3.1 Types of Applications Based on Neural Networks in Polymerization Reaction Engineering
10.3.2 Different Types of Neural Network Models Used in Modeling of Polymerization Processes
10.4 Soft-computing Hybrid Configurations Applied to Polymerization Processes
10.5 Machine Learning Techniques Applied to Methyl Methacrylate Free Radical Polymerization
References

Chapter 11 Machine Learning and Scoring Functions (SFs) for Molecular Drug Discovery: Prediction and Characterisation of Druggable Drugs and Targets
I. L. Hudson, S. Y. Leemaqz and A. D. Abell

11.1 Introduction
11.2 Data and Methods
11.2.1 Data and Druggability Scoring for Molecules and Disease Targets
11.2.2 Description of the MPs and Targets by Score Function Strata
11.2.3 Statistical and Program Approach
11.3 Results
11.3.1 PCA Results
11.3.2 Machine Learning Classification Results
11.4 Conclusion and Discussion
Acknowledgements
References

Chapter 12 Artificial Intelligence Applied to the Prediction of Organic Materials

Steven Bennett, Andrew Tarzia, Martijn A. Zwijnenburg and Kim E. Jelfs

12.1 Introduction
12.2 Extracting Data from the Literature
12.3 Building Block Libraries
12.4 Predicting Properties
12.4.1 Training Data
12.4.2 Descriptors
12.4.3 Accelerated Property Prediction
12.5 Structure Prediction
12.6 Exploration of Phase Space
12.7 *De Novo* Molecular Design of Materials Building Blocks
12.8 Predicting Synthetically Accessible Materials
12.9 Case Studies
12.9.1 Organic Electronics
12.9.2 Property Prediction and Phase Space Exploration of Porous Organic Materials
12.9.3 Other Functional Organic Materials
12.10 Conclusions
Acknowledgements
References

Chapter 13 A New Era of Inorganic Materials Discovery Powered by Data Science
Ya Zhuo, Aria Mansouri Tehrani and Jakoah Brgoch

13.1 Introduction
13.2 Workflow for Materials Discovery *via* Machine Learning
13.2.1 Data Collection
13.2.2 Data Representation
13.2.3 Data Pre-processing
13.2.4 Model Construction
13.2.5 Model Evaluation

13.3 Applications of Machine Learning in Inorganic Materials
13.3.1 Machine Learning in Synthesis Optimization
13.3.2 Machine Learning in Structure Prediction
13.3.3 Machine Learning in Material Property Prediction
13.4 Challenges and Opportunities
13.4.1 Learning from Small Data Sets
13.4.2 Generating Efficient Data Representation
13.4.3 Enhancing Model Interpretability
13.4.4 Increasing Model Accessibility
13.5 Concluding Remarks
Acknowledgements
References

Chapter 14 Machine Learning Applications in Chemical Engineering
Y. Yan, T. N. Borhani and P. T. Clough

14.1 Introduction
14.2 Modelling and Simulation
14.2.1 Application of AI for Modelling and Simulation of Separation Units
14.2.2 Application of AI for Modelling and Simulation of Reactors
14.2.3 Application of AI for Modelling and Simulation of Whole Plants
14.2.4 Application of AI for Modelling and Simulation of Energy Management
14.3 Control and Operation
14.3.1 The Baseline of Process Control
14.3.2 Machine Learning Used to Control Chemical Processes
14.3.3 Traditional Controllers with Integrated Machine Learning
14.4 Estimation and Prediction

14.4.1 Available AI Algorithms for Estimators in Chemical Engineering
14.4.2 Selection and Applications of AI-based Estimators
14.5 Future Trends and Outlook
References

Chapter 15 Representation Learning in Chemistry
Joshua Staker, Gabriel Marques and J. Dakka

15.1 Introduction
15.2 Non-learned Molecular Representations
15.2.1 Molecular Fingerprints
15.2.2 Computable Features
15.3 The Need for Learned Representations
15.4 Learning Molecular Representations
15.4.1 Graph Models
15.4.2 Grid Models
15.4.3 Sequence Models
15.4.4 Physically Inspired Architectures
15.5 Data Considerations
15.6 Current Challenges and Directions for Future Research
15.7 Conclusion
Acknowledgements
References

Chapter 16 Demystifying Artificial Neural Networks as Generators of New Chemical Knowledge: Antimalarial Drug Discovery as a Case Study
Alejandro Speck-Planche and Valeria V. Kleandrova

16.1 Introduction
16.2 Materials and Methods
16.2.1 Dataset and Molecular Descriptors
16.2.2 Building the Ms-QSBER_EL model

16.3 Results and Discussion
16.3.1 Ms-QSBER-EL model
16.3.2 Applicability Domain
16.3.3 Gathering Physicochemical and Structural Information from the Molecular Descriptors
16.3.4 Computational Design of Virtually Potent and Safe Antimalarial Chemicals
16.3.5 Druglikeness
16.4 Conclusion
Acknowledgements
References

Chapter 17 Machine Learning for Core-loss Spectrum
T. Mizoguchi and S. Kiyohara

17.1 Introduction
17.2 Methodology
17.2.1 Spectrum Database Construction
17.2.2 Spectral Data Clustering
17.2.3 Decision Tree for Material Information
17.2.4 Feedforward Neural Network
17.3 Results and Discussion
17.3.1 Overview of the Data-driven Prediction and Interpretation Method
17.3.2 Demonstration Using the O-K Edges of Mono-metal Oxides
17.3.3 Interpretation and Prediction of the O-K Edges of Polymorphous SiO_2
17.3.4 Machine Learning to Quantify Structures and Properties from ELNES/XANES
17.3.5 Quantification of Structures and Properties from Noisy Spectra
17.4 Conclusions
Acknowledgements
References

Chapter 18 Autonomous Science: Big Data Tools for Small Data Problems in Chemistry

Andreas C. Geiger, Ziyi Cao, Zhengtian Song, James R. W. Ulcickas and Garth J. Simpson

18.1 Introduction
18.2 Autonomous Design of Experiments
18.2.1 Sparse Sampling Strategies for “Small Data” Challenges
18.2.2 Simplex Approaches for Autonomous Design of Experiments
18.2.3 Interpolation Algorithms for Dynamic Sampling
18.2.4 A Supervised Learning Approach for Dynamic Sampling
18.2.5 SLADS for Raman Hyperspectral Imaging and X-Ray Diffraction Imaging
18.2.6 Noniterative Dynamic Sampling for Autonomous Design of Experiments
18.2.7 Active Machine Learning
18.2.8 Dynamic Sampling for Materials Synthesis
18.2.9 Modular Architectures for Autonomous Design of Experiments
18.3 Generative Adversarial Methods for Data-limited Training
18.3.1 Review of Generative Adversarial Networks in Image Recognition Applications
18.3.2 Illustrative Linear Example of Adversarial Attacks: Adversarial Spectroscopy
18.3.3 Illustrative Example of Data Generation: Generative Adversarial Linear Analysis
18.4 Conclusion
Acknowledgements
References

Chapter 19 **Machine Learning for Heterogeneous Catalysis: Global Neural Network Potential from Construction to Applications**
Sicong Ma, Pei-Lin Kang, Cheng Shang and Zhi-Pan Liu

19.1 Introduction
19.2 Methods
19.2.1 High-dimensional NN Architecture
19.2.2 Structural Descriptors
19.2.3 Neural Network Training
19.2.4 Data Set Generation and SSW Global Optimization
19.2.5 Self-learning Procedure
19.3 Applications
19.3.1 PES Exploration
19.3.2 Ultrasmall Au Supported on CeO_2
19.3.3 Hydrogen Evolution Reaction on Amorphous TiO_xH_y
19.3.4 Syngas Conversion on ZnCrO Catalysts
19.4 Prospects
Acknowledgements
References

Chapter 20 **A Few Guiding Principles for Practical Applications of Machine Learning to Chemistry and Materials**
S. Shankar and R. N. Zare

20.1 Introduction
20.2 Guiding Principles for Applications of Machine Learning
Guiding Principle 1: *Use ML for Interpolation but with Care for Extrapolation*
Guiding Principle 2: *Ensure Consistency Between Sources of Data Used in the ML Development and the Targeted Application*

Guiding Principle 3: *Correlation is not Causation*
Guiding Principle 4: *Optimize Information Extraction when Using ML*
Guiding Principle 5: *Combine Different Methods Consistently, Including Experiments, Theory, and Simulations, to Provide a Larger Window of Applicability*
20.3 Concluding and Cautionary Remarks
Acknowledgements
References

Subject Index

CHAPTER 1

Computers as Scientists

TIMOTHY E. H. ALLEN[a,b]

[a] MRC Toxicology Unit, Hodgkin Building, Lancaster Road, Leicester LE1 7HB, UK
[b] Centre for Molecular Informatics, Department of Chemistry, University of Cambridge, Lensfield Road, Cambridge CB2 1EW, UK
Email: teha2@cam.ac.uk

1.1 What Is Computational Science?

Computers excel at many tasks that humans are very bad at (see Figure 1.1). For example, if you have a smartphone with a voice assistant, try asking it what the square root of 32 761 is. For the phone, this problem is very straightforward, and it almost immediately returns the correct answer of 181. If instead, we were to ask a person the same question, they would probably find it very challenging. However, if they get the correct answer as quickly as the phone, we will think they are a genius (or perhaps more likely that they are cheating!). Now, for comparison, try to engage your voice assistant with a knock-knock joke, and it doesn't work. In contrast to the mathematical challenge, the machine finds this social interaction impossibly complicated, but we would think a person a simpleton if he/she is not able to partake in it. Children learn how to make these jokes before they learn to add. This illustrates why computers make such good scientific companions as they help scientists to be better at the things that they are naturally much worse at.

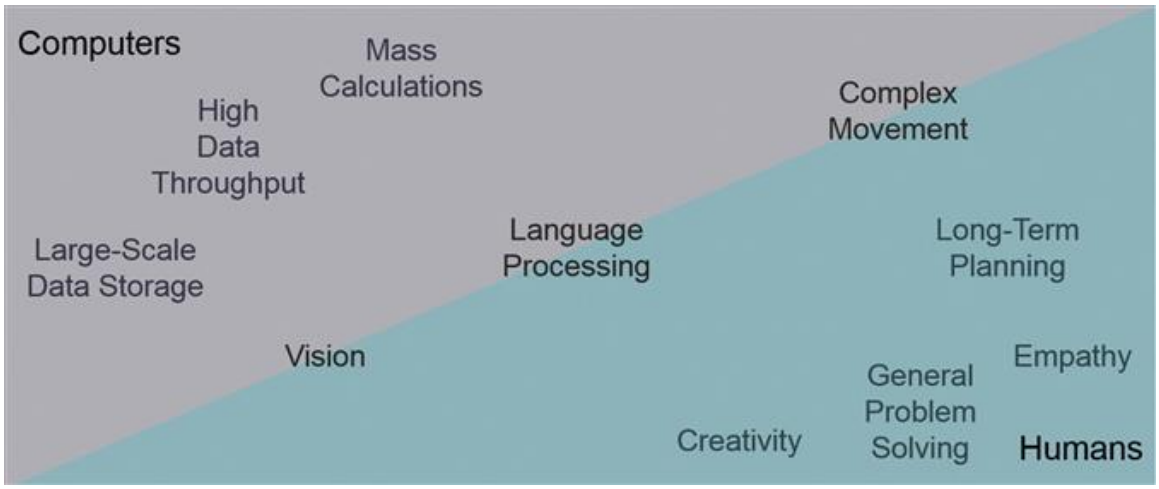


Figure 1.1 Strengths and weaknesses of humans and computers.

Computational science is a scientific field, which uses a computational methodology to answer scientific questions. Computational scientists are most often trying to answer questions in fields such as chemistry, physics or biology using the algorithms developed by computer scientists. This is, in some respects, analogous to an experimental chemist who uses a reaction to make a molecule in the lab, as opposed to the one who proposes that reaction, or a scientist who uses an NMR machine to determine the chemical composition of a biological sample using a procedure devised by an NMR expert. This is not to say that the scientists deploying these techniques, which might be algorithmic, experimental or otherwise, are any less important than their discoverers. The most impactful scientific techniques are those which have long-term applications across multiple fields.

The use of artificial intelligence (AI) and machine learning (ML) algorithms, as described later in this book, could be considered as computational science, but chemistry has a long track record in other computational methods too. Numerous computational techniques do not require AI but rather use the power of computers to solve mathematical equations. Many of the most complex problems are in quantum mechanics.

The field of quantum mechanics is just over 100 years old. In 1900, Max Planck[1] proposed that energy is absorbed and radiated in discrete quanta (or packets of energy). Albert Einstein[2] used quantum theory to explain the

photoelectric effect in 1905, which led to the theory that light acts as both a wave and a particle. Louis de Broglie[3] extended wave–particle duality to include all types of matter, including, importantly, electrons. Once it was accepted that electrons behave as waves, the hunt began to find a way of describing these waves mathematically. A mathematical model was proposed by Erwin Schrodinger[4] in 1925, and this is shown below, in its time-independent form;

$$\hat{H}|\Psi\rangle = E|\Psi\rangle \tag{1.1}$$

where $\hat{H}$ is the Hamiltonian operator, Ψ is the wavefunction and E is a constant equal to the energy of the system. The mathematics of this equation is far beyond the scope of this chapter, but it is worth noting that finding solutions to the Schrodinger equation is not trivial for any system. At present, the equation cannot be solved exactly for species with multiple electrons. However, with suitable assumptions and high-powered computers, approximate probability distributions can be found, which tell us where electrons are (approximately, as for quantum systems, it is impossible to know exactly where the electron is) and their associated energy levels. Such details are useful in chemistry to help explain how, and why, the reactions happen.

Because of the wealth of detail that quantum calculations provide, the use of computational calculations to explain, predict and even mimic chemical reactions is an extremely important field. Ken Houk's group at UCLA, for example, have been exploring asymmetric chemical catalysis for many years using computational calculations.[5] Such chemical reactions are crucial in the manufacture of molecules with chiral centres, and many of which have important biological properties. Chemicals can have handedness or chirality, which can lead to important biological consequences, as some molecules fit into spaces that others cannot. To manufacture such molecules, we must consider which product(s) might be generated by a chemical reaction that could follow several different pathways. Computers can help us predict why certain reactions are more favourable. To do this, we need, among other things, to calculate the energy barrier that exists between the reactants and the products of a reaction. Computers can consider the shapes of molecules in 3D, their interactions

with one another and the energies of these systems, allowing the calculation of these so-called activation energies (see Figure 1.2).

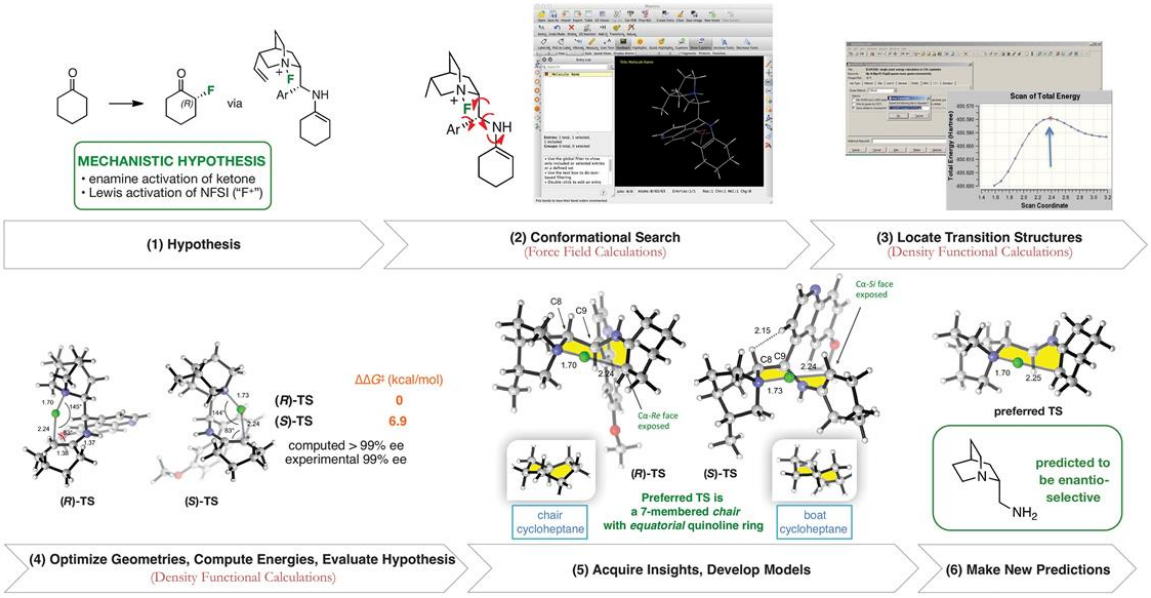


Figure 1.2 A common procedure used in the Houk group for the computational investigation of chemical reactions. Reproduced from ref. 5 with permission from American Chemical Society, Copyright 2016.

From among all the chemically reasonable routes from reactants to products, how can we select the one that has the lowest activation energy? This low-energy pathway will often dominate the reaction, and therefore its product should be the one that is experimentally observed. Calculations of the activation barrier can be considered alongside the experimental observations to evaluate the hypothesis and gain insight – does what is expected through the calculations match the observation? If so, the model can provide useful insight into the chemistry of the interaction by providing information about the structure and dynamics of the transition state as chemicals move from reactants to products. Perhaps, is there an unfavourable interaction between two large groups in the unfavoured pathway that raises its energy? Or is there any favourable electronic interaction in the favoured pathway that lowers the barrier? Such insights can be used subsequently to design new and more efficient reactions.

1.2 What Is Artificial Intelligence?

Defining AI is surprisingly challenging. When trying to do so, I often think of the idea of a generally intelligent machine (a machine able to do a wide range of different tasks as a human does) and would, therefore, define an intelligent machine as *A machine capable of doing things that humans consider to be intelligent*.

Although this may not be entirely satisfactory, it sums up the goal of AI science quite well as humans are the most intelligent species on earth. It also aligns with the well-known Turing test, developed by Alan Turing in 1950. The standard version of this test requires a human interpreter to converse with, and subsequently judge, two other hidden players, of which one is another human and one is a machine. If, after having conversations with both players, the interpreter cannot distinguish between the machine and the human, the machine would be deemed to have passed the test and therefore be considered "intelligent" (within the boundaries of Turing's test of course) (see Figure 1.3).[6] In this context, the test is often referred to as "the imitation game".

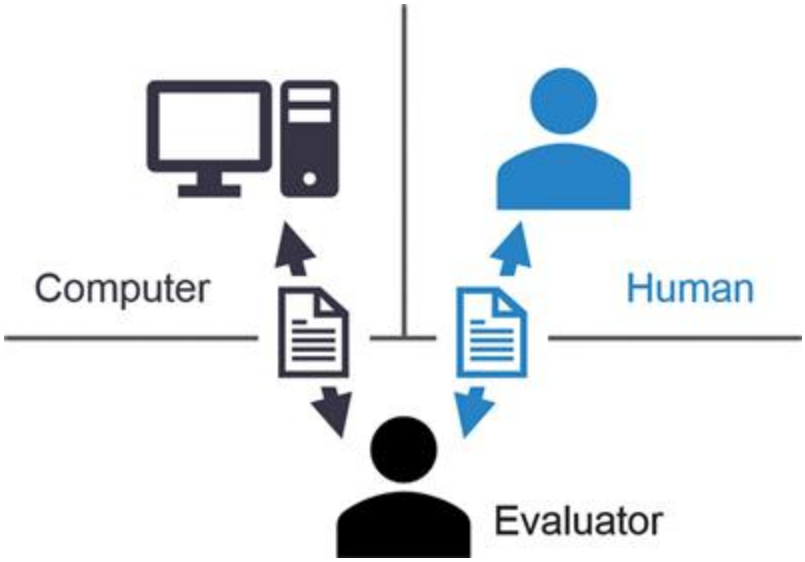


Figure 1.3 The Turing test.

Some tasks that we might consider require a degree of intelligence include speech and vision, scientific and medical decision making, and strategic gameplay. We will consider some examples of these tasks in the rest of this section.

AI has been a pursuit of scientists at least since the 1950s. During that decade, there were three important meetings. A “Session on Learning Machines” in Los Angeles in 1955, a “Summer Research Project on Artificial Intelligence” at Dartmouth College in 1956, and a symposium on the “Mechanization of Thought Processes” at the National Physical Laboratory in the United Kingdom.[6] These early meetings tackled many of the problems that still occupy AI researchers today. These include examples of tasks such as pattern recognition,[7] understanding of language,[8] and playing chess.[9] Other, more mechanistic, issues were also discussed, including the imitation of the human brain and central nervous system as the basis for learning machines,[7] and the use of iterative learning to ultimately improve computer performance in learning tasks,[10] as well as the realisation that powerful machines would be required to address many of these challenging topics.

Let us consider again the definition of AI given earlier. What is it about humans that makes them intelligent? And does answering this help us understand what kinds of things machines need to be able to do to also be considered intelligent?

Humans sense the world around them through vision. They explore the world by moving around it. They can speak to one another and recognise incoming speech. They can also recognise patterns in objects they encounter and events they experience. These goals are all currently being explored in AI, in the fields of computer vision, robotics, speech recognition, natural language processing, and pattern recognition. Let's cover some examples from the history of AI in these different research areas.

In the late 1950s and early 1960s, a number of projects attempted to use pattern recognition algorithms to aid in the identification of target objects in aerial photographs (aerial reconnaissance). Laveen N. Kanal, Neil C. Randall, and Thomas Harley at the Philco Corporation, for example, attempted to screen aerial photographs for military tanks.[6,11] A small

section of the film was processed to enhance any edges, and the result presented to the target detection system as a 32×32 array of 1's and 0's. This array was segmented into 24 overlapping 8×8 “feature blocks”, and each of which was then evaluated using a statistical test to establish if the block contained a part of the tank.

These tests assessed 50 photos containing tanks and 50 without tanks. This allowed a statistical model to be prepared to discriminate tank-containing feature blocks from non-tank-containing feature blocks using a linear boundary in 64 dimensions. A score was calculated for each image by considering how many of the 24 feature blocks in an image suggested the presence of a tank. This can be considered analogous to how a random forest[12] model works (see Figure 1.4).

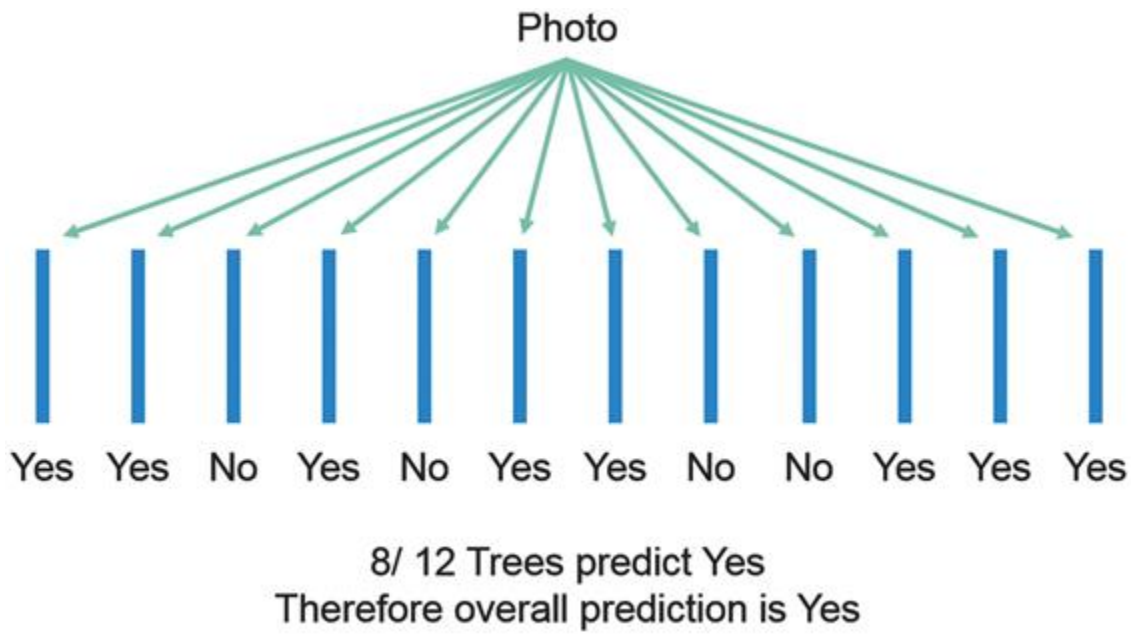


Figure 1.4 In a random forest model, each “tree” is a different model, and each evaluates the incoming data to make a prediction. The overall prediction from the random forest is a consensus of these.

The performance of the model on the test images was excellent. Every image containing tanks had a score of at least 11 while all images without a tank had a score of 7 or less. In addition, half of the test images were perfectly classified, *i.e.* a score of 24 for an image containing a tank or a

score of 0 for a non-tank image. Given the technological limits of the era (this work was not programmed on a computer, it used analogue circuitry), this is quite remarkable.

Another early project in pattern recognition and computer vision concerned human face recognition. Woodrow W. Bledsoe, Charles Bisson, and Helen Chan developed techniques for facial recognition at Panoramic Research in the 1960s.[6,13] In this early work, a human operator extracted the coordinates of facial features, such as the position of the pupils, from photographs. From these coordinates, a set of 20 distances was calculated, such as the width of eyes, and these distances were then stored on a computer to be compared to the same distances in previously unseen photos during the recognition phase. The closest records were returned, using a nearest neighbour[14] calculation to make a prediction (see Figure 1.5).

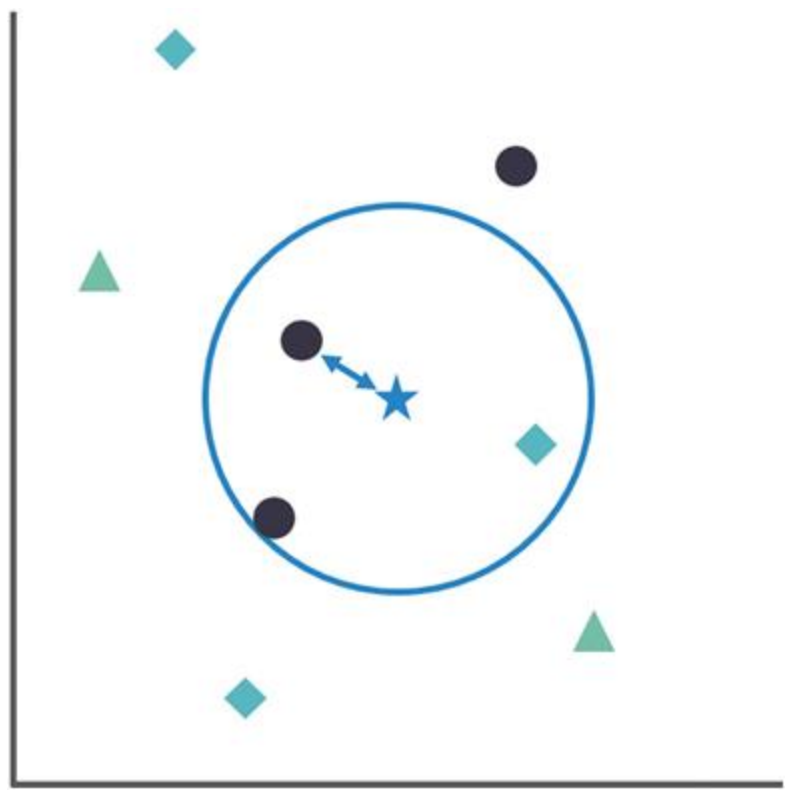


Figure 1.5 In a nearest neighbour model, the new example (blue star) is considered in the same feature space as all examples previously seen (other shapes). The nearest neighbour (double-headed arrow) or more commonly a collection of near neighbours (large

circle) are used for the classification of the new case. In this case, the new example fits most closely with the circles.

In 1970, this work was taken further by M.D. Kelly during his PhD at Stanford.[6,15] Kelly wrote a computer program, which was able to detect the facial features by itself, for the identification of people, enormously reducing the workload required.

Subsequent work on facial recognition software progressed rapidly. A 2007 National Institute of Standards and Technology report showed the results of testing seven state-of-the-art face recognition algorithms against humans under different lighting conditions.[6,16] While all the algorithms performed admirably in a number of case examples, three of the algorithms outperformed the humans in all the test cases. The best of these methods for facial recognition use ML algorithms on extremely large data sets.

One goal of true AI is to produce a machine that can understand written and spoken language. Such a machine would approach, and perhaps pass, the Turing test. Garnering meaning from words and sentences, as opposed to merely being able to recognise their occurrence, is an extreme challenge – as those who are talented enough to have learned a second language will attest. Consider the simple sentence;

"I never said he stole my money"

Depending on the emphasis, this sentence can have seven different meanings;

I never said he stole my money – Somebody else said it.
I ***never*** said he stole my money – I didn't say it.
I never ***said*** he stole my money – I implied it.
I never said ***he*** stole my money – Somebody did, maybe him.
I never said he ***stole*** my money – It was only borrowed.
I never said he stole ***my*** money – It was stolen from someone else.
I never said he stole my ***money*** – He stole something else.

And this is even before we get to truly crazy sentences such as *"Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo"* using three

meanings of the word buffalo to construct a grammatically allowed sentence from eight repeats of the same word!

To assist computers in this task, we break sentences into parse trees – diagrams that represent sentences and their structure.[6] As some sentences have more than one meaning, they also have more than one parse tree. A key breakthrough here was to use probabilities to identify which meaning is most likely in the given context. This, in turn, allows rules to be established as to which combinations of words are allowed in which orders, and hence which meanings are most likely from given sentences. The huge number of rules that would require coding make writing such a rulebook impossible. Luckily, the vast array of written words catalogued through human history make this a perfect task for the algorithms covered in the next section – ML.

1.3 What Is Machine Learning?

ML is a subcategory within the field of AI and specifically pattern recognition. ML algorithms are computer programs that, through the inspection of a data bank, and without explicit instruction from a human programmer, learn to carry out specific tasks. What kinds of questions can these algorithms answer?

Many modern AIs, including ML algorithms, require a large amount of data for their training. They also require well-defined questions, as they do not currently have the capacity for independent thought. This will be further explored in the next section, but I want to delve into a couple of examples from the world of ML to set the scene for this.

Firstly, I will discuss some of my own research, which includes the development of ML algorithms. Chemicals can have a range of effects in the human body. Some of these are desirable (such as the therapeutic effect of a medicine), whereas others are not. Undesirable effects are investigated in toxicology studies, which are required before a new drug can be prescribed, a new hair shampoo released, or a new pesticide used on crops. The primary aim of these studies is to ensure that the new chemicals are safe for humans and the environment, taking into account the amount of

chemical that is used (the dose or exposure). Traditionally, toxicology studies were conducted using animal experiments, and an estimated safe dose for humans was extrapolated from these tests. However, animal testing is expensive, time-consuming, unethical, and humans and animals have different biology that can impact the results. Consequently, there is now a major drive in toxicology to look at methods that do not require animals; computational toxicology is one of these areas.

In my research, I aim to understand how the interaction of chemicals with proteins and enzymes in the human body might lead to a toxic outcome. Using computer models, new chemicals can be assessed before any experimentation is carried out. This mechanistic toxicology approach, based on understanding if and how interactions happen that lead to toxic outcomes, is an effective approach to toxicological problems. These rapid and cost-effective computational results can be followed up with targeted *in vitro* experimental assays on cells to provide further evidence. By making use of advanced new computational techniques, including ML, toxicology can become a more efficient, ethical, and reliable science.

To construct any kind of computational model, experimental data are required. Luckily numerous biological experiments on cells have been conducted, with widely disseminated results. Databases such as ChEMBL (https://www.ebi.ac.uk/chembl/)[17] and ToxCast (https://www.epa.gov/chemical-research/toxicity-forecasting)[18] collate much of this data and are open source. This data can provide the basis for models.

Initial modelling approaches included the use of structural alerts, coded fragments of molecules that are found more commonly in experimentally active molecules than inactive ones.[19,20] While this procedure is relatively simple, it is popular among toxicologists as it allows them to easily identify the parts of a molecule that have caused it to be considered active by the computer. The procedure is extremely transparent. Furthermore, the toxicologists can also look at other molecules containing the fragment in a process known as read-across, in which the experimental results for an existing chemical are used for a new, untested, chemical when an argument can be made that the two molecules are similar enough. In the structural alert approach, a computer program is used to identify the common

chemistry in the training data to be built into a model. If a new chemical contains a known active chemical feature, it is predicted as active by the algorithm (see Figure 1.6).

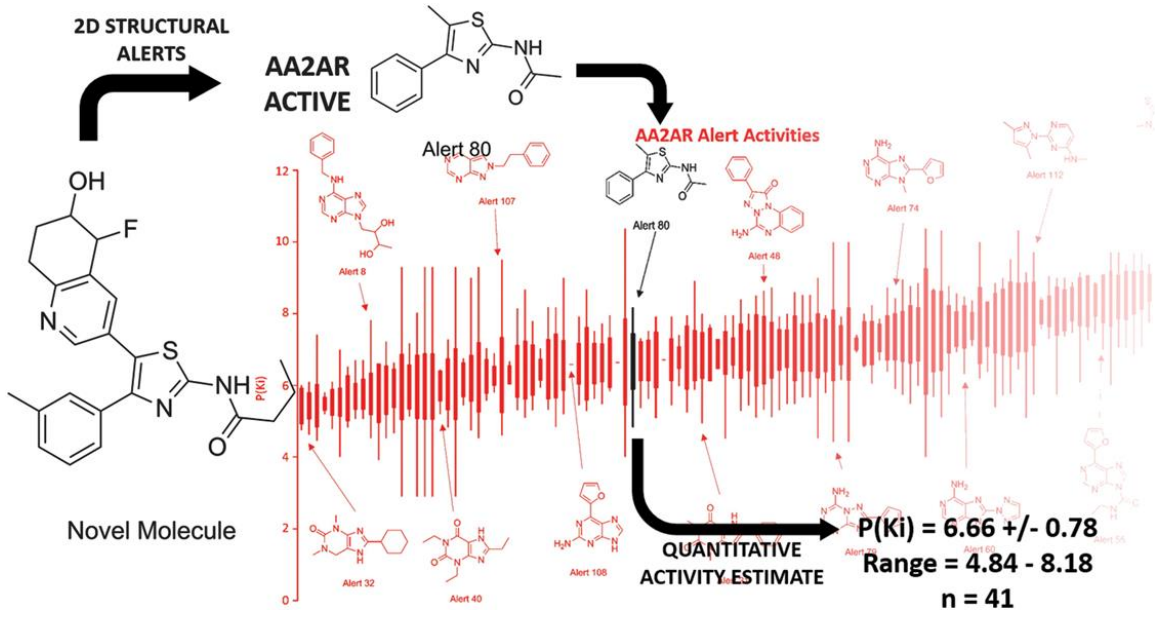


Figure 1.6 The use of 2D structural alerts to classify novel molecules to predict their biological activity. Reproduced from ref. 20 with permission from Oxford University Press, Copyright 2018.

In a second modelling approach, quantum chemistry calculations were used as discussed earlier in the computational science section. Data were obtained for the Ames mutagenicity assay, and this was modelled by considering the activation energy for the reaction between the chemical toxicants and methylamine.[21] The methylamine is a substitute for DNA in the calculation, as the Ames assay measures mutagenicity, and one way a chemical can be mutagenic is *via* a direct chemical reaction with DNA. Several experimentally tested chemicals were considered, and their activation energies were calculated computationally. The results showed a good correlation with the experimental results – with higher activation energies being calculated for Ames negative compounds and lower ones for Ames positive compounds (see Figure 1.7).

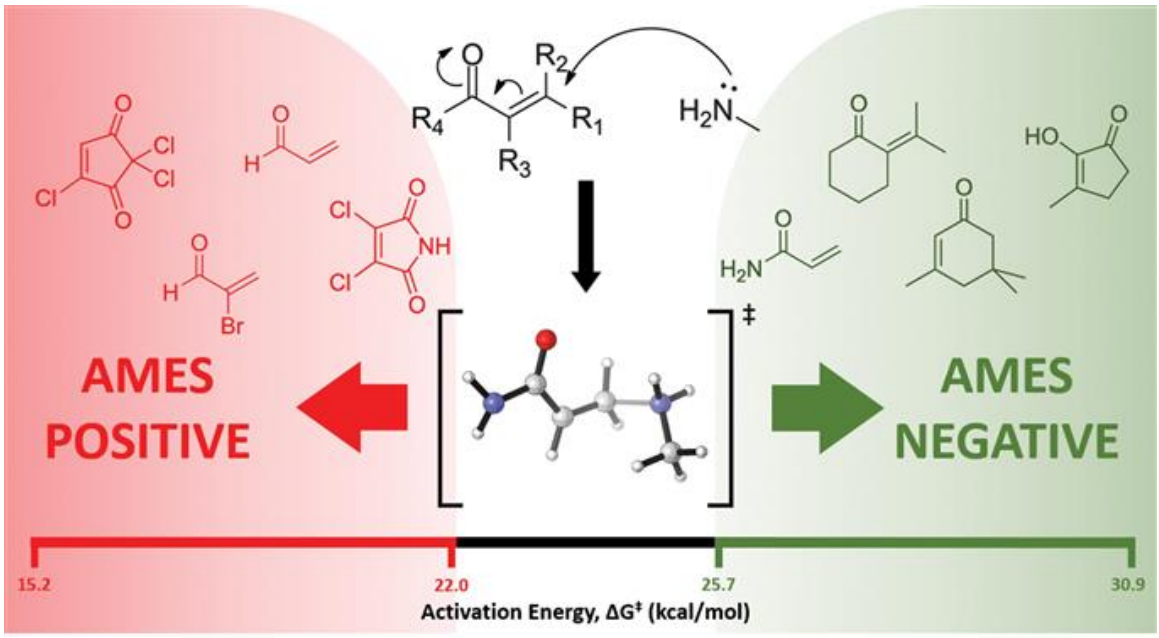


Figure 1.7 Calculated activation energies can be used to classify molecules as Ames positive, negative, or unknown. Reproduced from ref. 21, https://doi.org/10.1021/acs.jcim.8b00130, with permission from American Chemical Society, Copyright 2018.

This is as expected: the Ames negative compounds have high activation energies and hence cannot react with DNA to cause mutations under normal biological conditions. The research allowed us to establish activation energy thresholds for chemicals that were likely to be Ames positive, negative, and unknown (in the middle, uncertain area) that can be used in the future to classify new chemicals.

ML neural networks are also used. These have involved the use of the TensorFlow algorithm in Python 3 to classify molecules in a similar fashion to structural alerts (see Figure 1.8).

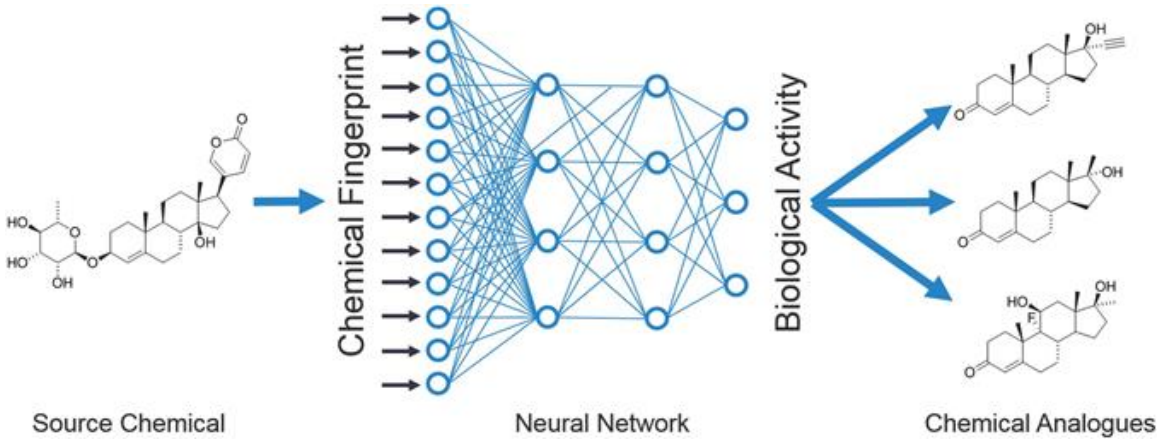


Figure 1.8 Neural networks can be used to make predictions of molecular biological activities.

Open source receptor-binding data are used to train neural networks to classify molecules as binders or nonbinders. These networks are extremely powerful predictors and show a high level of prediction accuracy, outperforming the structural alerts. The networks are also able to provide a confidence estimate of how well a new molecule matches what it knows to be a binder or a nonbinder, which can be treated as a measure of certainty in the prediction.

A disadvantage of using ML algorithms in toxicology is the difficulty in understanding why a specific prediction has been made. Unlike structural alerts, a trained neural network does not point out the parts of the new molecule that make it active. This is a challenge we have been looking to overcome using similarity calculations and examining which molecules the network "thinks" about in the same way, to identify chemical analogues during the prediction process.

A second, exciting ML example dates from the beginning of 2019.[22,23] DeepMind, a leading AI research lab, and Blizzard Entertainment, a computer game development company, released a video demonstration on YouTube of AlphaStar, an AI, battling against professional human gamers in the game StarCraft II. DeepMind already had experience in building AIs to play games against humans in both Chess and Go, but StarCraft II is an entirely different challenge and shows how powerful AIs have become, and how much scope there is for them in the future.

Why is StarCraft II a more complex game than Chess or Go? Well, StarCraft II is a real-time strategy (RTS) game, meaning that players do not take turns, but are instead expected to strategically outwit their opponents and execute their moves in real time. In addition, StarCraft II is more complex than Go or Chess, with considerably more potential moves available at any given time and more moves to be made per game. Finally, the players have imperfect information, as they cannot directly observe the whole game area at all times, so it is very challenging to analyse what other players might be doing. This is obviously very different from Go and Chess where the players can always observe the whole board. As a result of these factors, success in an RTS game can be seen as a step towards a more general AI, which is able to solve a variety of problems like a human.

It is worth noting for context that in this demonstration, StarCraft II was slightly simplified to assist the training and operation of the AlphaStar algorithm. Only a single game map was considered (whereas human players are expected to play across a variety of different maps) and only a single faction, Protoss, was used (whereas RTS games tend to have a number of different factions to play as, all with their own strengths, weaknesses, and suitable strategies). This made the game slightly easier for AlphaStar to compete with the human professionals, in terms of the complexity of the full StarCraft II experience. That said, AlphaStar was also handicapped in the demonstration. A computer might be expected to have an advantage over a human player by being able to make faster moves during the game. Unlike a human player, AlphaStar does not have to click or press buttons to execute its commands. DeepMind limited AlphaStar to a number of actions-per-minute lower than that of a typical StarCraft II professional player.

AlphaStar itself is a neural network ML algorithm. The algorithm observes the game space, and its imperfect information, in a manner analogous to a human player and then makes decisions on what moves to make. While it perceives information, AlphaStar uses a neural network called a long short-term memory (LSTM)[24] to make decisions (see Figure 1.9).

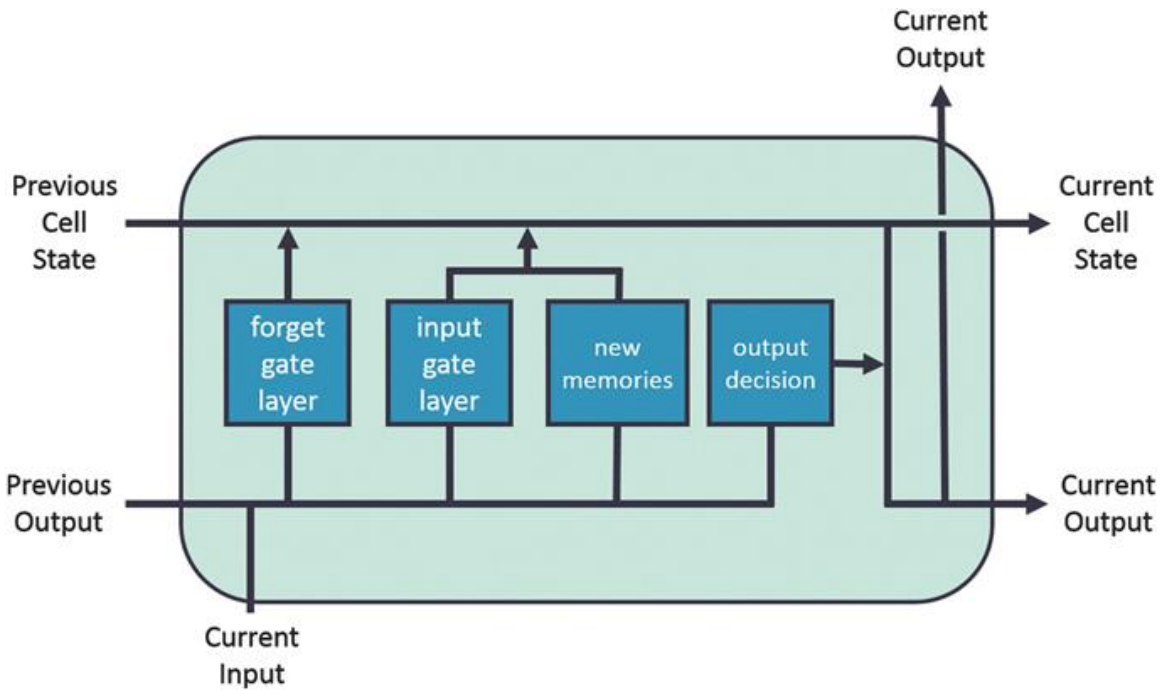


Figure 1.9 A simple graphic showing an LSTM neural network. An LSTM is a combination of four machine learning algorithms (shown in blue boxes). These algorithms can be thought of as a cell that makes the overall decision. These use the current input, previous output, and previous cell state to decide; (i) what previous information to forget (forget gate layer), (ii) what needs to be remembered (new memories) and where this information goes (input gate layer), and (iii) what current decision should be made (output decision). The current cell state and current output are then used again as the process restarts itself. This ability to remember and forget makes LSTMs great at tasks involving continuous thinking, such as games and language recognition.

LSTMs are ideal for tasks when decision-making changes with time, as it allows AlphaStar to remember its previous decisions and thought processes to impact its current move. The LSTM then makes a decision to execute an action at a specific location, *i.e.* to build units or attack the opponent. Finally, AlphaStar considers how the overall game is going in terms of a win prediction to feed into its strategy. For example, if AlphaStar thinks it is winning, it is more likely to attack and try to finish off the opponent. This is one of the most difficult things for a human player to do, and hence it is very impressive that AlphaStar is able to consider this.

So, before the demonstration, AlphaStar needed to be trained to play StarCraft II. Its training began with an initial setup phase where human *vs* human games were used to extract the basic moves of StarCraft for

AlphaStar to imitate. Once the algorithm had these basic moves, it was used as the basis for the AlphaStar League, an internal competition between different algorithms to establish the strongest ones. In each iteration of the League, the strongest algorithms were kept and slightly altered and the weakest discarded (which is the sort of approach used in the AI technique known as genetic algorithms). This allowed the algorithms to discover new strategies and attempt a variety of different tactics. Over many iterations, the best algorithms win out, and these are the ones that played against the professional players in the demonstration. Through the seven-day training procedure AlphaStar was put through in the AlphaStar League, it was estimated that it had played the equivalent of 200 years of StarCraft II for a human pro!

Now, I don't want to spoil the demonstration for those of you who have not seen it, but I can say that AlphaStar performed admirably against human professional players, and the pros and commentators who took part in the demonstration were impressed with the AI. During its training League, AlphaStar was also able to invent new strategies that would be considered unconventional for a human player to choose and sometimes it made in-game, real-time decisions, which baffled experienced commentators but turned out to be excellent choices. I cannot recommend checking out the demonstration highly enough.

References

1. H. Kragh, *Phys. World*, 2000, **13**(12), 31–36.
2. J. S. Rigden, *Phys. World*, 2005, **18**(4), 18–19.
3. H. A. Medicus, *Phys. Today*, 1974, **27**(2), 38–45.
4. M. Brooks, *The Times Literary Supplement*, 2017, https://www.the-tls.co.uk/articles/public/erwin-schrodinger-misunderstood-icon/.
5. Y. Lam, M. N. Grayson, M. C. Holland, A. Simon and K. N. Houk, *Acc. Chem. Res.*, 2016, **49**, 750–762.
6. N. J. Nilsson, *The Quest for Artificial Intelligence: A History of Ideas and Achievements*, Cambridge University Press, 2010.
7. W. A. Clark and B. G. Farley, *Proceedings of the March 1-3, 1955, Western Joint Computer Conference*, 1955, pp. 86–91.

8. J. McCarthy, *Philos. Log. Artif. Intell.*, 1990, 161–190.
9. A. Newell, *Proceedings of the March 1-3, 1955, Western Joint Computer Conference*, 1955, pp. 101–108.
10. O. Selfridge, *Mechanisation of Thought Processes*, 1959, 511–531.
11. L. N. Kanal and N. C. Randall, *ACM '64: Proceedings of the 1964 19th ACM National Conference*, 1964, pp. D2.5–1–D2.5–10.
12. L. Breiman, *Mach. Learn.*, 2001, **45**(1), 5–32.
13. W. W. Bledsoe, *Technical Report SRI Project*, vol. 6693, 1968.
14. J. S. Beis and D. G. Lowe, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997, pp. 1000–1006.
15. M. D. Kelly, PhD Thesis, Stanford University, 1970.
16. P. J. Phillips, W. T. Scruggs, A. J. O'Toole, P. J. Flynn, K. W. Bowyer and C. L. Schott, *et al.IEEE Trans. Pattern Anal. Mach. Intell.*, 2010, **32**, 831–846.
17. The ChEMBL Database, (https://www.ebi.ac.uk/chembl/, accessed Sept 2019).
18. ToxCast, (https://www.epa.gov/chemical-research/toxicity-forecasting, accessed Sept 2019).
19. T. E. H. Allen, S. Liggi, J. M. Goodman, S. Gutsell and P. J. Russell, *Chem. Res. Toxicol.*, 2016, **29**, 2060–2070.
20. T. E. H. Allen, J. M. Goodman, S. Gutsell and P. J. Russell, *Toxicol. Sci*, 2018, **165**, 213–223.
21. T. E. H. Allen, M. N. Grayson, J. M. Goodman, S. Gutsell and P. J. Russell, *J. Chem. Inf. Model.*, 2018, **58**, 1266–1271.
22. DeepMind, *Deepmind StarCraft II Demonstration* (https://www.youtube.com/watch?v=cUTMhmVh1qs, accessed Sept 2019), 2019.
23. The AlphaStar Team, *AlphaStar: Mastering the Real-Time Strategy Game StarCraft II*, (https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii, accessed Sept 2019), 2019.
24. S. Hochreiter and J. Schmidhuber, *Neural Comput.*, 1997, **9**, 1735–1780.

CHAPTER 2

How Do Machines Learn?

TIMOTHY E. H. ALLEN[a,b]

[a] MRC Toxicology Unit, Hodgkin Building, Lancaster Road, Leicester, LE1 7HB, UK
[b] Centre for Molecular Informatics, Department of Chemistry, University of Cambridge, Lensfield Road, Cambridge CB2 1EW, UK
Email: teha2@cam.ac.uk

A variety of machine learning (ML) algorithms exist, and a range of key methods are discussed elsewhere in this text. All ML algorithms draw on the power of computational mathematics to spot and exploit patterns in data. As a result, the machines can be considered as mathematical pattern spotters rather than true learners. That said, the kinds of algorithms that have been developed in computer science to exploit this ability are extremely powerful and can be applied to a great number of problems, which is one of the key reasons for the current level of excitement around ML.

In this chapter, I will describe the general procedure that can be used to build an ML algorithm and describe how it is trained. Here, we will be specifically considering artificial neural networks (ANNs), but many of the issues raised are common to a variety of ML algorithms. To help guide the discussion, two examples are considered throughout – an image classification task to recognise hand-drawn numbers, and a quantitative

prediction task, where molecular solubility values are predicted from chemical structure.

2.1 Framing the Question

Computers excel at answering certain types of question but are poor at answering others. Therefore, an essential first step in using ML is to consider the type of question that we are asking the computer.

A common misconception about artificial intelligence (AI) and ML exists – the current methods are not what we would call general intelligence. In other words, they cannot reliably perform tasks outside the range of knowledge on which they have been trained, nor they can display human-like problem solving ability across multiple layers of thought. For example, they cannot break down a problem into smaller steps and then solve each of those individually to reach a conclusion. Rather, current ML algorithms are far better at solving specific problems that are well framed by their human creators. One way to consider if your question is suitable is to consider the "features" and "labels" associated with it (see Figure 2.1).

Figure 2.1 A (very) simple machine learning algorithm.

Simply put, ML features are the inputs and the labels are the outputs. To give an example, in an image classification task, the features are derived from the image and its label describes what the image is. If a question has clearly defined features and labels that are logically linked, even if these links are complicated, it is potentially a suitable area for ML. ML algorithms have shown a remarkable, and rapid, ability to recognise and classify images, now readily demonstrated online. There are simply too many images on the internet to have humans classify every one, so software

is trained on a subset of images that have been labelled by humans. In my work, I train ML algorithms to predict the biological activity of molecules. The molecules are the features, and their biological activity is the label. These features and labels are logically linked, as the chemical structure of a molecule determines its ability to bind to an enzyme or disrupt a membrane. Hence, this question is also suitable for ML.

Many computer scientists are confident that AI will, in due course, start to show general intelligence and be able to handle complex questions and provide correspondingly complex answers. At that point, genuinely interesting conversations with a machine should be routine – breaking through the famous Turing test. For now, though, we use what we have at hand. After all, asking my chemistry-based algorithms to classify pictures wouldn't work at all, and the same is true for asking a machine trained to identify images to predict the activity of new molecules. The questions must be carefully considered, and the machines should be used for the purpose for which they were built.

2.2 Gathering Data

A key aspect of ML is the requirement for large quantities of high-quality data. No algorithmic procedure can overcome issues with training data quality. A prediction is of questionable value if it is made by an algorithm that was trained on unreliable or poor-quality data. If, for example, you are trying to build a model to predict the outcome of a chemical reaction, you want the experiments to have been performed by well-qualified scientists who follow good laboratory practice and use new reagents of high purity. If the data are quantitative, you want that information to be measured precisely with equipment that was recently well-calibrated. It may also be desirable that laboratory conditions such as temperature and pressure were consistent, or at least recorded, while all the experiments were being conducted. Obviously, the chances of a single dataset conforming to all the above statements are extremely low. Even when the experimental procedure was of high quality, some uncertainty in the data is almost unavoidable; dealing with this uncertainty is challenging. Data points can be checked

manually for consistency, considering the experimental procedures used, the lab in which the experiments were run, and reputation of those running them, but this quickly becomes an overwhelming, and extremely tedious, task.

The volume of data required for hungry ML systems may also be substantial. Generally, ML algorithms require more data points than traditional applications of statistics, such as quantitative structure–activity relationships, because the former are more complex in their mathematical construction. However, all methods benefit from larger data sources as this allows their application to be better assessed across a larger number of scenarios. Luckily, modern techniques, such as high throughput screening (HTS), allow for many data points to be generated relatively quickly. HTS uses robotic and automated experimental procedures to allow large numbers (sometimes on the scale of millions) of experiments to be performed rapidly and consistently. These procedures are popular in drug discovery where the assessment of the ability of new potential drugs to activate or inhibit biological targets can yield their binding coefficients and assess their likely therapeutic activity. Taken together, a balance often is to be sought between having a large number of data points for ML model construction and using only the highest quality data available to ensure that the algorithms produced can be trusted.

A final important consideration is how machines perceive data. As humans, we perceive information in a variety of ways, which we often take for granted. Current computers are not so lucky, and we need to feed data into machines in a way they can understand. This often involves reducing data into strings of numbers that machines are excellent at handling. Consider the example of interpreting images of handwritten numbers, a classic example that has been tackled in ML numerous times. Humans can easily recognise the numbers in Figure 2.2. But to feed these images into a machine, we need to break down each image into pixels and affix a value to each pixel, denoting its grayscale value in the image (see Figure 2.3).

Figure 2.2 Handwritten figures are instantly recognisable to the human eye.

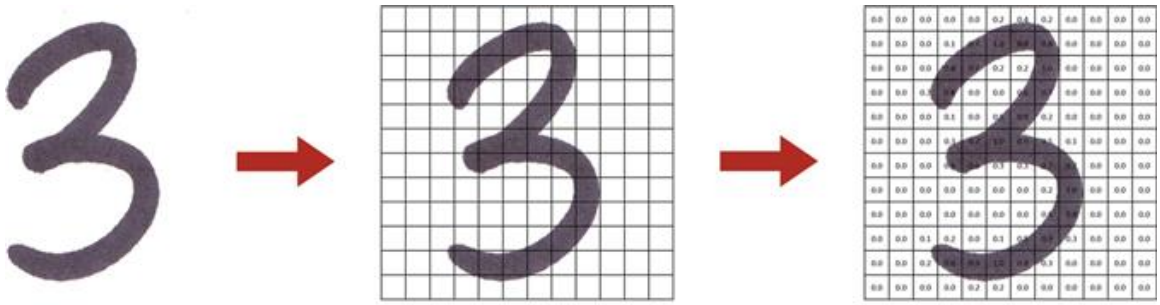


Figure 2.3 A handwritten number can be broken down into computer-readable values. In this example, a 12×12 grid is applied over the number to generate 144 values ranging from 0 to 1 depending on the amount of black in each one.

But how is this relevant to chemistry? All chemists are familiar with chemical structures presented in a similar manner to the one in Figure 2.4. Instead of a drawing, the chemist could describe the main features of the chemical over the phone, or in an email, but such descriptions are perhaps less precise than we would like, and certainly not precise enough for ML. “Chemical fingerprints” are often used to overcome this problem. Chemical fingerprints consider the presence of chemical features in a molecule and convert these into “on” bits in a bitstring. This can be done using features that would be easily identifiable to a human, such as the presence of a particular atom or functional group in the MACCS keys, or using a circular fingerprinting algorithm. Extended Connectivity Fingerprints (ECFPs) are one class of these, and they can be quickly calculated computationally based on individual atom environments (see Figure 2.5).

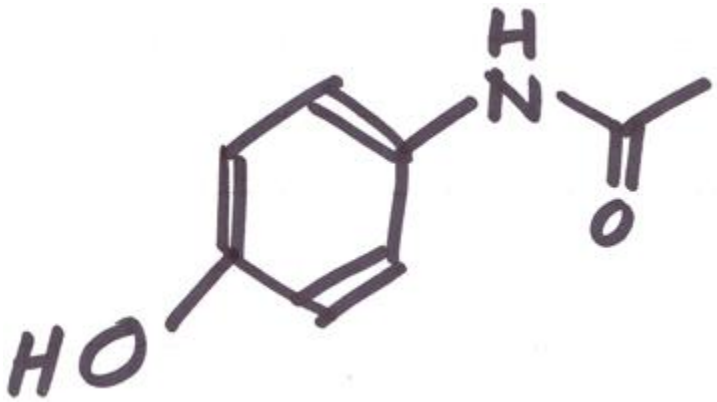


Figure 2.4 A hand-drawn chemical structure which a human chemist can easily interpret. A human chemist may recognise this molecule as paracetamol.

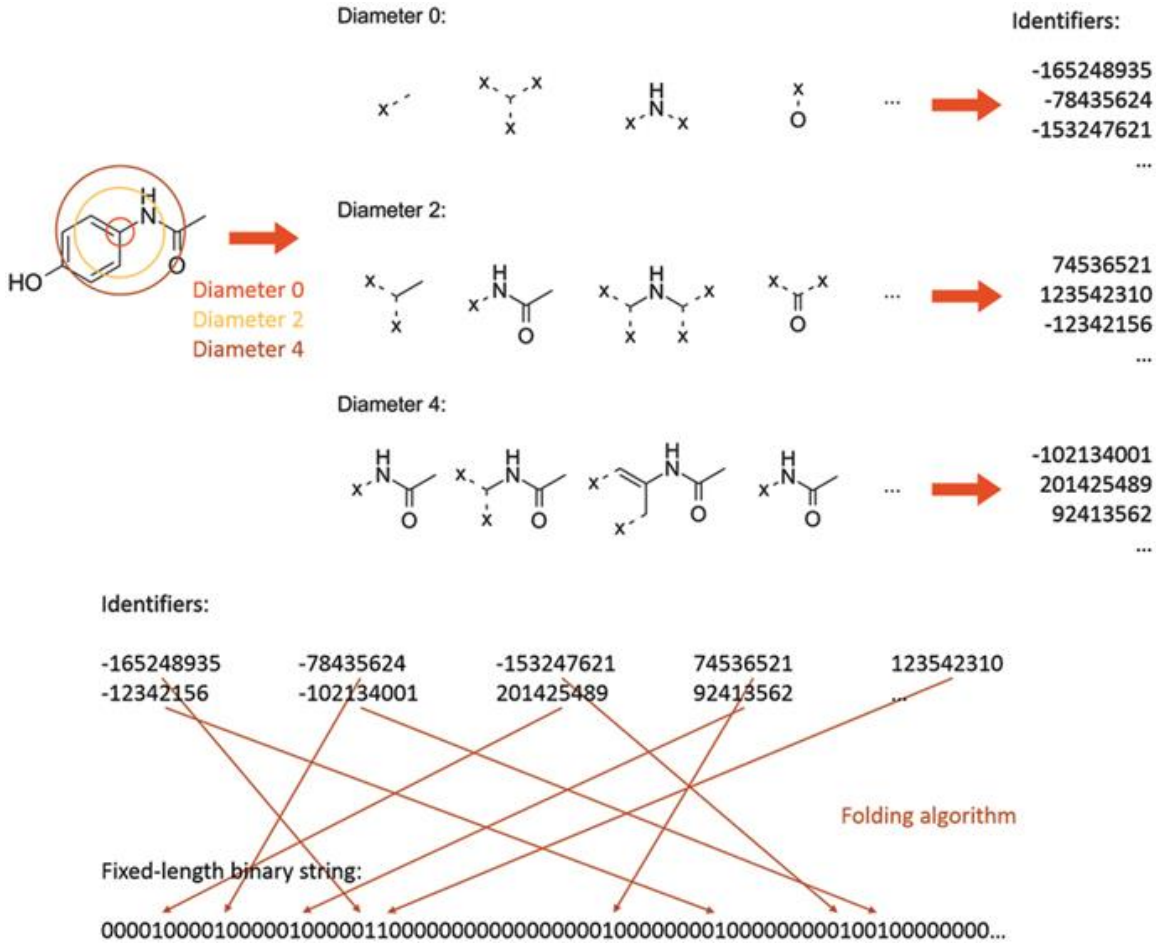


Figure 2.5 The paracetamol molecule broken down into a set of identifiers, which can be computationally "folded" onto a bit string of a predefined length.

The user selects a radius, and each atom environment, along with its neighbours within that radius, is converted by the computer into an integer identifier. The user also selects a length for the output fingerprint, and the computer hashes the identifiers onto a fingerprint of the fixed-length that the user selected. This is useful for ML as the algorithm will always output the same length string, providing consistency no matter which molecule has been entered. Extended connectivity fingerprints are commonly used in medicinal chemistry and to evaluate the similarity between two molecules by considering the number of "on" bits the molecules share in a procedure such as a Tanimoto similarity calculation. They are not perfect, however, as decoding an ECFP to find the parent molecule is not possible as various different functionalities can be hashed onto a single bit. This is known as a bit collision (see Figure 2.6).

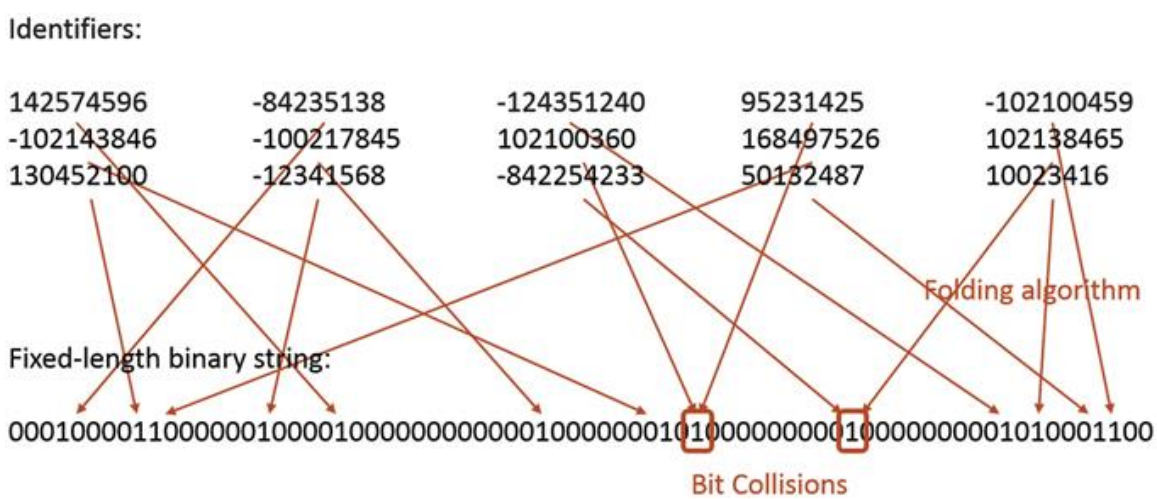


Figure 2.6 For larger molecules and shorter bit string lengths multiple identifiers can be folded onto the same bit, leading to a collision and a loss of chemical information.

Once a database has been obtained for the machine to be trained on, an appropriate algorithm can be chosen for the task and set up for training.

2.3 Setting up the Algorithm

Artificial neural networks are a subclass of ML algorithms based on the workings of animal brains. When you or I learn new information, connections are made between biological neurons in our brains that allow subsequent recall. Connections between biological neurons are known as biological synapses, and they pass the signal across from one neuron to another, given sufficient electrical "push" to get the signal across. These electrical signals jumping through our brains allow us to think and learn, and these are the reasons our brains "light up" in certain areas when we are considering new chemical problems, experiencing pain, or falling in love. Neuroscientists can measure these electrical signals using a technique known as electrophysiology to identify which neurons or set of neurons are active under specific conditions.

Artificial neural networks mimic this process using artificial neurons and artificial synapses (henceforth simply neurons and synapses). The neurons are organised in layers and can be considered as placeholders for numbers. The synapses connect each neuron with all the neurons in both the preceding and the following layers with a mathematical relationship. The first layer of neurons is known as the input layer, the last is the output layer, and any layers in between are the hidden layers (see Figure 2.7).

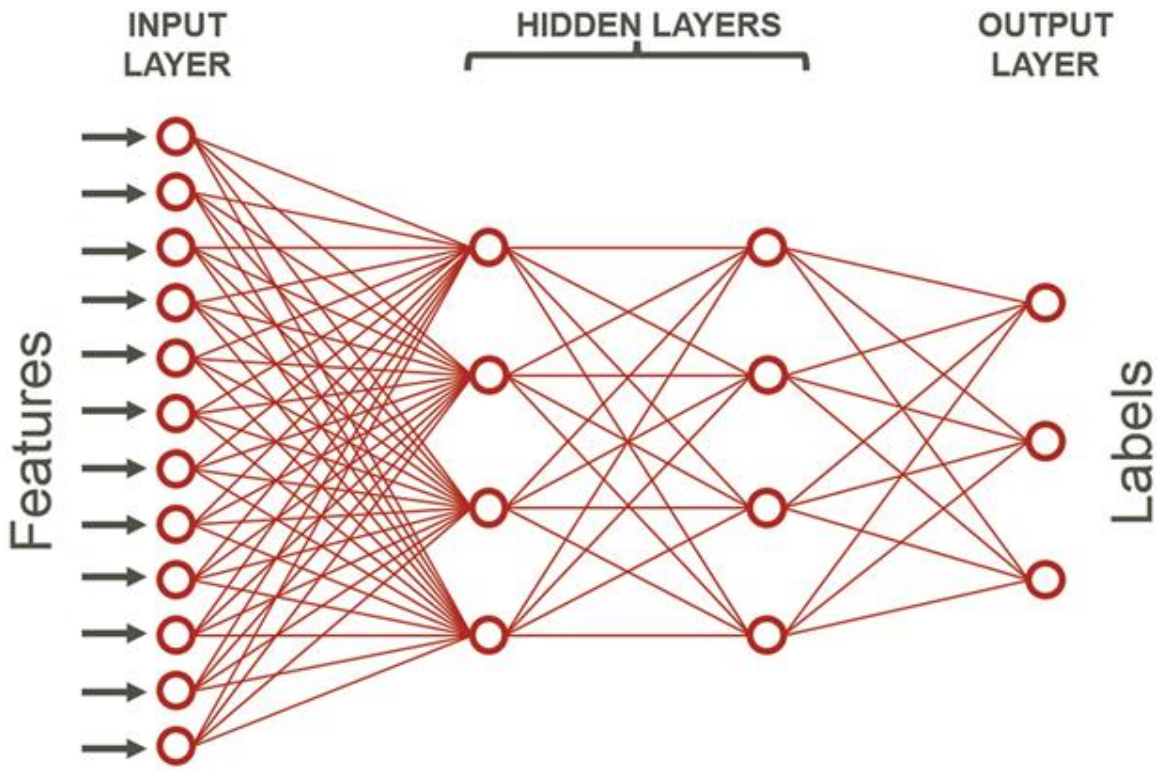


Figure 2.7 A simple neural network showing the input, output and hidden layers of neurons connected by synapses.

Data are fed into the network through the input layer, and predictions or decisions are revealed by the output layer. In our image classification task, each pixel is assigned a number according to its grayscale. This list of numbers constitutes the inputs. The outputs would be labels associated with the numerical value each image represents (see Figure 2.8). In chemistry, we may wish, for example, to train a neural network to predict the intrinsic solubility of chemicals based on their chemical structure. The chemical structure in the form of a chemical fingerprint can be entered as our list of features, and our labels would be experimentally measured intrinsic solubility values.

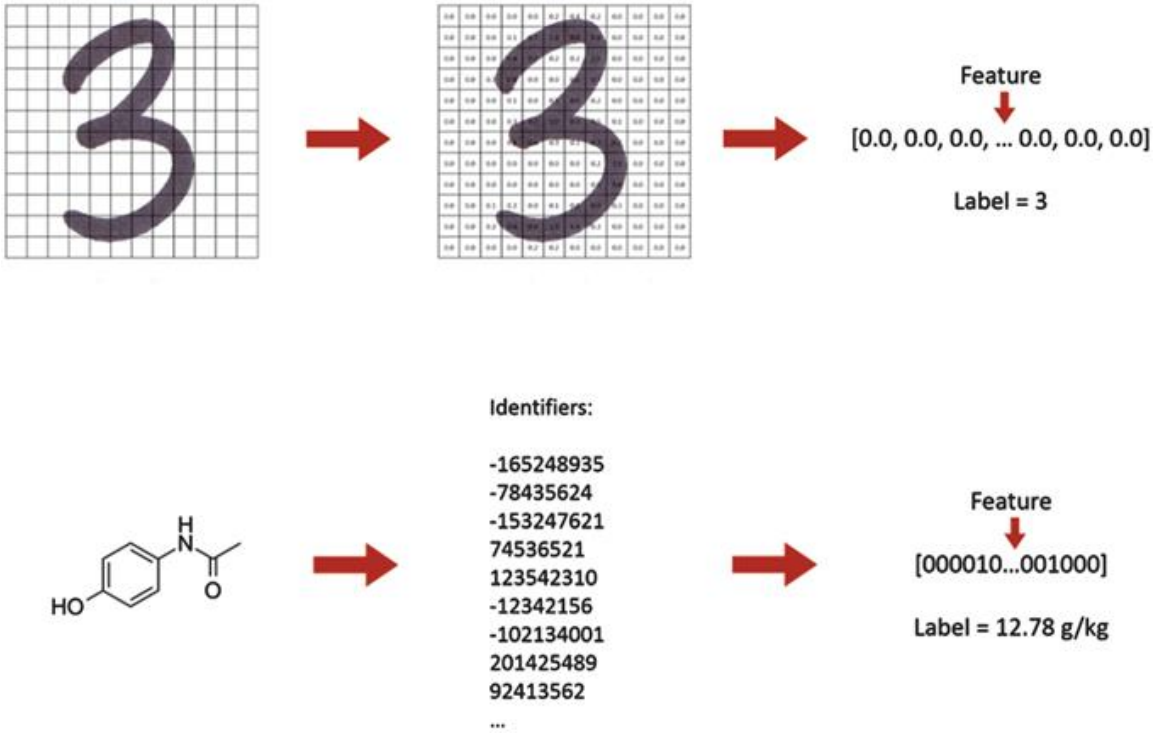


Figure 2.8 How different features are generated from handwritten numbers and chemical structures.

Before training can commence, a few important decisions have to be made, specifically concerning the architecture of the network being constructed. Neural networks have one or more hidden layers, each containing a number of neurons. How do we know how many layers to have? How many neurons should each layer consist of?

The answers to these questions are not trivial. In fact, an ML engineer will often consider a number of different network architectures to establish which is best, using trial and error. Generally, more complex tasks, with larger input and output layers, high scientific complexity, and large training datasets will require deeper (more hidden layers) and wider (more neurons per layer) networks. This is not always true though, and shallower and narrower networks train more rapidly and, therefore, are faster to reach a solution. These networks are also less prone to overfitting, a phenomenon we discuss later. Because of this, ML engineers try to abide by Occam's razor, the idea that simpler solutions are more likely to be correct than complex ones. In ML, the search for Occam's razor involves trying a

number of parameters and choosing the simplest set which provides a good model.

Once the numbers of neurons and hidden layers have been set, it is time to consider what the synapses between neurons do. I have already mentioned that the synapse is a mathematical relationship, and the simplest of which could take the form:

$$y = mx + c \tag{2.1}$$

which in ML is conventionally written:

$$y = wx + b \tag{2.2}$$

where *w* is the weight, and *b* is the bias. During training, the ML algorithm changes the weight and bias values with each training step in a way that, one hopes, helps it to "learn" patterns in the data provided. However, if all the synapses are linear functions, then the network will be unable to learn. Almost every meaningful problem to which we might wish to apply ML is non-linear, but this cannot be represented by a linear combination of linear functions. To overcome this issue, at each neuron, there are two processing steps: the sum of all incoming synapse signals is calculated, and then the result is put through an "activation function" to determine the output from the neuron. The use of a non-linear activation function is a key reason neural networks can learn and predict non-linearity in data. Some examples of activation functions used in neural networks include the Sigmoid, Rectified Linear Unit (ReLU), and TanH functions (see Figure 2.9).

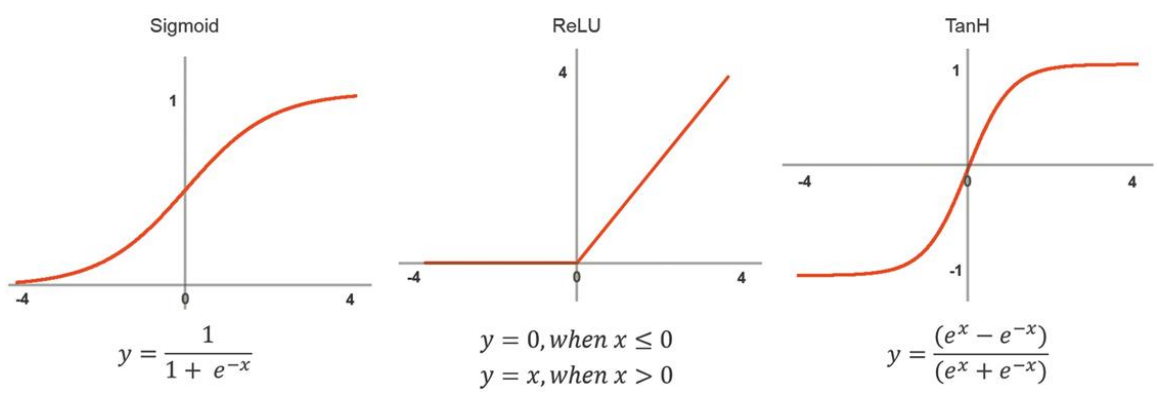


Figure 2.9 Three widely used activation functions, Sigmoid, ReLU and TanH.

The number of useful activation functions is modest, so the choice between them is often performed using trial and error. Sometimes, the same activation function is used for each layer in the network, while other times different functions are chosen. All allow the network to learn non-linearity in the data, which is the most important consideration.

2.4 The Training Procedure

We have framed our question, selected our data, made it readable by a computer, and set up our network architecture. We are almost ready to press the button and watch our network train. But how does this work?

The first thing to consider is how does the network know how well it is performing. In classification tasks like recognising a handwritten number, the answer is straightforward – if the predicted label (produced by the network given the input) matches the observed label (the label attached to the input in the dataset), then the prediction is correct. Otherwise, it is incorrect. This is modelled computationally using a cost (or loss) function. The simplest cost functions tally how often the network is wrong and add up the cost associated with each wrong prediction. In this case, a perfect predictor would, therefore, score zero, and our objective during training is to minimise the cost function.

Now you might see why training is iterative. When trying to find the best possible model, the algorithm will choose a set of weights and biases and calculate the cost function. This gives a measure of how it is performing, allowing it to make changes to the weights and biases. It can then recalculate the cost function and try again. Each iteration is one training step towards a machine that has "learned" patterns in the data it is given.

Now consider how the value of the cost function changes as we adjust the weights and biases within our network. If we consider only having one parameter (w), we could prepare a plot like Figure 2.10. In this problem, there is a well-defined minimum where the cost function converges, and the slope is 0. Calculating the cost function for every possible value of w to find the minimum is impractical, and the situation is even worse as the number

of internal parameters increases. A technique called gradient descent is used to overcome this issue. Initially, an arbitrary starting value for *w* is chosen, and the cost function calculated. The gradient descent algorithm then calculates the gradient in the cost function curve at this point, telling the ML algorithm which direction is closer to the optimal solution. Where there are multiple weights and biases to consider, the gradient is a vector of partial derivatives with respect to these parameters. The gradient descent algorithm then takes a step in the direction of maximum negative gradient to reduce the value of the cost function as quickly as possible. The weights are updated, more training data run through the network, and a new value of the cost function is calculated (see Figure 2.11).

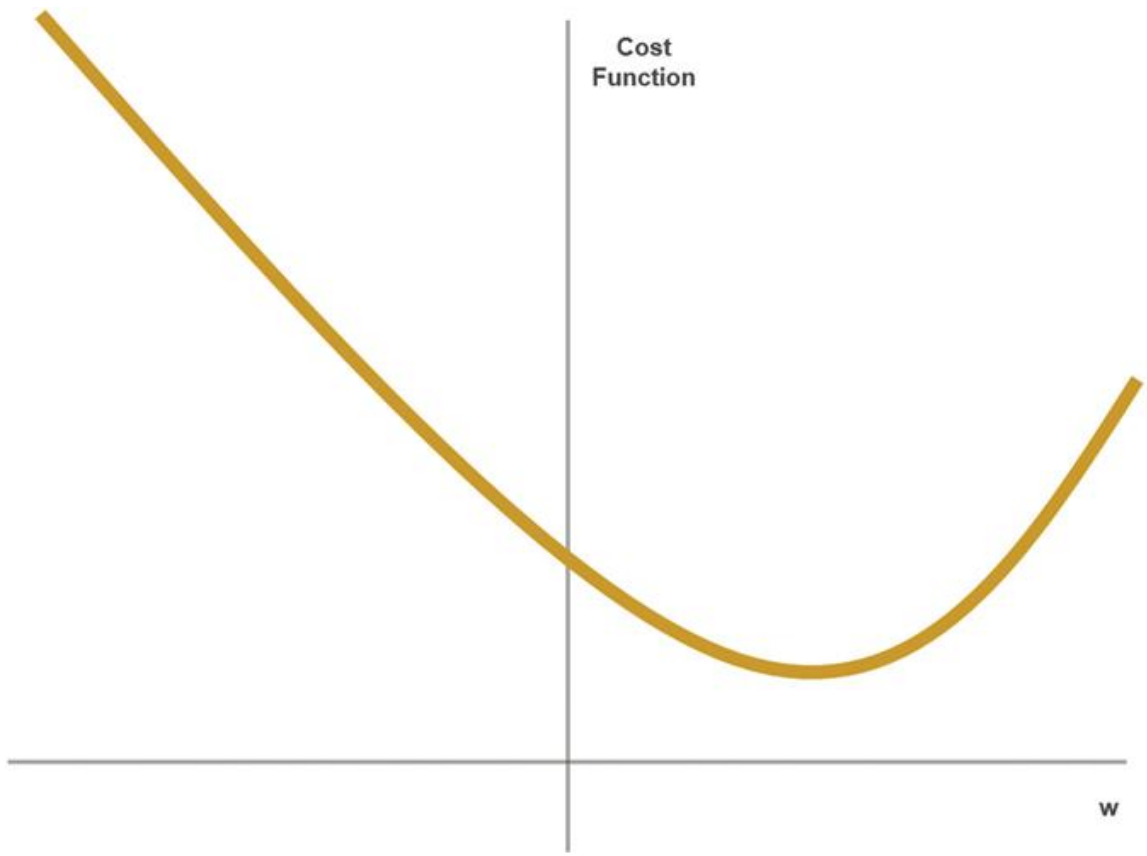


Figure 2.10 The variation of the cost function with a single changeable parameter (*w*).

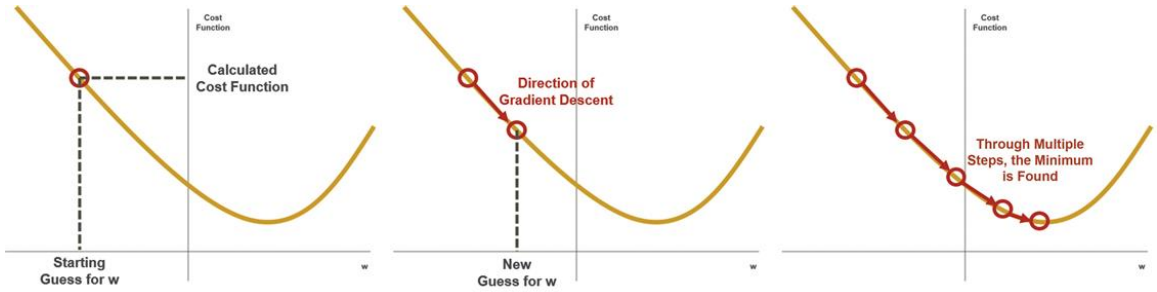


Figure 2.11 Variation of the cost function during training.

By repeating this process, the cost function closes in on the minimum. Over the course of the training procedure, which may contain tens, hundreds, or thousands of training steps, the cost function is reduced to a minimum, and the network is trained. Once training is complete, the network will be evaluated to see how well it performs.

The image recognition task example is a classification task, with the algorithm aiming to correctly identify the number. In cases such as these, a given prediction is either correct or incorrect. In this type of classification task, the percentage of correct predictions is known as accuracy and, as you might expect, the best ML classifiers will have the highest accuracy values. For example, consider our predictions for several hand-drawn number 1s, shown in Figure 2.12.

Hand Drawn Figure	Actual Category	Predicted Category	Score
1	1	1	1
1	1	1	1
1	1	9	0
1	1	1	1

Correct Predictions = 3

Incorrect Predictions = 1

Prediction Accuracy = 75%

Figure 2.12 Outcomes from a hypothetical predictor on some hand-drawn numbers, a score for each prediction, with correct predictions being assigned a 1 and incorrect predictions a 0, and a calculated accuracy for these predictions.

In our solubility prediction network, the prediction produced by the network is a number that can be compared to experimentally measured values. The closer these values, the better the predictions are; root mean square error (RMSE) is one way to measure this, and lower RMSE values indicate better models.

$$\mathrm{RMSE} = \sqrt{\frac{\sum(\theta_{\mathrm{pred}} - \theta_{\mathrm{exp}})^2}{n}} \tag{2.3}$$

where θ_{pred} is the predicted value, θ_{exp} is the experimental value, and n is the number of values for which predictions are made. In this example, RMSE becomes the basis for the cost function, and the lowest RMSE gives the best model.

2.5 Overcoming Flaws

Once a model has been constructed, it is time to see how it performs. We have already discussed how classification tasks can be evaluated using accuracy, and quantitative prediction tasks can be evaluated using RMSE. An issue here is how to best make these evaluations.

The data used to train the model is known as the training set. As has already been mentioned, the best training sets contain a large number of reliable data points. But any measure of network performance calculated using the training set has an obvious flaw – the model has already seen these examples and learned from them. Hence, it can be considered as a memory task rather than a predictive task. This means that the machine is not learning how to make intelligent predictions about new examples, but it is simply regurgitating information it has already been given. Therefore, to reliably assess performance, we need to use examples from outside the training set to make this evaluation.

A common way to do this is to use a test set – a subset of the entire database that has been held out of the training procedure. Splitting data like this is called the holdout method. Often a ratio of 80% : 20% or 75% : 25% is used, with the larger set used as the training set. While any split can be used, it is worth noting that making the training set too small will make it difficult to train the model and making the test set too small will make validation unreliable (would you trust a model that scored 100% on a classification task in the test set if the test set only included one example?). Sometimes a third set is also used, called the validation set. In this case, the model is trained on the training set, evaluated to ensure optimal training on the validation set, and finally tested on the test set. This helps to ensure that the final model produced is not biased by samples in the test set.

Now that we have a training set and a test set, and we can evaluate model performance on both sets to see how the model performs. Often this is the point at which a major flaw in training becomes apparent – overfitting. Overfitting is the result of a model that, having been trained to such an extent that it learns all the tiny quirks and peculiarities in the training set, shows optimum performance on the training data, but fails to learn general rules and relations in the training data, so works poorly on the test data.

In Figure 2.13, the model is attempting to classify the circles and triangles into different classes. It is possible to draw a mathematical function to perfectly classify the training set data with an accuracy of 100%. Initially, this model looks ideal, but when the test set is considered, many of the data points are placed in the wrong class, and accuracy here is only 50%. Hence, the model is overfitted. What we can do is to recalculate our model with a simpler mathematical function to avoid overfitting (see Figure 2.14). With this model, training accuracy drops noticeably to 83%, but test accuracy increases to 70%. When models are not overfitted, they tend to have closer performance on both the training and the test sets.

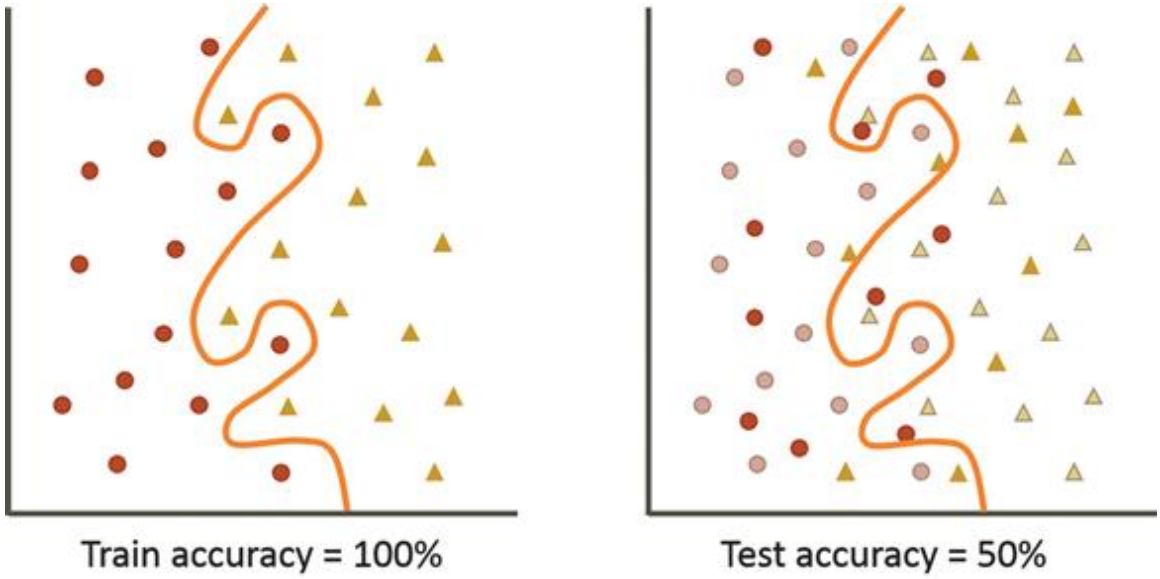


Figure 2.13 An overfitted model classifying circles and triangles. This complex model shows good performance on the training data (left-hand plot) but poor performance on the test data (added in darker colours on the right-hand plot).

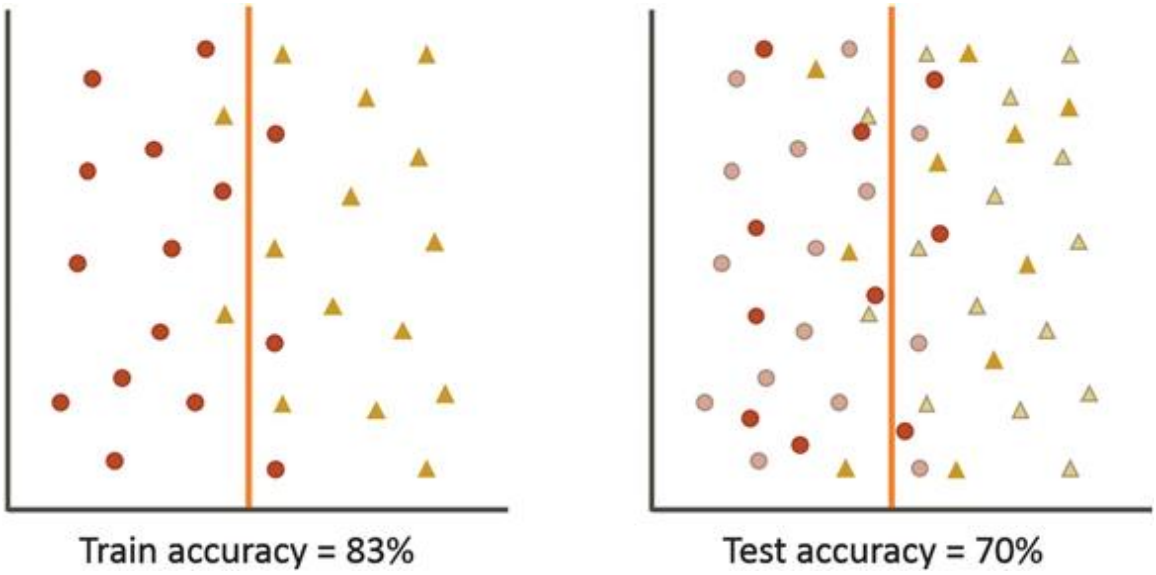


Figure 2.14 A better model classifying circles and triangles. This simpler model shows poorer performance on the training data (left-hand plot) but improved performance on the test data (added in darker colours on the right-hand plot).

This is great, so how do we take it into account in our ML algorithm? A common strategy is early stopping. This is based on the idea that during early iterations of the training procedure, the algorithm learns the more general features of the training data. While during later iterations, more minor details, relating to just a few or possibly a single training example, are learnt.

Consider Figure 2.15. During early training, the algorithm is learning the general features of the dataset, and as a result, training and test accuracies improve. After a certain number of iterations, however, this trend ends and while the training set accuracy continues to increase, the test set accuracy drops off. In this example, if training was stopped early overfitting would be avoided.

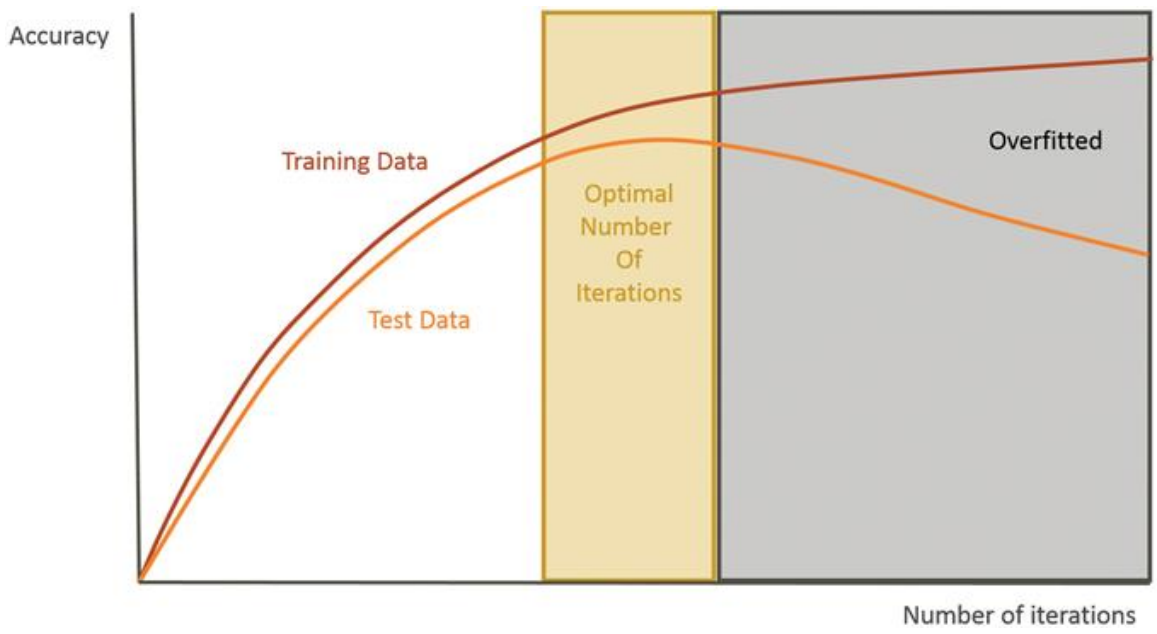


Figure 2.15 The variation of training and test set accuracy with the number of iterations.

Another way to prevent overfitting is to alter the cost function by adding a term to it (the regularisation term) to penalise complexity in the neural network.

Model complexity can be considered as a function of the weights of all the synapses in the neural network. Synapses with high weight values are considered to add more complexity to a model than those with low weights.

L_2 regularisation is a procedure used to quantify this in the regularisation term, by summing the squares of all the weights in the network.

$$L_2 \text{ regularisation term} = \sum_{i=1}^{n} w_i^2 \tag{2.4}$$

where w_i is the weight value for synapse i among a total of n synapses in the network. The effect of regularisation can be tuned by multiplying the regularisation term by a constant, known as the regularisation rate. The regularisation rate must be high enough to prevent overfitting, but not so high that it prevents the model from learning. After all, a model, which is so simple that it cannot learn from the data, is of no use whatsoever. The ideal regularisation rate allows the model to learn on the training set, but also produce correct predictions when applied to previously unseen examples, either in the test set or real world.

When the regularisation term is added to the cost function it now looks like this:

$$\text{Cost} = \text{Loss (Data}\,|\,\text{Model)} + \lambda\ (L_2 \text{ regularisation}) \tag{2.5}$$

where Loss (Data|Model) represents the loss associated with the original cost function (*i.e.* how well our predictions match the observations), L_2 regularisation represents the L_2 regularisation term shown above, and λ is a constant called the regularisation rate, which controls how much regularisation is used in the model training – too little and the model will be overfitted, too much and the model will be too simplistic to accurately reflect the observed data.

Overfitting is not the only issue that can be encountered during training. Let us consider again the issue of gradient descent towards a minimum value of the cost function (see Figure 2.16).

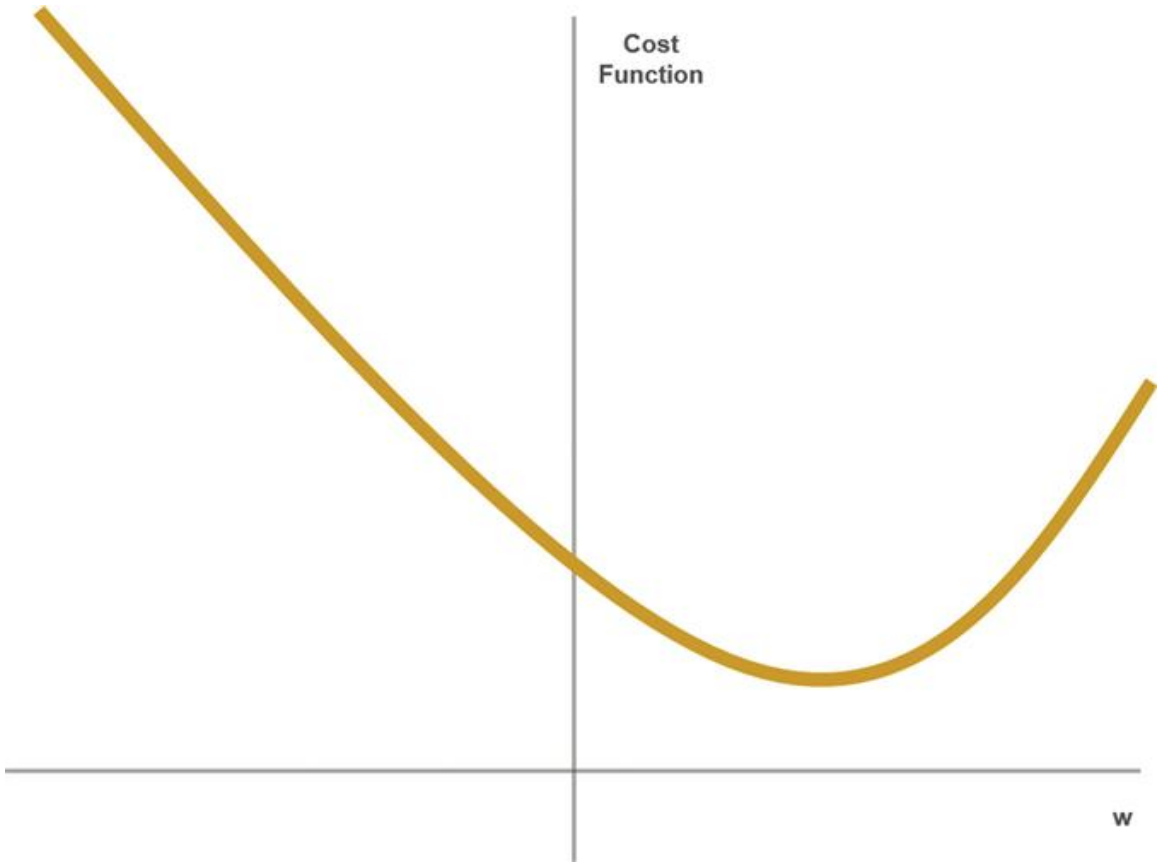


Figure 2.16 A simple cost function with only one minimum.

In this case, we only have one minimum in the cost function surface, which is convenient. However, most surfaces have multiple minima (see Figure 2.17). As I am sure you have already figured out, this could lead to two trained networks with different statistical performances (see Figure 2.18), depending on where on the surface training begins and how large the taken training steps are. This is obviously an issue, as one of these is not the global minimum and hence not the best possible network.

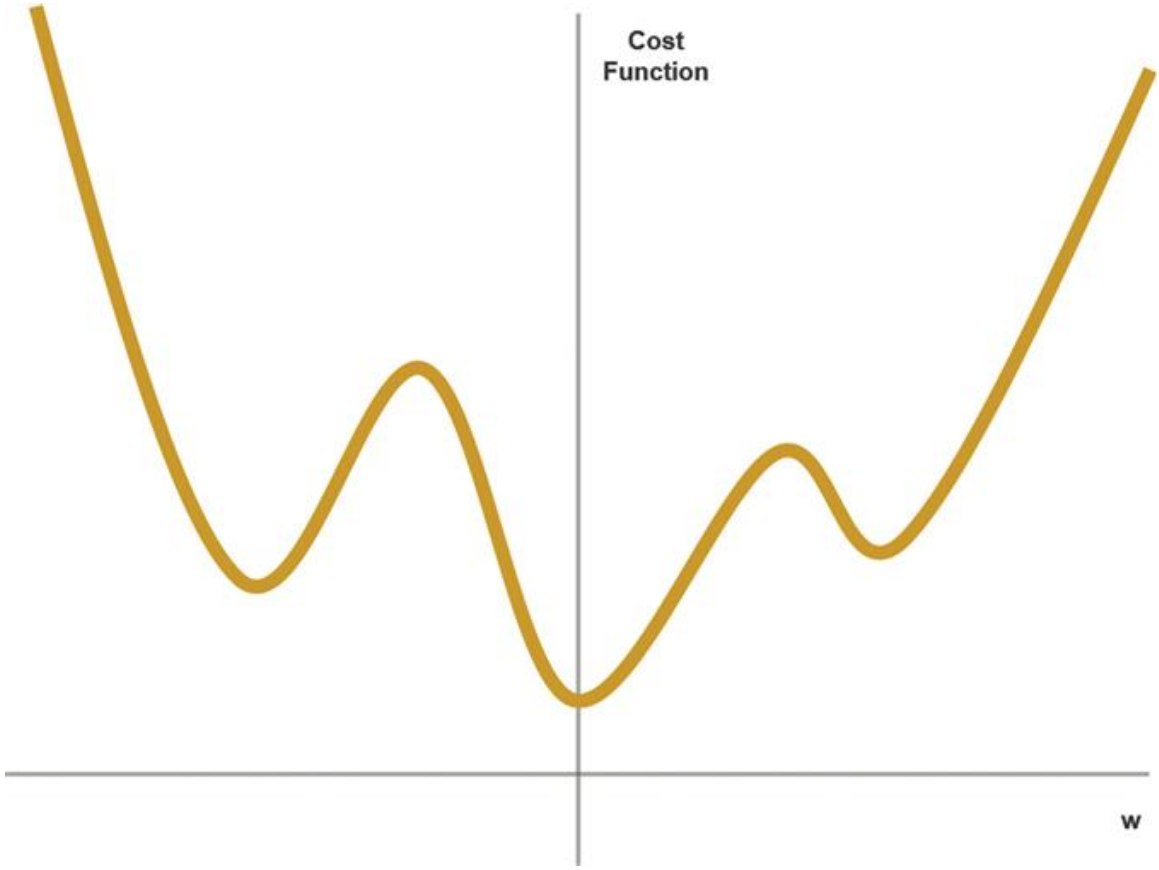


Figure 2.17 A more complex cost function with multiple minima.

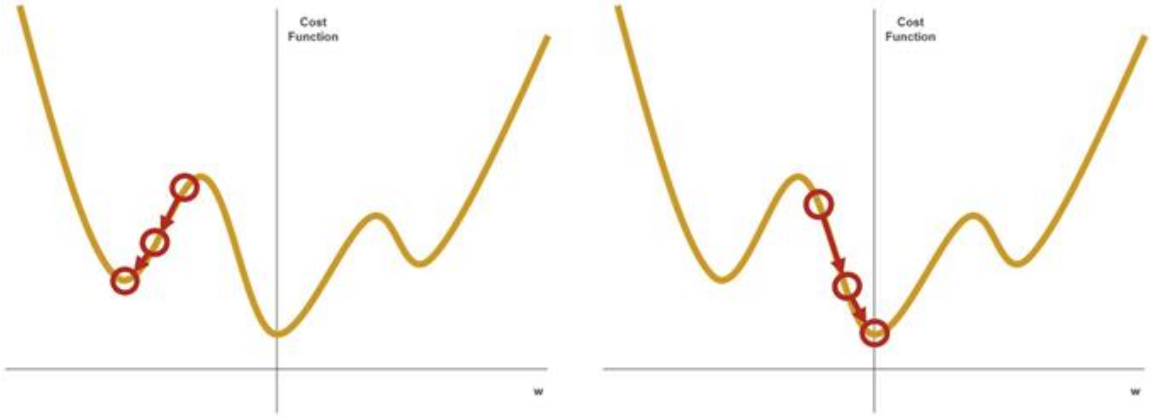


Figure 2.18 On this surface, two descents can find different minima, depending on the start point. Only one is the global minimum – the lowest possible value of the cost function, and hence the best possible model.

As a result of this, it is a common practice to repeat training several times – in the same way, you might repeat an experiment – to ensure that the global minimum is found and that the calculation can be trusted.

2.6 Deploying the Algorithms

So, you've gathered appropriate data and constructed and trained an ML algorithm, taking into account the dangers of data set bias and overfitting. What's next? Well, hopefully, you've constructed a model worthy of being applied to further external data from the real world. In our example models, it might involve classifying further written digits on scanned forms for data entry or predicting physicochemical properties for evaluation during new chemical development.

Just before you head out to change the world, I want finally to go a little further into the subject of model validation and touch on a couple of examples that I think offer valuable lessons.

In computational science, validation can be a touchy subject, and everyone has their favourite and least favourite statistical performance metrics (and during model development, your favourite will likely become the one which paints your model in the best possible light), but they all have strengths and weaknesses. In our hand-drawn numbers classification task, we assigned each prediction as either correct or incorrect, and this allowed us to calculate the accuracy of our predictions. For binary classification tasks, where the actual answer is yes or no, predictions can be classified using a confusion matrix (see Figure 2.19).

Confusion Matrix		Actual Value	
		Positive	Negative
Predicted Value	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Figure 2.19 A confusion matrix for binary classification.

In a classification task, it is common to calculate the number of true positives (TPs), false positives (FPs), true negatives (TNs), and false negatives (FNs). From these, a variety of statistical measures can be calculated including

Sensitivity (also known as true positive rate or recall), the probability an experimental positive is correctly classified;

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{2.6}$$

Specificity (also known as true negative rate or selectivity), the probability an experimental negative is correctly classified;

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{2.7}$$

Positive predictive value (PPV, also known as precision), the probability a predicted positive is experimentally positive;

$$\text{PPV} = \frac{TP}{TP + FP} \tag{2.8}$$

Negative predictive value (NPV), the probability a predicted negative is experimentally negative;

$$\text{NPV} = \frac{TN}{TN + FN} \tag{2.9}$$

These metrics each consider half of the predictions made, so these should always be considered in pairs. However, sometimes one being high and the other being low is acceptable, depending on the modelling scenario. For example, consider if you are searching for a rare bird in a jungle, and you have trained a model to predict the location of the bird within a certain area. A model with high PPV would be useful as it reduces the chance of a wasted expedition to try and locate the bird! A high NPV does not really matter in this case as you do not really care where the bird is not. Let's consider another example where you need to test a large number of water samples to detect, which are contaminated. Your machine can test the samples and predict if they come from contaminated sources. In this scenario, it is extremely important to have a high NPV, as otherwise, this could result in tainted water sources going uninvestigated. A high PPV is desirable, but if you must conduct further tests on some water sources that are safe, that is less bad than having contaminated sources remaining untreated.

Some other statistical measures help with data imbalance, including

Balanced accuracy, which averages the sensitivity and specificity, helping to account for small data imbalances between the positive and negative classes;

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2} \tag{2.10}$$

Matthews correlation coefficient (MCC), a measure of model quality, provides a value between −1 and 1. 1 indicates a perfect model, −1 an entirely incorrect model (*i.e.* every prediction is exactly wrong), and 0 a model thatassigns labels at random. MCC can be used even when classes are very unbalanced;

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2.11}$$

Finally, a popular measure of model performance in ML is to use the receiver operating characteristic (ROC) curve and calculate the area under it (AUC) (see Figure 2.20).

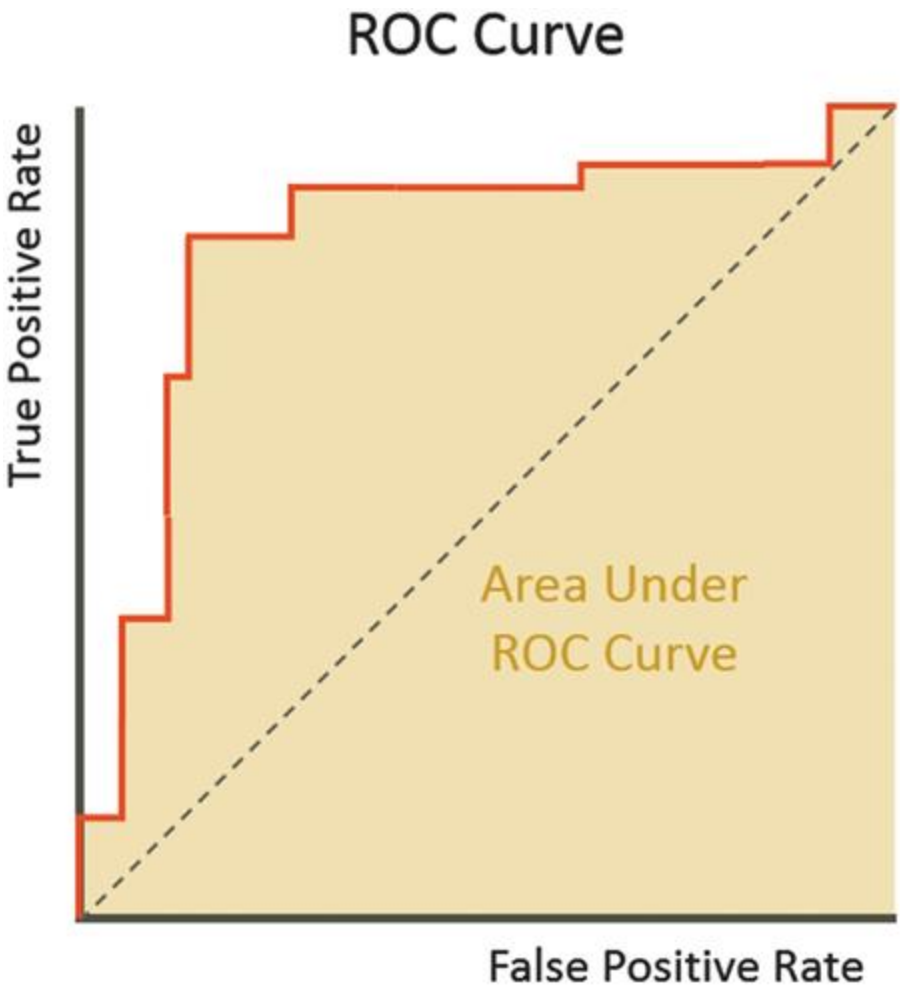


Figure 2.20 A ROC curve shows how the true positive rate (sensitivity) varies with the false positive rate (specificity). The AUC is to the right of the dotted line; a higher AUC is associated with higher model performance.

This is far from an exhaustive list, and when evaluating models it is best to consider multiple measures of quality to ensure that you are not biased by favouritism. By using good validation practice, you can ensure that you produce good models, and by validating them on additional data when that becomes available you can measure how well they perform in the real world. After all, this is the ultimate objective of any ML model.

One final note is based on ML algorithms and their place within the world of AI. Earlier I considered defining AI as "*A machine capable of doing things that humans consider to be intelligent*". This definition suggests that within AI, there is far more than just ML algorithms and

neural networks. A great many tasks that require intelligence do not require neural networks to solve them. Consider the computer game minesweeper: with relatively little effort a decent computer player could be programmed, but it would take many hours to train a neural network to learn that game, and the result may not be better. Ultimately, care must be taken when computational approaches are chosen to assist us with scientific problems.

Note on References

The References section was written with the aim of combining many of the most important lessons I learnt when first building ML algorithms, and specifically neural networks. Some of the online resources I found most useful are listed below.[1–3] These may be useful to readers wishing to gain an initial insight into ML and build their own ML algorithms.

References

1. Google's Machine Learning Crash Course: https://developers.google.com/machine-learning/crash-course, accessed Sept 2019.
2. 3Blue1Brown's series on neural networks: https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi, accessed Sept 2019.
3. CGP Grey's videos on how machines learn: https://www.youtube.com/watch?v=R9OHn5ZF4Uo and https://www.youtube.com/watch?v=wvWpdrfoEv0, both accessed Sept 2019.

CHAPTER 3

MedChemInformatics: An Introduction to Machine Learning for Drug Discovery

MATTHEW G. ROBERTS AND RAE LAWRENCE*

Cancer Research UK – Manchester Institute, Drug Discovery Unit, Alderley Park, Macclesfield SK10 4TG, UK
*Email: Rae.Lawrence@cruk.manchester.ac.uk

3.1 Introduction

Before delving into the mysteries of Machine Learning, it is appropriate to categorically state that every discussion of computational results should be prefaced with statements like "the model *suggests…*" and "the models are lies, but they can be useful." In generating models, we generate ideas – lots and lots of ideas. We fully recognise that most of those ideas are less than stellar; but in the muck of weak ideas, there are flecks of gold that can be panned out. And the harsh reality is that in order to establish whether an idea is gold or silt, we need to nicely ask our wet chemistry colleagues to go into a laboratory and do real work. With those newly minted real-life results, our models can be iteratively improved and evolve so that they can start approaching the truth.

3.1.1 Artificial Intelligence *vs.* Machine Learning

In recent years, a prevailing theme across multiple areas of research (from marketing, to economics, to drug discovery) has been Artificial Intelligence and Machine Learning. There is not a clear delineation of what these terms mean, and how they are related to, and differ from each other.

Artificial Intelligence

In the most basic of terms, Artificial Intelligence (AI) refers to a machine/computer that has been programmed to do things that humans can presently do but better; further, AI algorithms can also be taught to do things that humans can not do using only their brains – for example, separating signals from noise. Systems incorporating AI use massive amounts of data with the goal being that knowledge of data and previous decisions will allow an algorithm to successfully solve new problems using complex problem-solving skills, as a human would. A good example of this approach is using AI algorithms to have computers beat chess masters – or more recently, the application of AI technology by Google's DeepMind to solve problems like protein folding,[1] to analyse diagnostic imaging results[2] or to play Alpha Go.[3]

For an example relevant to medicinal chemistry, let's consider having a system incorporate and learn every reported chemical reaction, and ask the algorithm to devise efficient synthetic routes (with high yields) for new chemical entities. Can this be done today? It turns out that it can be done to some degree.[4]

Can AI be used to condense all of the available protein, ligand, activity, selectivity, physicochemical properties, literature information, *etc.*, and suggest a clinical candidate? This is the million pound question. There are a number of dedicated companies working in this space, including BenevolentAI, GTN Ltd., and Exscientia, all of which have had initial technical successes and have secured investor capital[5] and pharmaceutical industry collaboration[6] to continue their efforts to perfect their AI approaches to accelerate development cycles, reduce spending, and produce effective drugs.

As of this writing, AI approaches have not been made routine, reduced to practice, or deployed as box-ware. However, we believe that it is only a matter of time before currently highly customised solutions are made generic and become widely available. Given the amount of time and money

being invested, it is anticipated that success will come sooner rather than later.

Machine Learning

By contrast, Machine Learning (ML) is a sub-discipline of AI, but the approach focuses on modelling data and being able to extract knowledge. AI works in much the same way; however, many people now interpret AI as a synonym for Deep Learning methods, where the data are modelled in increasing layers of complexity.

In ML, the extracted knowledge is tested (in real life), results are fed back into the machine, and the model(s) are refined. For example, data can be mined to uncover the knowledge that active compounds have a particular combination of structural features. When a new compound that has a similar combination of features is encountered, it may be flagged as potentially active, followed by submission to the design-make-test-refine modelling cycle (see Figure 3.1), which incorporates real-life data into the model during iterative refinement.

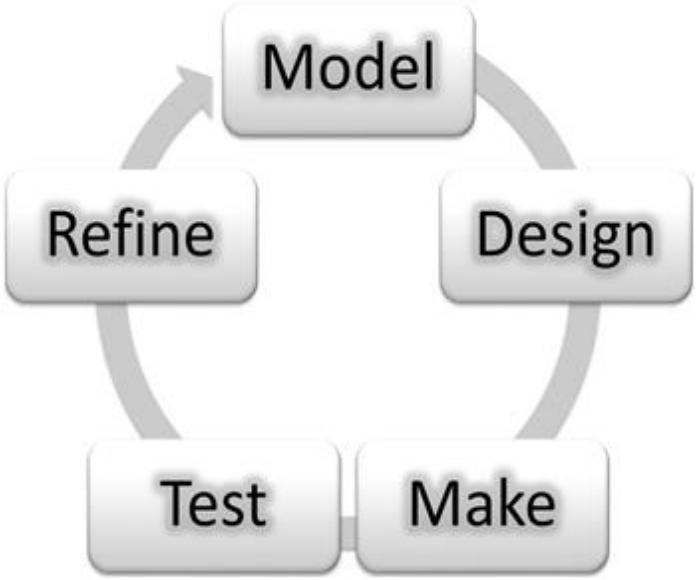


Figure 3.1 The Design-Make-Test-Refine modelling cycle illustrates how modelling *via* Machine Learning relates to medicinal chemistry discovery.

This iterative, cyclic process is how the learning in ML takes place. Machines are slower than humans to learn – they need more examples, but

they are faster at predicting results once they have learned.

Computational chemists have been using ML to successfully model data for a very long time: Hansch called it QSAR (Quantitative Structure–Activity Relationship) in his seminal paper in 1962,[7] where plant growth regulators were linked to Hammett parameters and hydrophobicity using multivariate regression. Hansch described the serendipitous birth of QSAR in his republished[7] piece in the issue of the Journal of Chemical Information and Modelling that celebrated his life and contributions to science. Shortly after came the publication of the Free-Wilson analysis,[8] which is still used today (with improvements, of course).

You may have just realised that you are already familiar with some ML approaches in medicinal chemistry!

3.1.2 Supervised *vs.* Unsupervised Learning

Machine Learning methods that employ "supervised learning" work with data that includes an observable (*e.g.*, potency or activity) and relate a set of features (descriptors, fingerprints) to the observable. Supervised learning methods use collections of features to generate predictions that can then be measured and fed back into the machine to further train the model.

Unsupervised learning methods do not make predictions, but these rather explore the relationships between features of a set of data. For example, we can cluster a collection of molecules based on their 3D fingerprint; however, this gives no prediction about their potency against any target. What this approach tells us is, simply, how similar molecules are to members of the same cluster or how different they are from other clusters.

3.2 Forewarned Is Forearmed

When embarking on data mining and the adventure of building ML models, there are a few common pitfalls and limitations to consider, which will be discussed briefly in this section.

3.2.1 Dataset Assembly and Curation

With the accessibility of big data, assembling large collections of information has become easier than ever. PubChem[9] alone has close to 96 *million* compounds spanning over 1 million BioAssays and boasting over 250 million bioactivities against over 70 000 targets. ChEMBL[10] features close to 2 million small molecule entries. Despite efforts by both groups to curate and ensure that the archived data are correct, errors are inevitable.[11]

Assuming the odd error in structure or activity slips through, it may be absorbable in the noise of the dataset. What is likely of greater importance in assembling a dataset of structures with potency information is ensuring that the data comes from consistent experiments (assays) so that the comparison is of value.

The key thing to remember is that with any model, whether AI, ML, or anything else, if garbage goes in, garbage comes out (GIGO: Garbage In, Garbage Out).

3.2.2 Model Building

The first step to linking a chemical structure with an activity is coming up with an optimal set of "features" to describe the structure – descriptors or fingerprints. We have the capacity to generate thousands of descriptors, ranging from simple topological atom counts and combinations, bond counts, physicochemical properties (measured or calculated), molecular volumes, to more complex descriptors based on molecular graph theory and 3D conformation. While these "numerical features" will not be covered in detail here, there is an excellent review of descriptors and fingerprints in a previous study.[12]

If we were to attempt to generate a regression model for a dataset using *all* of the available descriptors, there is a high likelihood that model predictions for the training set would be great and abysmally bad for the test set. In this case, the over-fitted model would have memorised the training set information and be unable to generalise it to new problems; the model captures random error instead of the true relationships.

To avoid over-fitting, we should aim to use as few descriptors as possible to capture the relationship between the structural and property features of the molecules and their activity and avoid modelling noise in the training set data. These descriptors should also be useful across a broad chemical space,

and should show minimal degeneracy, meaning that descriptor calculations should generate different values for structurally different molecules, even if the differences are minute. Additionally, it is good practice to ensure that descriptors used in a model are as orthogonal as possible, in that, individually, they define characteristics that other descriptors do not cover.

To select an appropriate set of descriptors, a dimensionality reduction technique like Principal Component Analysis[13] can be used.

3.2.3 Principal Component Analysis

The most common dimensionality reduction method encountered in MedChemInformatics is Principal Component Analysis. PCA is about reducing the number of "descriptors" that describe the data. "Descriptors" was put in quotes because, when it comes to PCA, chemically meaningful descriptors no longer exist, as PCA combines correlated descriptors into components (hence the name) and each component is orthogonal (uncorrelated) to all others. The components are then used to explain the variance in the dataset and are ordered such that the first principal component explains the most variance, the second principal component the second most variance, and so on. Figures that show the amount of variance (as a percentage) that each component explains are known as scree plots; a typical example is shown in Figure 3.2.

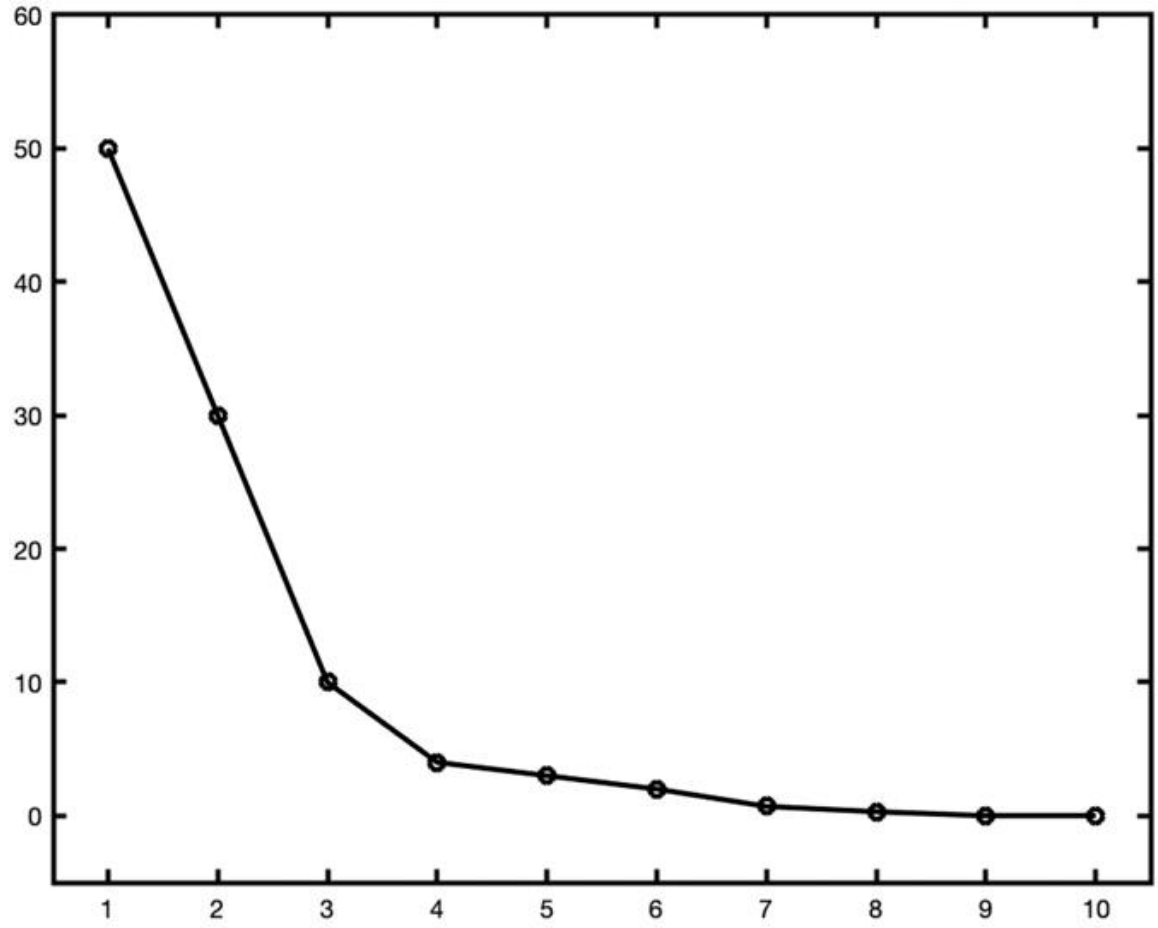


Figure 3.2 In this scree plot for fictitious data, the first principal component explains 50% of the variance, the second 30%, and the third 10%.

So how many components would you use for your model based on the scree plot in Figure 3.2? The curve plateaus from the 4th component onwards. This plateauing is referred to as the PCA "elbow" and can be the cut-off for the number of components that would be used. A general rule of thumb is to pick the number of components that explains around 80% of the variance.

3.2.4 When Machines Learn in 3D, Alignment IS EVERYTHING

2D descriptors have been used to successfully model activity. However, chemistry and life happen in 3D; how features map to a bioactive

conformation of a molecule is usually more interesting in 3D.

In translating to a 3D model, we need to understand how molecules relate to each other in their 3D conformations. There are multiple ways of aligning molecules in three dimensions, ranging from "shape only" (*e.g.*, ROCS[14]), to "shape+pharmacophore" (*e.g.*, coloured ROCS[14]), to "shape+electrostatics" (*e.g.*, field-based alignment[15]), and all of which can be biased with substructure matching to various degrees of stringency. There is equal merit in all of these approaches, but the key task at the end of any alignment *prior* to generating any model is to inspect the alignment and adjust as necessary to ensure that it is sensible and consistent.

The temptation will always be to generate the model, then look at the outliers and tweak alignments as necessary to generate models with improved statistics.

DO NOT DO THIS!

This introduces bias into the model, and while the revised model may give a brilliant improvement in the model statistics, the model is reliably nonsense. The best practice is to spend some time looking at your dataset and mining out compounds that do not belong (*e.g.*, wrong chemical series, big differences in molecular weight or other topologies, or the ones where you simply do not trust the data or the alignment), and THEN calculate the model.

3.2.5 Activity Range and Distribution of Activities for QSAR Modelling

As a rule of thumb, it is preferable to have an activity range of at least 2–3 log units, with an even distribution of data points through the activity range. Too many or too few data points in one slice of the activity range can be detrimental to the model.

All of these warnings can be summed up with GIGO.

Sadly, and in actuality, more often than we care to admit, you can have a super clean set of data, the perfect activity range, descriptors, and alignment, and you just can not squeeze out a predictive model that relates activity to structure features. Fortunately, most software companies working in this space have addressed this by providing tools for exploring the data qualitatively.

3.2.6 Outliers

Upon model inspection *via* plotting predicted *vs.* actual activities, there are occasionally data points that are quite far away from the regression line or far away from the rest of the data. How we deal with them depends on the nature of the outlier.

Sometimes, in assembling our data – especially from publicly available sources – we must consider whether the reported activity (or even the structure!) is accurate, or whether the compound may have an alternate mode of action. There are often discrepancies in reported activities when different assays are used. If we do not have confidence that the data are accurate, the mode of action is consistent with other members of the dataset, or whether the normalisation across different assay types is adequate, it is perfectly acceptable to remove the data point, with the considerations described below.

If the outlier does not affect the model or any assumptions derived from the model (see Figure 3.3), it can be removed with a footnote explaining what was done. For example, you may have a dataset that predominantly spans an activity range, but with a couple of points whose activities are quite far from the rest of the data, though model linearity is retained. If the regression equation does not change substantially when the point is removed, it is acceptable to do so, with a footnote.

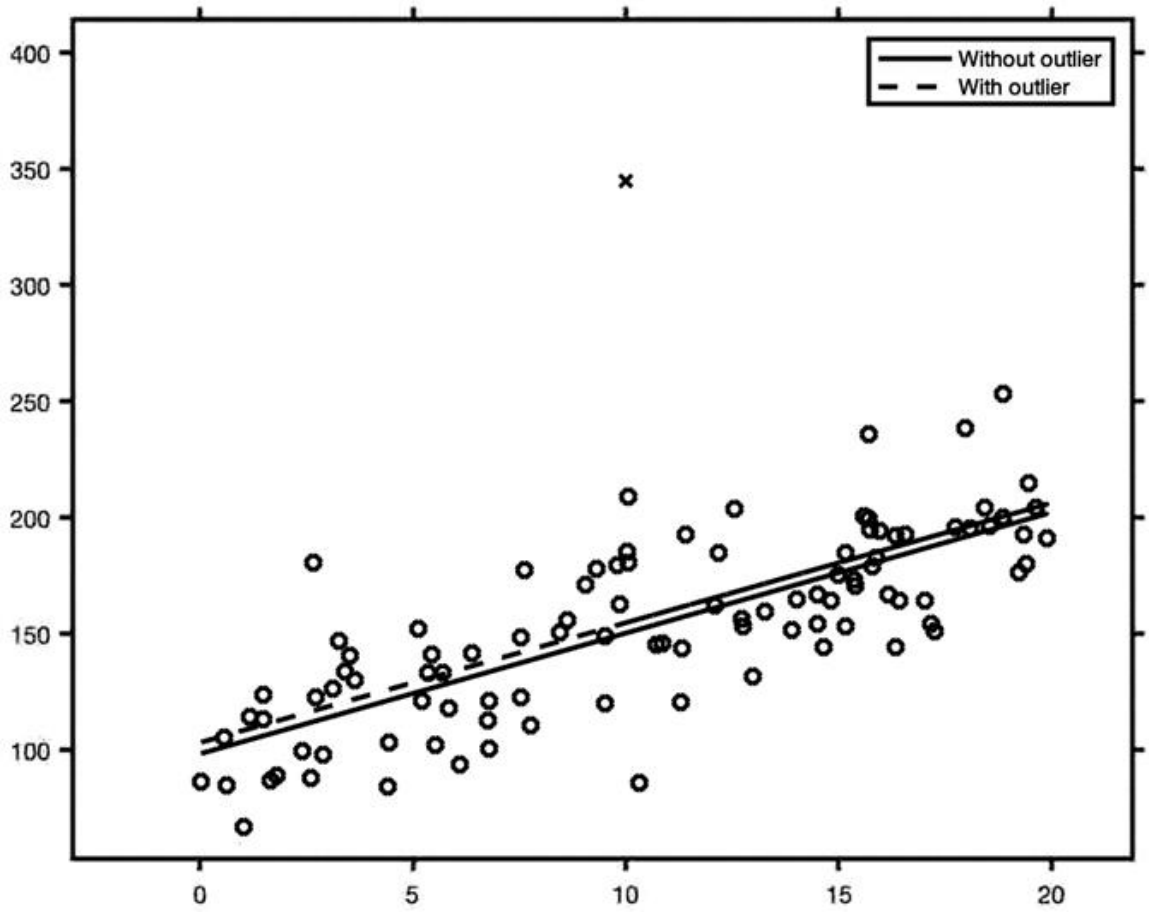


Figure 3.3 A collection of data points (o's) and a single outlier (the *x*). The regression line without the outlier is the solid line and that with the outlier is a dashed line. The inclusion (or exclusion) of the outlier does not change the regression line substantially, and thus, it is acceptable that the outlier can be excluded with an explanatory footnote.

More often than not, removing outliers results in the regression changing (see Figure 3.4), along with the assumptions. You can run the analysis with and without the data point – if you choose to drop the outlier, you should make a note about which points were dropped, the reasons why, and how the model adjusted.

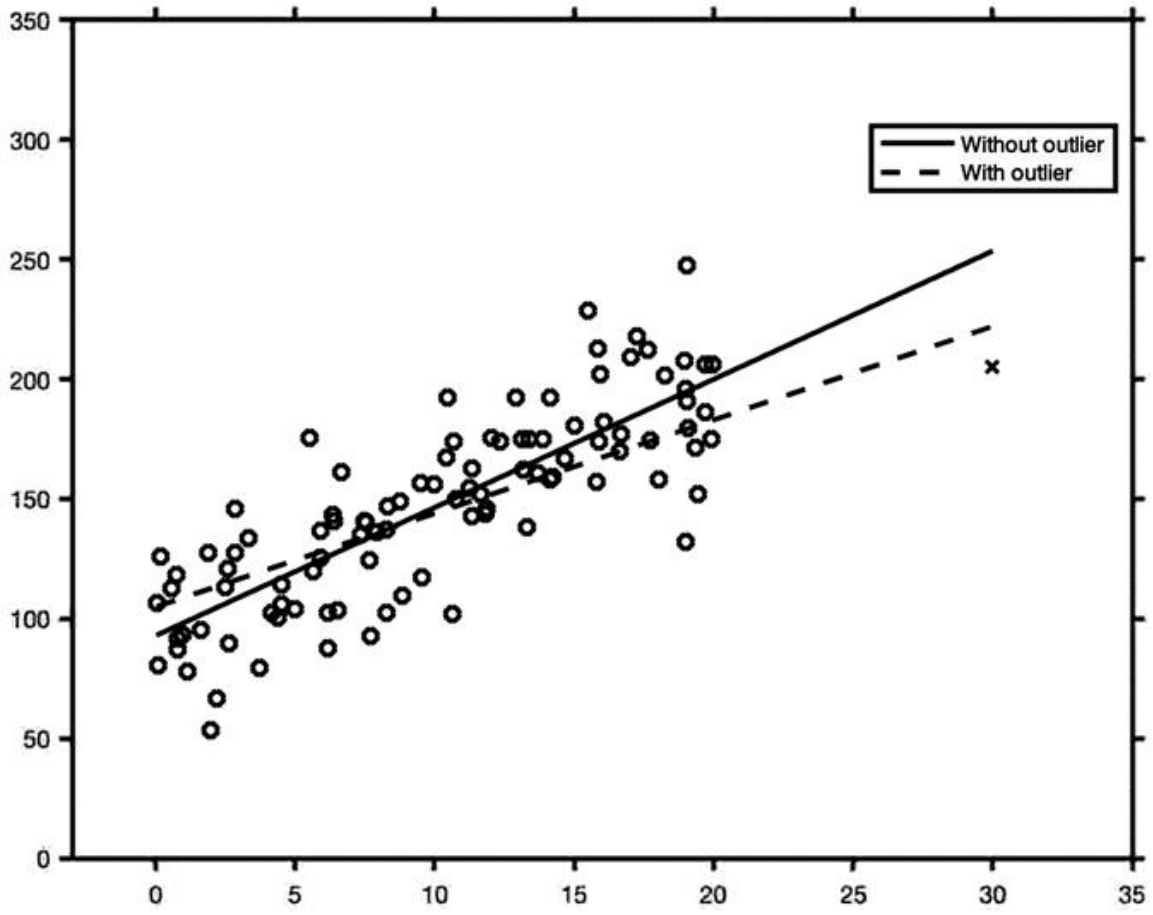


Figure 3.4 A collection of data points (o's) and a single outlier (the x) in the far upper right hand corner. The regression line without the outlier is a solid line, and that with the outlier is a dashed line. In this case, the inclusion (or exclusion) of the outlier changes the slope and intercept of the regression line, and thus, the outlier may be excluded with a footnote detailing how the model changed and why the outlier was excluded.

Other times, the data set can suggest that the outlier defines the relationship and without that outlier point, the data are completely random! In this case, the outlier should be dropped, and the model should not be reported with any significance or confidence.

If you are not comfortable dropping an outlier, sometimes a transformation like square root or log applied to your variables might yield a more reasonable model. Alternatively, you might consider that the relationship you are trying to model is nonlinear – and that is where Machine Learning can really help. *But in any case, it is critical that you*

understand your data, its origin, and the plausible relationships that can exist between the observable and the descriptors.

3.3 Deep Learning and Neural Networks

In recent years, mentions of “Deep Learning” and “Neural Nets” in the drug discovery literature have resurged after some years of dormancy. Deep Learning is a sub-discipline of ML which incorporates Artificial Neural Networks. Simply put, the data are modelled in layers, with each layer increasing in complexity. Traditional applications of this approach were in speech recognition and natural language processing, but it has since been applied to many other areas.

These approaches are beyond the scope of this chapter, but we do recommend reading about Merck's Molecular Activity Challenge competition[16] in which invited participants were asked to identify the best approaches to predicting biological activity (on- and off-target) using numerical descriptors for structures. A computer science group from the University of Toronto won the challenge by invoking multi-task Deep Neural Networks.[17]

3.4 Underlying Mathematics

3.4.1 “Observables” and “Features”

Before exploring the common ML methods used in medicinal chemistry informatics (which we are calling “MedChemInformatics”), we must first consider what features (variables) are to be correlated with the observable.

For example, if we consider a simple relationship between boiling point and the number of carbons in an alkyl chain, we know from our organic chemistry wisdom that smaller alkanes have lower boiling points, but the relationship becomes more complicated when functional groups are introduced, as additional molecular features (*e.g.*, polarity) are involved in making molecules stick together instead of escaping to the vapour phase.

An old article[18] details how boiling point is related to molecular features (derived empirically) *via* a linear relationship:

$$\text{Boiling Point} = py + q \tag{3.1}$$

Here, y is a constant for each alkyl group and is related to molecular surface area; p and q are constants for functional groups that capture non-electrostatic and electrostatic forces. This linear relationship, when applied to a defined compound class (*e.g.*, acetates) reportedly predicts more than 80% of the boiling points to within 2 °C. In this case, the equation relates an observable (boiling point) with a set of variables that can be considered descriptors for each compound class.

3.4.1.1 Molecular Descriptors and Fingerprints

In MedChemInformatics, we are often interested in ways that we can classify molecules as being highly similar or highly different in terms of their structural features and properties. There are many reasons we do this, including, but not limited to, virtual screening, identifying backup series, and establishing intellectual property position.

To assess similarity, we need to decide what underlying features are important. For example, we can group molecules together based on common scaffolds (*e.g.*, substructure matching, Murko scaffolds), or alternatively, we can use collections of properties (*e.g.*, DrugLike, BBB penetratability).

Descriptors have been covered at length in numerous reviews,[19] so these will only be briefly discussed here. Descriptors come in two flavours: those that are based on experimentally measured physicochemical properties (*e.g.*, logP, dipole moment) and those that are calculated in some fashion from the molecular structure ("theoretical descriptors").

Theoretical descriptors that are regularly used in modelling can be further categorised:

- 0-Dimensional: these are "counts" that capture the constitution of a molecule (*e.g.*, # carbons).
- 1-Dimensional: these capture fragments and substructural features (*e.g.*, fingerprints).

- 2-Dimensional: these are chemical graph invariant descriptors derived from the 2D structure and are topological indices that capture the connectivity of atoms (*e.g.*, Wiener index).
- 3-Dimensional: these relate to molecular size, sterics, surface, and volume descriptors. For example, VSA (van der Waals surface area) mapped with partial charges of the atoms contributing to the volume. There are 3D fingerprints that capture pharmacophoric features.
- 4-Dimensional: these are derived from 3DQSAR methods such as Comparative Molecular Field Analysis[20] (CoMFA). Field or volume sampling points are established for a group of molecules that have been aligned using their preferred (binding mode) 3D conformations, and a field or steric value is calculated at the sampling point in space.

To conceptualise what a Fingerprint is, let's simplistically consider it as a series of checkboxes that are either ON (1) or OFF (0) and use it to compare a horse and an alligator using the chart shown in Figure 3.5.

Feature	Horse	Alligator
4 legs	1	1
Warm-blooded	1	0
Wings	0	0
Hair	1	0
Teeth	1	1
Carnivore	0	1
Domesticated	1	0

Figure 3.5 On/off checkboxes for each fingerprint feature convert to a bit string that captures the predefined characteristics.

The horse's fingerprint is (1,1,0,1,1,0,1); the alligator's fingerprint is (1,0,0,0,1,1,0). The fingerprint string is highly dependent on what features are used to describe the molecule. Similarity metrics are used to compare the bit strings ((1101101) and (1000110)) to score how similar they are, and the calculated similarity value depends on the fingerprint chosen for the comparison.

Fingerprints for molecules are long bit strings that query the molecule for features (atom types, substructure, *etc.*), in an effort to capture the molecular graph (2D connectivity of atoms), and in the case of 3D fingerprints, the arrangement of functional groups with respect to each other. Descriptors and Fingerprints have been described in the literature for many years and are widely implemented in commercial and open-source software packages, and these will not be discussed in detail here, but we suggest suitable papers to read for more information.[21*a*–*c*]

Once descriptors are calculated for a set of molecules, we try to relate the combination of descriptors to the observable (*e.g.*, potency) to create groupings or clusters of molecules that have similar features and similar potencies, or to find a mathematical relationship between the features and the observable. The approaches to elucidating the mathematical relationship are discussed later.

3.4.2 Comparing Molecules Based on Their Descriptors

To determine whether two molecules are related to each other in some way based on their descriptors, we must be able to determine their similarity to each other (or conversely, distance from each other) in the descriptor space. The most common methods encountered in medicinal chemistry are Tanimoto Similarity, Tversky Index, and Dice Similarity. These values are usually normalised to be a number between 0 and 1, such that 1 is perfect similarity (100% match).

3.4.2.1 Tanimoto Similarity

Tanimoto Similarity[22] first appeared in the literature in the late 1950's in efforts to classify plants and has since become widely adopted across other disciplines. This is the similarity metric you will see most often.

In descriptor space, the Tanimoto Similarity (Index) is given by:

$$S_{A,B} = \frac{\left[\sum_{j=1}^{n} x_{jA} x_{jB}\right]}{\left[\sum_{j=1}^{n} (x_{jA})^2 + \sum_{j=1}^{n} (x_{jB})^2 - \sum_{j=1}^{n} x_{jA} x_{jB}\right]} \tag{3.2}$$

where $S_{A,B}$=Tanimoto Similarity coefficient; n=number of features; j=index/count variable; x_{jA}=jth feature of molecule A; x_{jB}=jth feature of molecule B.

In fingerprint space, this expression simplifies to:

$$S_{A,B} = \frac{c}{a + b - c} \tag{3.3}$$

where a=number of "on" bits in molecule A; b=number of "on" bits in molecule B; c=number of "on" bits in *both* A & B.

If we return to the comparison of horses (A) and alligators (B):

Horse	1	1	0	1	1	0	1
Alligator	1	0	0	0	1	1	0

$a = 5; \quad b = 3; \quad c = 2$

$$S_{A,B} = \frac{2}{5 + 3 - 2} = 0.3 \tag{3.4}$$

The Tanimoto Index (or score) is one of the most commonly encountered similarity metrics in MedChemInformatics. A general rule of thumb is that we use 0.85 as a cut-off for similar molecules;[5d] however, multiple studies have demonstrated that the cut-off is not reliable when different molecular representations are used.[21c,23]

Tanimoto, while commonly used, has limitations and weaknesses – for example, a Tanimoto score of 0.3 is frequently returned for structurally distant molecules.[24] If we return to our horse-alligator comparison, with a Tanimoto score of 0.3, some awareness that this could be an artifact of the method is in order.

3.4.2.2 Tversky Similarity

Another commonly encountered similarity metric used in MedChemInformatics is the Tversky coefficient,[25] which is given by:

$$Tv_{\alpha,\beta}(A,B) = \frac{c}{\alpha(a-c)+\beta(b-c)+c} \tag{3.5}$$

$Tv_{\alpha,\beta}(A, B)$=Tversky coefficient;
α,β=non-negative weighting factors;
a=number of "on" bits in molecule A;
b=number of "on" bits in molecule B;
c=number of "on" bits in *both A* & *B.*

The larger the value of α compared to β, the more weight is placed on the unique features of molecule A and *vice versa*. This allows the introduction of asymmetry into the calculation of a similarity index or similarity score and exploration of the importance of common and unique features in the comparison to the reference compound *A*.

When α and β are set to one, the Tversky coefficient reduces to the Tanimoto similarity. When both coefficients are equal to 0.5, the Tversky coefficient reduces to the Dice Similarity.

If we again return to our horse-alligator comparison, and decide that the horse's features are twice as important as the alligator's (α=2; β=1):

Horse	1	1	0	1	1	0	1
Alligator	1	0	0	0	1	1	0

$$a=5; \quad b=3; \quad c=2 \tag{3.6}$$

$$Tv_{\alpha=2,\beta=1}(A,B) = \frac{2}{2(5-2)+(3-2)+2} = 0.2 \tag{3.7}$$

Conversely, if the alligator features are twice as important, the Tversky coefficient becomes 0.28 – thankfully, still not similar.

3.4.2.3 Dice Similarity

A third approach to calculating similarity that is commonly encountered in chemistry is Dice Similarity.[26] Dice Similarity first appeared in 1940's ecology research but now is commonly used to compare molecules.

$$Dc_{A,B} = \frac{\left[2\sum_{j=1}^{n} x_{jA}x_{jB}\right]}{\left[\sum_{j=1}^{n}\left(x_{jA}\right)^2 + \sum_{j=1}^{n}\left(x_{jB}\right)^2\right]} \tag{3.8}$$

$$Dc_{A,B} = \frac{2c}{a+b} \tag{3.9}$$

$Dc_{A,B}$=Dice Similarity coefficient;
n=number of features;
j=index/count variable;
x_{jA}=jth feature of molecule A;
x_{jB}=jth feature of molecule B.

Returning one last time to our horse-alligator comparison:

Horse	1	1	0	1	1	0	1
Alligator	1	0	0	0	1	1	0

$$a = 5; \quad b = 3; \quad c = 2 \tag{3.10}$$

$$Dc_{A,B} = 2(2)/(5+3) = 0.5 \tag{3.11}$$

At this point, you should feel some intellectual indigestion about similarity coefficients. Using the same simplistic fingerprint, we arrived at three different similarity scores. This is often the case for the more complicated molecular fingerprints.

3.4.2.4 Which Similarity Metric Works Best?

There are a few observations to make – for example, Dice and Tversky Similarities values tend to be slightly higher and are less symmetric than

Tanimoto; further, Tversky and Dice give similar results, regardless of the reference feature weighting used in Tversky. But ultimately, a calculated similarity score has no absolute meaning – it is the ranking of compounds that is of more interest,[22d] so one is as good as the next for general purposes.

3.4.3 Model Quality and Statistics

Let's return to predictive models. When generating models, we use whatever data that are available, and rarely is anyone upset about having too much data!

If we use all of the data available to build a model, we are limiting the model to a training set that can only be cross-validated against itself. The predicted value is then compared to the measured value. This is acceptable if only a small amount of data is available, but it is not a reliable approach. This could be an early model that could be used to feed the design-make-test-refine machine (Figure 3.1), to achieve improved (and statistically relevant) models as the amount of data available increases.

It is preferable to partition the dataset into a training set and a test set, either randomly or by selecting a subset that is evenly dispersed across the observable range. The general rule of thumb is that 20% of the data goes to the test set. The training set is used to build the model and cross-validate against itself, while the test set is used to apply the model to real compounds and assess whether the predicted and actual values match up for data points that are not included in the model training.

When a model is calculated, several statistical parameters are reported that relate to the quality of the model in terms of its statistical relevance and how predictive the model will be on new data.

It is worth noting that 95%/5% split is commonly used in Deep Learning because the models are built using vast quantities of data. A 5% test set still contains a LOT of data points in absolute terms!

3.4.3.1 Coefficients of Determination

For linear models (assuming continuous data), the first value that comes to mind is the coefficient of determination (r^2 or R^2), which captures how well the data are fitted to the regression line. R^2 is always between 0 and 1; the higher the value of R^2, the better the model fits the data.

R^2 can be expressed as:

$$R^2 = 1 - \frac{\sum_i \left(y_{\mathrm{expt}(i)} - y_{\mathrm{calc}(i)}\right)^2}{\sum_i \left(y_{\mathrm{expt}(i)} - \langle y_{\mathrm{expt}} \rangle\right)^2} \tag{3.12}$$

i=number of samples;
$y_{\mathrm{expt}(i)}$=experimental/measured value of the ith sample;
$y_{\mathrm{calc}(i)}$=calculated value for the ith sample;
$\langle y_{\mathrm{expt}} \rangle$ =average experimental/measured value.

As the difference between the experimental value and the calculated value approaches zero, R^2 approaches 1, which suggests a perfect correlation between calculated and observed values in the training set (100% of the variance is captured by the model).

The second value to consider is the cross-validated coefficient of determination, Q^2 or q^2. This is similar to R^2; however, it incorporates what the model would predict if it was built with one (Leave-One-Out – LOO) or many (Leave-Many-Out – LMO) of the data points omitted. The simplest cross-validation approach, Leave-One-Out, excludes each sample once and constructs a model with the training set less one sample, then calculating the variable. The residuals (differences between experiment and calculated values) are then used to calculate the correlation coefficient of determination according to:

$$Q^2 = 1 - \frac{\sum_i \left(y_{\mathrm{expt}(i)} - y_{v(i)}\right)^2}{\sum_i \left(y_{\mathrm{expt}(i)} - \langle y_{\mathrm{expt}} \rangle\right)^2} \tag{3.13}$$

i=number of samples;
$y_{\mathrm{expt}(i)}$=experimental/measured value of the ith sample;
$y_{v(i)}$=calculated value for the ith sample from the internal validation (LOO model);
$\langle y_{\mathrm{expt}} \rangle$ =average experimental/measured value.

Cross-validation is intended to give an assessment of the robustness and predictiveness of the model. The higher the Q^2, the better the model should perform. Anecdotally, a Q^2>0.5 and R^2>0.6 is considered proof that a model is predictive, but that is not always the case. Additionally, a difference of 0.2–0.3 between R^2 and Q^2 suggests that the overfitting might be in play.

R^2 and Q^2, while a good place to start, are not the only considerations when assessing the quality of a model. The best approach is to use an external test set to assess whether a model is predictive or not.[27]

3.4.3.2 Kendall's Tau

Another statistical parameter that is worth considering is Kendall's Tau,[28] which is a measure of rank correlation. When calculated, it is a number bound by 0 and 1 and gives an estimation of how well a model can get the rank order correct, regardless of its success in calculating values with absolute accuracy.

3.4.4 Overtraining and Characteristics of Good Descriptors

Earlier, the term overfitting ("overtraining") was used when there was a large discrepancy between R^2 and Q^2. This often happens when too many descriptors are used to build the model from the training set, and those descriptors are not universally related to the observable for other data. In other words, with enough descriptors, you have captured all of the variance in the noise – the model has basically memorised the training data.

Models that are over-trained violate Occam's Razor: to avoid over-training, find the simplest relationship with the fewest variables. You can do this using a dimension reduction technique like Principal Component Analysis.[13]

3.5 Machine Learning Methods

In this section, we will discuss the more commonly encountered ML methods with drug discovery modelling as a context. Features will be called "descriptors" and the observable will be called "activity" (in most cases).

Thus, we have a collection of molecules (training/test/validation sets), calculate a set of descriptors that capture their features, then relate the features to their activity in some mathematical fashion.

3.5.1 *k*-Nearest Neighbours

The simplest Machine Learning algorithm used in medicinal chemistry is *k*-Nearest Neighbours (*k*NN); this approach can be used for both classification and regression analyses. The essence of the *k*NN approach is captured in the proverb: "When the character of a person is not clear to you, look to their friends."

*k*NN is a "lazy learning" method: in making predictions or classifications, the model is applied locally as opposed to globally. The *k*NN approach is non-parametric in the sense that it does not make any assumptions about the functional form of the relationship between the descriptor(s) and the activity – it could be linear, parabolic, quadratic, whatever. In the course of preparing a *k*NN model, the training set is "memorised", and a test query is compared to every single data point in the training set in order to classify or make a prediction. This means that the model "trains" very quickly (no fancy math happening in the background). However, large amounts of data need to be kept in active memory and it takes time to compare a query to every single data point in the training set in order to make a prediction or classification for the query.

*k*NN can be used for both classification and regression models using 2D or 3D chemical descriptors, and it is less sensitive to noise in 3D structural alignments than partial least squares (PLS). Classification models are used when the data are not continuous, for example, ranges of % inhibition and classes such as "high", "medium", "low." In a sense, *k*NN is about pattern recognition: molecules with similar features should exhibit similar biological activities.

A simplified *k*NN workflow is shown in Figure 3.6.

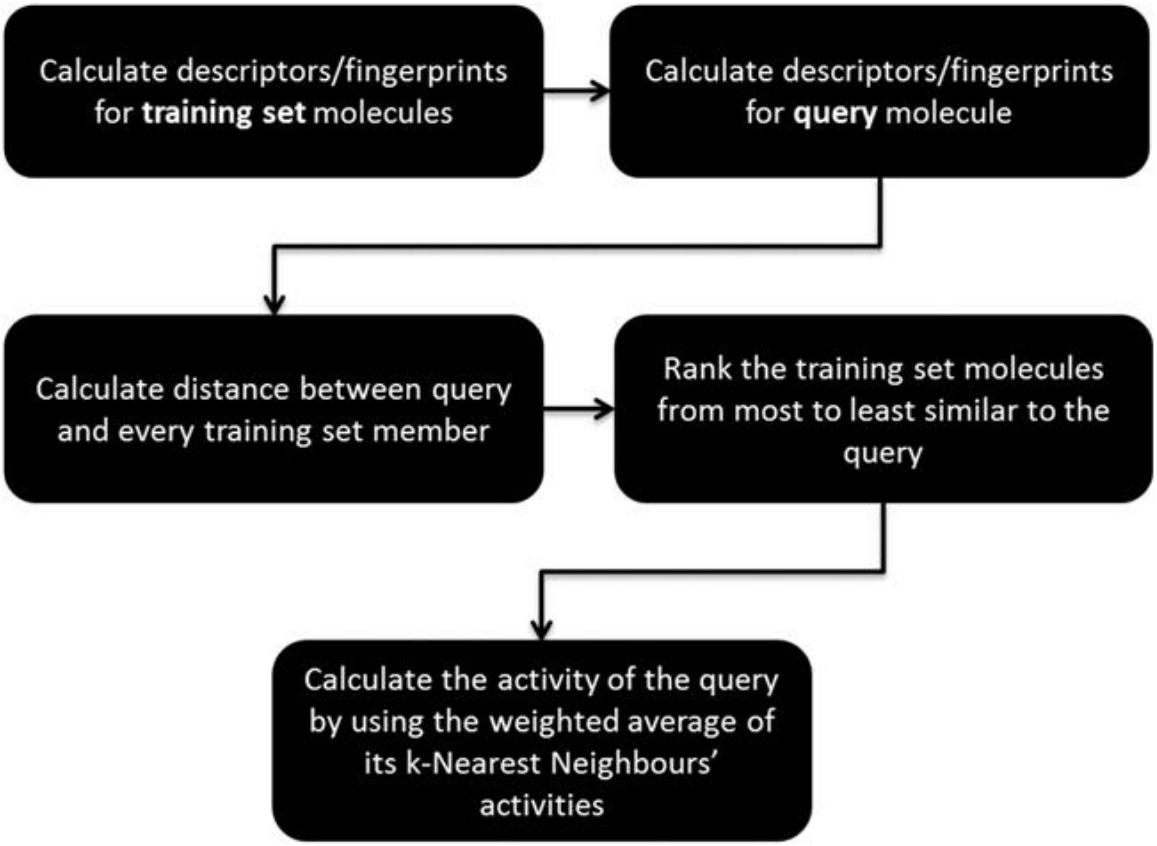


Figure 3.6 The kNN workflow for predicting the activity of a query based on the training set. In calculating the distance, any similarity metric will suffice (*e.g.*, Euclidean distance, Tanimoto, Tverskey, Dice, *etc.*). The weighing factors are inverse distance or similarity – *i.e.*, training set molecules that are more similar to the query will contribute more to the activity prediction.

The most important parameter to adjust when using *k*NN modelling is the number of nearest neighbours (*k*). A high value of *k* results in a model that has more training set contribution to the prediction and is more resilient to outliers. This means that there is lower variance, but increased bias. On the other hand, a low or more restrictive *k* gives the most flexible fit in those cases where the functional form of the relationship between the descriptors and the activity may vary. Lower *k* values give high variance but low bias.

So, how do we pick the optimal value of *k*? The best *k* is the one that corresponds to the lowest error rate. Generally, the optimal value of *k* is found heuristically based on RMSE (and Q^2) during cross-validation. Unfortunately, using a test set to tune *k* can result in over-fitting, so we need to consider using an independent validation set.

Final Thoughts

To summarise, *k*NN is the simplest of the ML methods that MedChemInformaticians are likely to encounter. Although it is a sound way to classify and predict (especially when the relationship is local as opposed to global), its application suffers from data storage and computational expense. *k*NN models have been employed successfully in drug discovery. A few examples include understanding QSAR and assisting with designing new potent inhibitors,[29] organ toxicity prediction,[30] hERG activity,[31] and protein function prediction.[32] For a more detailed description of kNN, see ref. 33.

3.5.2 Linear Regression

Probably, the most common of the ML methods encountered is based on different flavours of linear regression. Linear regression is the mathematical attempt to fit a straight line to a collection of data points to relate a descriptor (or set of descriptors) with an activity. We see this method regularly reported in articles detailing 2D and 3D QSAR models.

Regression modelling works on the assumption that there exists some relationship between a set of descriptors and the measured activity. One approach is to assume a linear relationship: meaning that each descriptor independently contributes with a *constant proportionality* to the activity. The predicted activity is, therefore, "explained" by the weighted sum of each descriptor; the magnitude of the weight indicates the significance of a descriptor towards the magnitude of the activity:

$$P_i = \sum_{n=1}^{N} w_n F_{i,n} \tag{3.14}$$

P_i=predicted response of the *i*th observation;
n=*n*th descriptor;
N=maximum number of descriptors;
$F_{i,n}$=value of the *i*th observation for the *n*th descriptor.

The goal of linear regression analysis/algorithms is to find values for the weights, w_n, that will minimise the error between the predicted activity (P_i)

and the experimentally observed activity (R_i):min $\sum\varepsilon_i$,

$$\min \sum\varepsilon_i, \qquad \varepsilon_i = |P_i - R_i| \text{ or } \varepsilon_i = (|P_i - R_i|)^2 \tag{3.15}$$

Ordinary Least Squares (OLS) and Partial Least Squares (PLS) are methods used to find weights under different conditions. What can be used, or definitely cannot be used, will depend on the number of observations (i) and the number of uncorrelated descriptors (n). The number of descriptors used for predicting the activity can be reduced, if required, by methods referred to as feature selection and feature extraction.

Feature selection refers to methods that select and use a subset of the original descriptors *without* modification to predict an activity. The regularisation regression algorithm Lasso[34] (Least absolute shrinkage and selection operator) is an example of a feature selection method. Lasso uses the OLS method to find weights for the descriptors that minimise the error. However, Lasso not only looks to minimise the error, but also attempts to minimise the number of non-zero weights used in the regression analysis. This is achieved by adding a penalty term for each non-zero weight:[35]

$$\min\ c = \sum_{i=1}^{M}\left(R_i - \sum_{j=1}^{N} w_j F_{i,j}\right)^2 + \gamma \sum_{j=1}^{N} |w_j| \tag{3.16}$$

c=cost function;
i=ith observation;
M=number of observations;
R_i=Response value for ith observation;
j=ith descriptor;
N=number of descriptors;
w_j=weight for feature j;
$F_{i,\,j}$=Value of ith observation for jth descriptor;
γ=penalty coefficient.

Feature extraction, as done in PCA, involves creating new components from the original set of descriptors.

3.5.2.1 Ordinary or Partial Least Squares?

Consider these equations:

$$x + y = 4 \qquad \text{(a)} \tag{3.17}$$

$$x - y = 0 \qquad \text{(b)}$$

$$x + y = 2 \qquad \text{(c)}$$

If we only had equation (a), there would be infinite many solutions to this equation: (x, y)=…; (5, −1); (4, 0); (3, 1); … (0,4); (−1, 5); … If we had (a) and (b), one solution exists: (x, y)=(2, 2).

However, with (a), (b), and (c), there is no solution that satisfies all three equations, but there is a solution that gives the best average result: (x, y)= (1.5,1.5) gives errors of −1, 0, and 1 for a, b, and c, respectively.

With only (a), there were 2 descriptors (x and y) and one observation so that the number of descriptors was greater than the number of observations. With (a) and (b), the number of descriptors equalled the number of observations. With (a), (b), and (c), the number of observations was greater than the number of descriptors.

When # Observations> # Descriptors

If and only if the number of observations is greater than the number of independent descriptors, it *may* be possible to use OLS to find the weights.

OLS is relatively simple. The dataset of observations and descriptors can be expressed as a matrix, **X**. The size of a matrix is expressed as number of rows first followed by the number of columns. So a matrix with m observations and n descriptors will have size [m, n]. If **Y** represents the response and we have a response for each observation, **Y** has size [m, 1]. As we want weights for each descriptor, weight matrix, **B**, will have size [n, 1]. The ordering of matrices is very important because, generally, **A**×**B**≠**B**×**A**. For matrix multiplication, we need the inner dimensions to agree. Therefore:

$$Y = X\,B \tag{3.18}$$

Taking a look at the dimensions: [m, 1]=[m, n][n, 1], this is possible as the number of columns in **X** is equal to the number of rows in **B** – the inner

dimensions agree. It is, therefore, important to understand the significance of multiplying on the left side or right side of matrices.

The next step involves multiplying by the transpose of **X**, or $\boldsymbol{X}^{\mathbf{T}}$. The transpose of a matrix means to convert the rows to columns and the columns to rows:

$$X^{\mathrm{T}}Y = X^{\mathrm{T}}X\,b \tag{3.19}$$

Next step involves left multiplying by the inverse of $\boldsymbol{X}^{\mathbf{T}}\boldsymbol{X}$, or $(\boldsymbol{X}^{\mathbf{T}}\boldsymbol{X})^{-\mathbf{1}}$:

$$(X^{\mathrm{T}}X)^{-1}\,X^{\mathrm{T}}Y = (X^{\mathrm{T}}X)^{-1}\,X^{\mathrm{T}}X\,b \tag{3.20}$$

$(\boldsymbol{X}^{\mathbf{T}}\boldsymbol{X})^{-\mathbf{1}}\boldsymbol{X}^{\mathbf{T}}\boldsymbol{X}$ will result in the identity matrix, **I**, such that:

$$(X^{\mathrm{T}}X)^{-1}\,X^{\mathrm{T}}Y = X^{\mathrm{T}}X)^{-1}\,X^{\mathrm{T}}X\,b = I\,b = b \tag{3.21}$$

The Achilles heel of OLS is that the inverse may not always exist. This occurs when the descriptors are not independent (or not independent enough), resulting in a determinant (a quantity necessary for calculating the inverse) that is zero and as the determinant is used in division, and the resulting matrix is referred to as being singular.

When # Observations <# Descriptors

When there are more descriptors than observations, OLS fails because the inverse, $(\boldsymbol{X}^{\mathbf{T}}\boldsymbol{X})^{-1}$, definitely does NOT exist. Principal Component Regression and PLS are two methods that may overcome this hurdle.

Although Principal Component Regression uses OLS, it first undergoes feature extraction. Principal Component Analysis uses components that can represent a combination of correlated descriptors. PCA finds components that maximise the variance in the descriptor space without considering the response data. Components refer to eigenvectors that are ranked according to their corresponding eigenvalue from largest to smallest. Each component is orthogonal, meaning that there is no correlation between components. A number of components are picked – the first component explains the most variance, the second component the second most variance, and so on. Then after selecting how many components to use for predicting outcome, OLS can be used to find weights for each component (assuming that the number of observations is now greater than the number of components). The

downside to this approach is that components that explain the most variance in the descriptor data are not necessarily the best predictors for the response.

Partial Least Squares addresses this issue by maximising the *covariance* between descriptors and the response. It does this by creating a covariance matrix, $\mathbf{X}^{\mathbf{T}}\mathbf{Y}$, where **X** is a matrix consisting of the descriptor data and **Y** is the response data. PLS actually covers a class of algorithms. It was introduced in 1966,[36] and it has been improved and enhanced over the past 50 years. A commonly used implementation is NIPALS (Nonlinear Iterative Partial Least Squares):[37] $\boldsymbol{X_0}=\boldsymbol{X}$, $\boldsymbol{Y_0}=\boldsymbol{Y}$

$$\boldsymbol{S}=\boldsymbol{X_0}^{\mathbf{T}}\boldsymbol{Y_0} \tag{3.22}$$

Initial weights (w) are calculated from eigenvectors of **S**:

$$w=Eig_{\max}\{S\ S^{\mathrm{T}}\} \tag{3.23}$$

where $Eig_{\max}$ refers to the eigenvector corresponding to the largest eigenvalue.

Variables required to update matrices **X** and **Y** are calculated next:

$$t=X_0\ w \tag{3.24}$$

$$c=\frac{\boldsymbol{Y_0}^{\mathbf{T}}\boldsymbol{t}}{\boldsymbol{t}^{\mathbf{T}}\boldsymbol{t}} \tag{3.25}$$

$$p=\frac{\boldsymbol{X_0}^{\mathbf{T}}\boldsymbol{t}}{\boldsymbol{t}^{\mathbf{T}}\boldsymbol{t}} \tag{3.26}$$

Next, matrices **X** and **Y** are updated and deflated so that new matrices are the residuals:

$$X_1=X_0-t\,p^{\mathrm{T}},\ Y_1=Y_0-t\,c^{\mathrm{T}} \tag{3.27}$$

This process is carried out *r* times with *r* being the *rank* of matrix **X**. *Rank* refers to linearly independent rows. If all the values of *w*, *t* and *p* are stored in **W**, **T** and **P** respectively, the weights, **B**, can be found by:

$$B = Z^{+}Y \tag{3.28}$$

where:

$$Z = T\,(P^{T}W)\,W^{T} \tag{3.29}$$

and $\mathbf{Z}^{+}$ denotes the pseudoinverse such that:

$$Z^{+} = (Z^{T}Z^{+})^{-1}Z^{T} \tag{3.30}$$

It is important to note that PLS can suffer from scaling of descriptors.[38] The magnitude of the change in a descriptor will affect the covariance matrix and therefore will change the dominating eigenvector, thereby changing the weights in a nonlinear way; as Cramer stated, this is contrary to ordinary least squares as changing a linearly independent descriptor will change the weight by a factor that is the inverse of the change.

Comparing Principal Component Regression (PCR) and PLS, PCR looks to maximise variance in the descriptor space without consideration of the response, and the maximum varied descriptors may not necessarily account for variation in response. PLS, on the other hand, maximises covariance between descriptors and response.

Final Thoughts

Overfitting is not usually an issue with linear regression when the number of observations is greater than the number of descriptors. If there are more observations than descriptors, OLS should generally work. If there are more descriptors, PLS may be implemented.

3.5.3 Decision Trees and Random Forests

Far more complex and far more powerful is the use of Decision Tree approaches to model the relationship between descriptors and biological activity.

Decision Trees encompass algorithms that partition observations through careful ordering of descriptors for mainly classification analysis. Decision Trees take their name based on how their results are visualised. A Decision Tree starts off with a single node (or *root node*). Nodes represent descriptors. Branches, representing a partitioning criterion (decision), then

stem off from this node that is either sufficient to categorise all observations that satisfy that decision, or lead to another node and, thus, another decision. Partitions are created until all observations are correctly categorised, or until a certain depth to the tree is reached. Two early and popular decision tree algorithms are ID3 and C4.5. ID3[39] is an algorithm that handles only categorical data whereas C4.5[40] can handle categorical and continuous data – we will discuss ID3 here, but refer you to Quinlan's article for further information on C4.5. To understand how these algorithms work, it is necessary to know what is meant by *information*.

3.5.3.1 Information

Information theory states that information, or (Shannon) entropy, is a measure of uncertainty. Revealing the outcome of an event with lower probability will generally have more information than an event with higher probability. The equation for entropy is:

$$H = \sum_{i=1}^{N} P_i \log_2 P_i \tag{3.31}$$

H=Entropy, (*N.B.:* Information Theorists use “H” instead of “S” for entropy);
P_i=Probability of event i occurring.

Tossing a biased coin with 100% chance of landing on heads will have zero entropy (this can alternatively be viewed as a 0% chance of landing on tails), in which case:

$$H = \lim_{P_n \to 0} P_n \log_2 P_n = 0. \tag{3.32}$$

A coin that has a 90% chance of landing on heads has slightly more entropy (~0.14) and a fair coin has the most entropy (0.5). The value itself is not so important as $\log_2$ is just used for convention ($\log_{10}$ or the natural log can just as easily be used). What *is* important is the ranking of entropy values for different events.

Another way to view entropy is to consider how many *efficient* questions would need to be asked to reveal the outcome of events. Let's use another coin toss example, this time, 8 tosses of a *heavily biased* coin with tails favoured. Each coin toss could be queried; however, this would not be efficient given the knowledge that the coin is heavily biased. The more efficient approach is to group the coin tosses into fours, then ask: “does this group contain any heads?” As the coin is heavily biased against heads, the probability of a rare event (coin landing on heads) occurring in each group is low. Thus, we would expect to reveal 50% of the remaining toss outcomes after each question, and to reveal all coin toss outcomes after 3 questions. This illustrates why a fair coin has more entropy than a biased coin – each toss of a fair coin would need to be queried.

3.5.3.2 Algorithms

The ID3 algorithm uses entropy to decide which descriptor to use at a node. Firstly, the entropy of the response is calculated as follows:

$$H(R) = -\sum_{i=1}^{N} P(R_i)\ \log P(R_i) \tag{3.33}$$

H=Entropy;
R=Response;
N=Number of unique responses;
$P(R_i)$=Probability of Response i, or the ratio of response i to total number of observations, such that $P(R_i) = \frac{\# i}{\# \text{observations}}$.

Next, the information that is gained by partitioning the dataset using a descriptor is calculated for each descriptor:

$$I(F,H) = H(R) - \sum_{j=1}^{N} P(F_j)\ H(R|F_j) \tag{3.34}$$

$I(F, H)$=Information gained by partitioning on descriptor F given the entropy H;

$P(F_j)$=Probability of descriptor F with value j;
N=Maximum number of unique classes in a descriptor;
$H(R, F_j)$=Entropy of response for descriptor F with value j.

Consider the data set shown in Table 3.1:

Table 3.1 15 Molecules with information about molecular weight (MW), logP, and number of rotatable bonds. In the final column is a classifier to describe their response (activity).

Compound	MW	logP	No. Rot. Bonds	Response
1	<300	<1.5	5–10	Med
2	>300	>2.5	5–10	Low
3	>300	<1.5	5–10	High
4	>300	<1.5	<5	High
5	>300	<1.5	<5	Med
6	<300	>2.5	>10	Med
7	>300	>2.5	5–10	Low
8	<300	>2.5	>10	Med
9	>300	<1.5	>10	Low
10	<300	1.5–2.5	5–10	High
11	>300	<1.5	>10	Low
12	>300	>2.5	5–10	Low
13	<300	1.5–2.5	<5	High
14	>300	>2.5	>10	Low
15	>300	1.5–2.5	5–10	Med

Now, if we calculate each of the variables for "Low", "Med," and "High" responses, possibly referring to the sensitivity of a cell line to the compound, we get the results as shown in Table 3.2.

Table 3.2 Expanded variables and their calculated values. For example, $\boldsymbol{P(R_{low})}$ calculates to 6/15: 6 low responses divided by 15 total responses.

Variable	Value	Variable	Value	Variable	Value

$P(R_{\text{Low}})$	$\frac{6}{15}$	$P(R_{\text{med}})$	$\frac{5}{15}$	$P(R_{\text{High}})$	$\frac{4}{15}$
$H(R_{\text{Low}}\|MW_{<300})$	$\left(\frac{0}{6}\right)\log\left(\frac{0}{6}\right)$	$H(R_{\text{Med}}\|MW_{<300})$	$\left(\frac{4}{6}\right)\log\left(\frac{4}{6}\right)$	$H(R_{\text{High}}\|MW_{<300})$	$\left(\frac{2}{6}\right)\log\left(\frac{2}{6}\right)$
$H(R_{\text{Low}}\|MW_{>300})$	$\left(\frac{6}{9}\right)\log\left(\frac{6}{9}\right)$	$H(R_{\text{Med}}\|MW_{>300})$	$\left(\frac{1}{9}\right)\log\left(\frac{1}{9}\right)$	$H(R_{\text{High}}\|MW_{>300})$	$\left(\frac{2}{9}\right)\log\left(\frac{2}{9}\right)$
$H(R_{\text{Low}}\|\log P_{<1.5})$	$\left(\frac{2}{6}\right)\log\left(\frac{2}{6}\right)$	$H(R_{\text{Med}}\|\log P_{<1.5})$	$\left(\frac{2}{6}\right)\log\left(\frac{2}{6}\right)$	$H(R_{\text{High}}\|\log P_{<1.5})$	$\left(\frac{2}{6}\right)\log\left(\frac{2}{6}\right)$
$H(R_{\text{Low}}\|\log P_{<1.5-2.5})$	$\left(\frac{0}{3}\right)\log\left(\frac{0}{3}\right)$	$H(R_{\text{Med}}\|\log P_{1.5-2.5})$	$\left(\frac{1}{3}\right)\log\left(\frac{1}{3}\right)$	$H(R_{\text{High}}\|\log P_{1.5-2.5})$	$\left(\frac{2}{3}\right)\log\left(\frac{2}{3}\right)$
$H(R_{\text{Low}}\|\log P_{<2.5})$	$\left(\frac{4}{6}\right)\log\left(\frac{4}{6}\right)$	$H(R_{\text{Med}}\|\log P_{>2.5})$	$\left(\frac{2}{6}\right)\log\left(\frac{2}{6}\right)$	$H(R_{\text{High}}\|\log P_{<2.5})$	$\left(\frac{0}{6}\right)\log\left(\frac{0}{6}\right)$
$(R_{\text{Low}}\|RB_{<5})$	$\left(\frac{0}{3}\right)\log\left(\frac{0}{3}\right)$	$H(R_{\text{Med}}\|RB_{<5})$	$\left(\frac{1}{3}\right)\log\left(\frac{1}{3}\right)$	$H(R_{\text{High}}\|RB_{<5})$	$\left(\frac{2}{3}\right)\log\left(\frac{2}{3}\right)$
$H(R_{\text{Low}}\|RB_{5-10})$	$\left(\frac{3}{7}\right)\log\left(\frac{3}{7}\right)$	$H(R_{\text{Med}}\|RB_{5-10})$	$\left(\frac{2}{7}\right)\log\left(\frac{2}{7}\right)$	$H(R_{\text{High}}\|RB_{5-10})$	$\left(\frac{2}{7}\right)\log\left(\frac{2}{7}\right)$
$H(R_{\text{Low}}\|RB_{>10})$	$\left(\frac{3}{5}\right)\log\left(\frac{3}{5}\right)$	$H(R_{\text{Med}}\|RB_{>10})$	$\left(\frac{2}{5}\right)\log\left(\frac{2}{5}\right)$	$H(R_{\text{High}}\|RB_{>10})$	$\left(\frac{0}{5}\right)\log\left(\frac{0}{5}\right)$

Now, if we calculate the entropy and information gain (see Table 3.3) for each:

$$\begin{aligned} \boldsymbol{H}(\boldsymbol{R}) &= -P(R_{\text{Low}})\log P(R_{\text{Low}}) - P(R_{\text{med}})\log P(R_{\text{med}}) - P\left(R_{\text{High}}\right)\log P\left(R_{\text{High}}\right) \\ &= 0.5288 + 0.5299 + 0.5085 = \mathbf{1.5656} \end{aligned} \tag{3.35}$$

Table 3.3 Calculated information gain (***I***) for each feature.

$\boldsymbol{I}(\boldsymbol{MW}\|\boldsymbol{H}(\boldsymbol{R}))=H(R)-P(MW_{<300})(H(R\|MW_{<300})-P(MW_{>300})(H(R\|MW_{>300})$	0.4637
$\boldsymbol{I}(\boldsymbol{\log P}\|\boldsymbol{H}(\boldsymbol{R}))=H(R)-P(\log P_{<1.5})(H(R\|\log P_{<1.5})-P(\log P_{1.5-2.5})(H(R\|\log P_{1.5-2.5})-P(\log P_{>2.5})(H(R\|\log P_{>2.5})$	0.3806
$\boldsymbol{I}(\boldsymbol{RB}\|\boldsymbol{H}(\boldsymbol{R}))=H(R)-P(RB_{<5})(H(R\|RB_{<5})-P(RB_{5-10})(H(R\|RB_{5-10})-P(RB_{>10})(H(R\|RB_{>10})$	0.3317

We find that the most information is gained by partitioning the data using MW because $I(MW|H(R))$ with a value of **0.4637** is greater than the calculated values of $I(\log P|H(R))$ and $I(RB|H(R))$.

In Table 3.4, we show the same 15 compounds grouped by MW after the first node. Note that the MW <300 grouping does not contain any compounds with "Low" response. The lower MW compounds all show "High" or "Med" responses, proving MW to be a good first pass classifier.

Table 3.4 The data table has been reordered to demonstrate separation based on the first decision node. The lower MW compounds all show high or med responses, proving MW to be a good first pass classifier.

Compound	MW	logP	No. Rot. Bonds	Response
1	<300	<1.5	5–10	Med
5	<300	<1.5	<5	Med
6	<300	>2.5	>10	Med
8	<300	>2.5	>10	Med
10	<300	1.5–2.5	5–10	High
13	<300	1.5–2.5	<5	High
2	>300	>2.5	5–10	Low
3	>300	<1.5	5–10	High
4	>300	<1.5	<5	High
7	>300	>2.5	5–10	Low
9	>300	<1.5	>10	Low
11	>300	<1.5	>10	Low
12	>300	>2.5	5–10	Low
14	>300	>2.5	>10	Low
15	>300	1.5–2.5	5–10	Med

Continuing with the algorithm to determine the next node we use in our decision (see Table 3.5):

Table 3.5 Calculation of entropy and information gain for each feature on the $\boldsymbol{MW_{<300}}$ node.

$\boldsymbol{H(R|MW_{<300})}$	0.9183
$\boldsymbol{I(\mathrm{log}P|MW_{<300})}$	0.9183
$\boldsymbol{H(RB|MW_{<300})}$	0.2516
$\boldsymbol{H(R|MW_{<300})}$	0.9183
$\boldsymbol{I(\mathrm{log}P|MW_{<300})}$	0.9183
$\boldsymbol{I(RB|MW_{<300})}$	0.2516

Thus, partitioning on log$P$ results in all the observations being correctly categorised so partitioning on this branch ($MW_{<300}$) can cease.

For the branch $MW_{<300}$, our expanded variables are shown in Table 3.6:

Table 3.6 Calculation of entropy and information gain for each feature on the $\boldsymbol{MW_{>300}}$ node.

$\boldsymbol{H(R|MW_{>300})}$	1.2244
$\boldsymbol{I(\log P|MW_{>300})}$	0.78
$\boldsymbol{I(RB|MW_{>300})}$	0.4628

We would next partition using log$P$, as the Information gain is greater (0.78).

At this moment, we have a Decision Tree (see Figure 3.7) that starts with root node partitioning the data using MW. The next node at the end of both branches will partition the data according to logP values. The compounds that have MW <300 were all correctly classified using logP. For compounds with MW>300, we will cheat and manually finish off the Decision Tree.

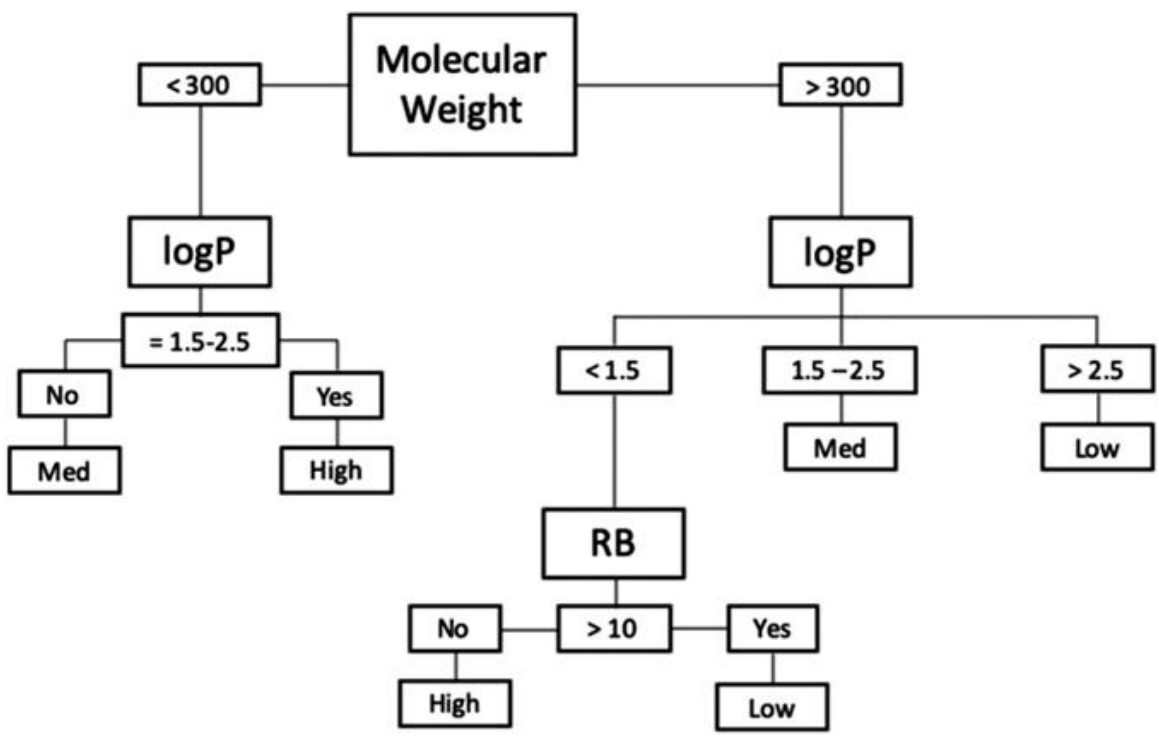


Figure 3.7 A Decision Tree classification model, which determines the priorities of descriptors on which classifications are made.

It is worth noting that the Decision Tree in Figure 3.7 has split the data over three branches. Decision Tree algorithms typically perform binary splits due to computational simplicity, and to avoid a combinatorial explosion of information. In this case, we would obtain the Decision Tree illustrated in Figure 3.8.

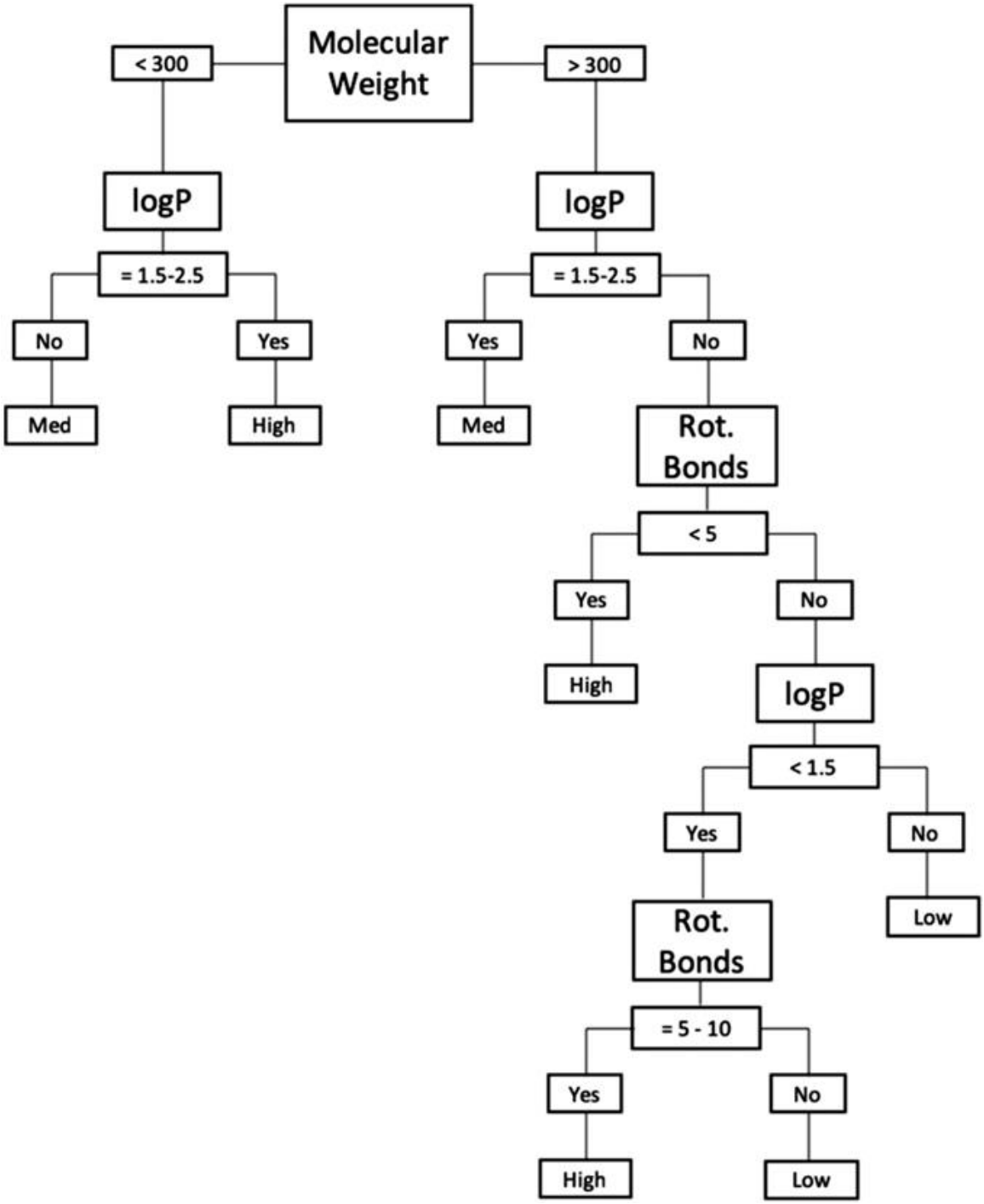


Figure 3.8 Binary Decision Tree classification model using the ID3 algorithm.

The C4.5 algorithm is very similar to ID3 with some small differences. The algorithm still uses information/entropy, but the decision function is

based on *information gain ratio* instead of *information gain*. If the ID3 equations could be viewed generally as:

$$\text{Information gained} = \text{Entropy} - \text{Split Information} \tag{3.36}$$

C4.5 algorithm uses:

$$\text{Gain ratio} = \frac{\text{Information gained}}{\text{Split Information}} \tag{3.37}$$

For more information about C4.5 and Gain ratios, see [40].

Decision Tree algorithms are considered "greedy." Their calculations result in the 'best' decision made for the next immediate node. This does not necessarily result in the optimal classification for the entire dataset. Additionally, a Decision Tree does not incorporate any randomness, so the result will always be the same. To improve classification, ensemble Decision Tree approaches may be used.

3.5.3.3 Decision Tree Ensembles

What is better than one decision tree? Thousands of decision trees, of course! However, it was just mentioned that Decision Trees follow a very strict procedure that would result in the same classification each time, so how would performing thousands of Decision Trees be helpful? By creating Decision Trees with different subsets of the original dataset, the ensemble allows generalisation across large datasets. Ensemble methods can be used to create multiple Decision Trees using random selection of the data. As the data are different, different decision trees will be created. Two different types of ensemble methods are bagging[41] and boosting.[42]

Bagging stands for "*b*ootstrap *ag*gregation." Bootstrapping means to sample with replacement. Therefore, bagging algorithms create new datasets from the original dataset where observations may be duplicated. Decision Trees are then created from these new datasets. Observations then get put through all the Decision Trees and a voting schema between the results from all the Decision Trees will decide how the observation gets classified. This method promotes generality and significantly reduces overfitting. Random

Forest (RF) is a bagging algorithm that, in addition to using bootstrapping, creates decision trees using a random subset of the descriptors.

Boosting is another ensemble Decision Tree method. The goal of boosting algorithms is to strengthen *weak* classifiers; this is achieved by assigning weights to all the observations. Observations that are incorrectly classified have their weight increased while those that are correctly classified experience a decrease in their weighting. It is important to note that decision trees are generated sequentially. The second decision tree is a modified version of the first, the third is modified from the second, and so on.

Final Thoughts

Decision Trees and ensemble methods are an efficient way to classify large amounts of high-dimensional data, with the added benefit that the data can be used as it is, without scaling, transforming, or converting to other forms. On the downside, a large volume of data is required before these methods become truly useful. Further, the approach is simply a sorting algorithm – the model makes no attempt to interpret or understand the relationship between the data and the descriptors. Decision Trees (and Ensembles) have been used in drug discovery for tasks such as choosing the most informative set(s) of descriptors for large collections of compounds, and predicting and classifying on ADMET properties; a few sample articles are cited in ref. 43. More recently, these methods (along with PLS) were used to create Profile-QSAR at Novartis.[44]

3.5.4 Support Vector Machines

Support Vector Machine (SVM) algorithms were initially purposed to classify data into one of two groups with the assumption that the data were linearly separable.[45] Data are separated by finding a hyperplane that separates the data – any data on one side of the hyperplane are classified into one group while data on the other side are classified into a second group. A hyperplane is math jargon for a sheet or plane that cuts through an array of multidimensional data and separates (classifies) it; this separator would be a line if the data are 2-Dimensional.

This means that SVM algorithms are used when the descriptors are numerical and the response data are binary (*i.e.*, yes/no, 1/0, on/off,

active/inactive). SVMs can be broken down into hyperplanes, kernels, and support vectors.

3.5.4.1 Hyperplanes

SVMs find hyperplanes that separate data into their respective categories. But it is not just any hyperplane that successfully separates the data: it is a hyperplane that maximises the distance between the closest point(s) in both classes (see Figure 3.9).

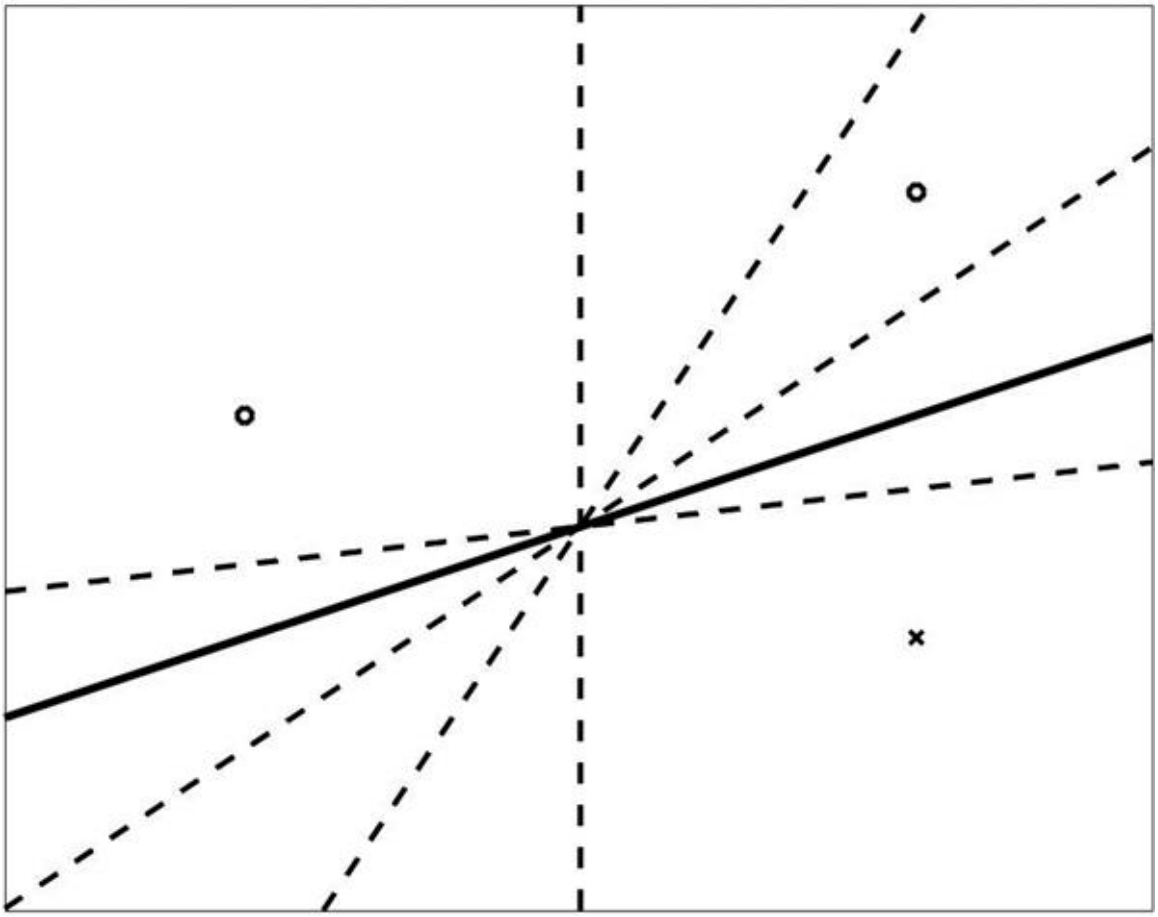


Figure 3.9 Five hyperplanes (black lines) have been generated for three data points (two o's and one x) that represent two classes.

In Figure 3.9, two of the hyperplanes (the two with steepest slopes) do not successfully classify the data points as there is one “o” data point on either side. The two remaining dashed hyperplanes separate the data points

successfully; however, the distances from the hyperplane and the points are not maximised. The solid black hyperplane is the optimal hyperplane, as the data points are successfully classed and the distance from the data points to the hyperplane is maximised. The logic behind maximising the margin (described later) is that if any new data points are added, new points will possibly be close to existing data points of the same class. If a hyperplane was too close to a data point, it could dramatically change in order to accommodate the new data point. Maximising the margin promotes the most stable model.

3.5.4.2 Kernels

Kernels are functions that transform the data or project the data into higher dimensions – the so-called "kernel trick." In Figure 3.10, we illustrate the idea of linearly separable and nonlinearly separable data and discuss kernels that are used for classifying/separating the data.

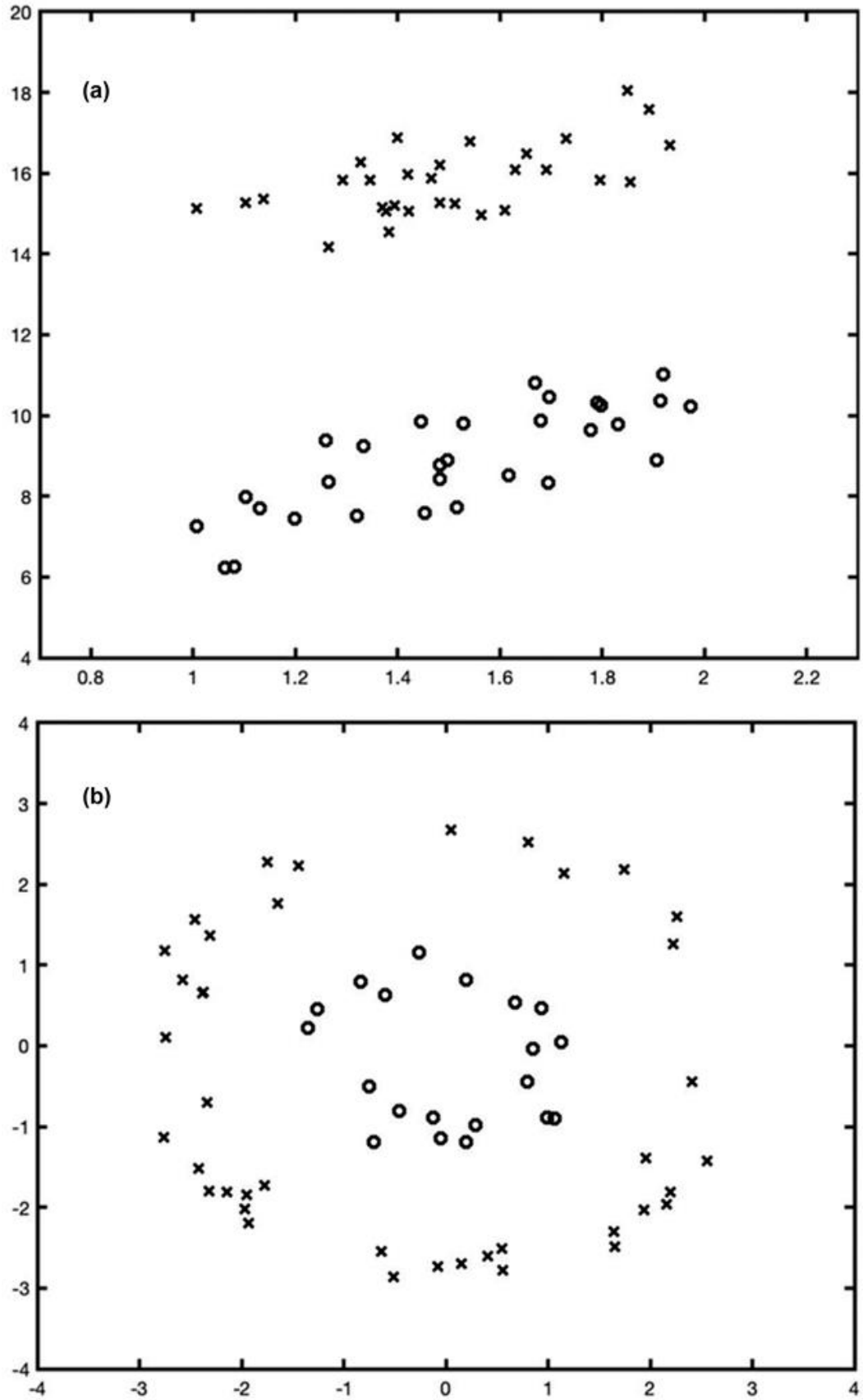
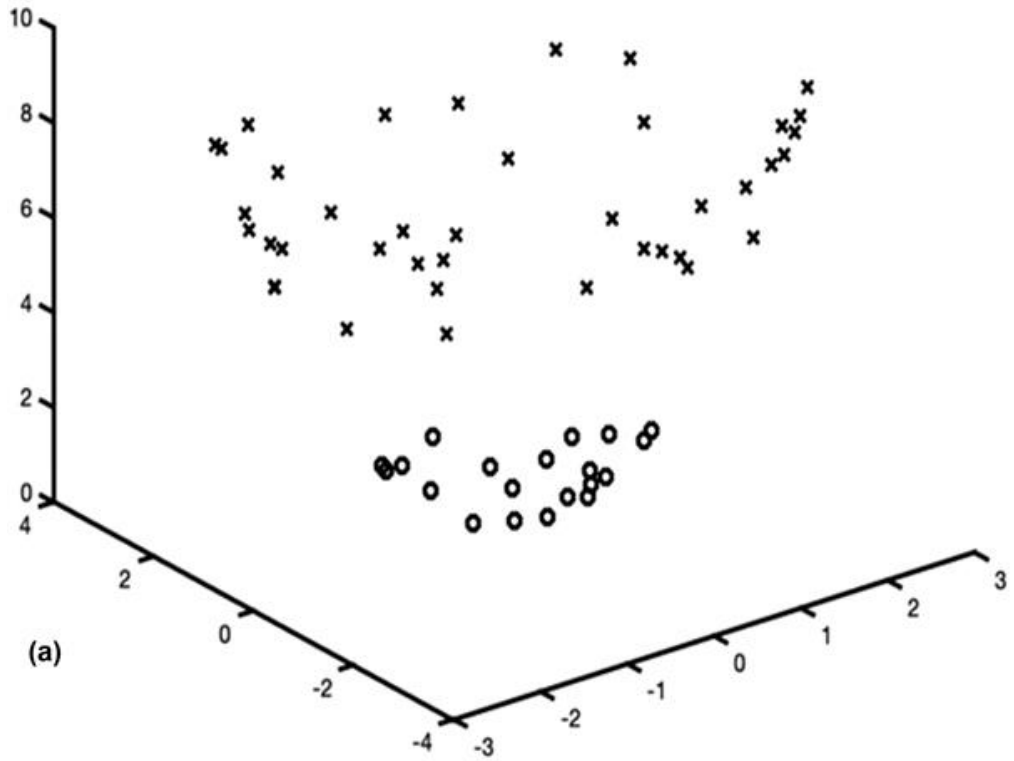

(a)
20
18
16
14
12
10
8
6
4
0.8
1
1.2
1.4
1.6
1.8
2
2.2
(b)
4
3
2
1
0
-1
-2
-3
-4
-4
-3
-2
-1
0
1
2
3
4

Figure 3.10 (a) Linearly separable data, and (b) nonlinearly separable data.

A SVM model may only be successful if the appropriate kernel is used. A linear kernel would not be appropriate if the data were distributed as Figure 3.10(b). When data are not linearly separable, different kernels need to be used. For example, in Figure 3.10(b), we have data plotted in 2D. If we project the data into a third dimension, Z, where $Z=x^2+y^2$, we can see that the data separates (see Figure 3.11).


(a)

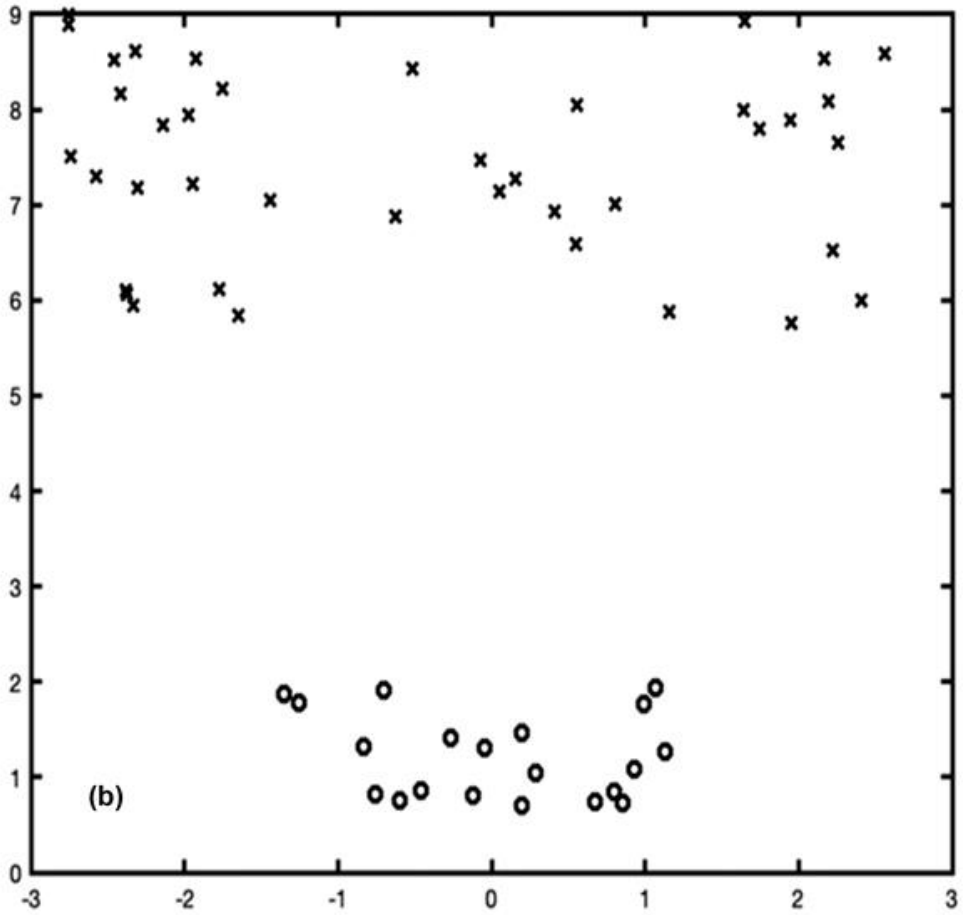

(b)

Figure 3.11 (a) Data from Figure 3.10(b) projected into the third dimension; (b) The projected data are then plotted on the *X*–*Z* axis to highlight how the nonlinearly separable data are now linearly separable after transformation.

Post-transformation, we can find a value of *Z* that classifies the data. With this example, we see that using the right kernel is fundamental to the success of the SVM model.

3.5.4.3 Support Vectors

As previously described, the optimal hyperplane depends on the closest point(s) from either class. The distance between the hyperplane and the closest points is called the margin, and the closest points to the margin are called support vectors.

In Figure 3.12, a hyperplane is generated (dark black line). The normal to the hyperplane is a unit vector that is perpendicular to the hyperplane. The arrow indicates the direction of the unit vector towards a data point. It does not matter where on the hyperplane the normal is found.

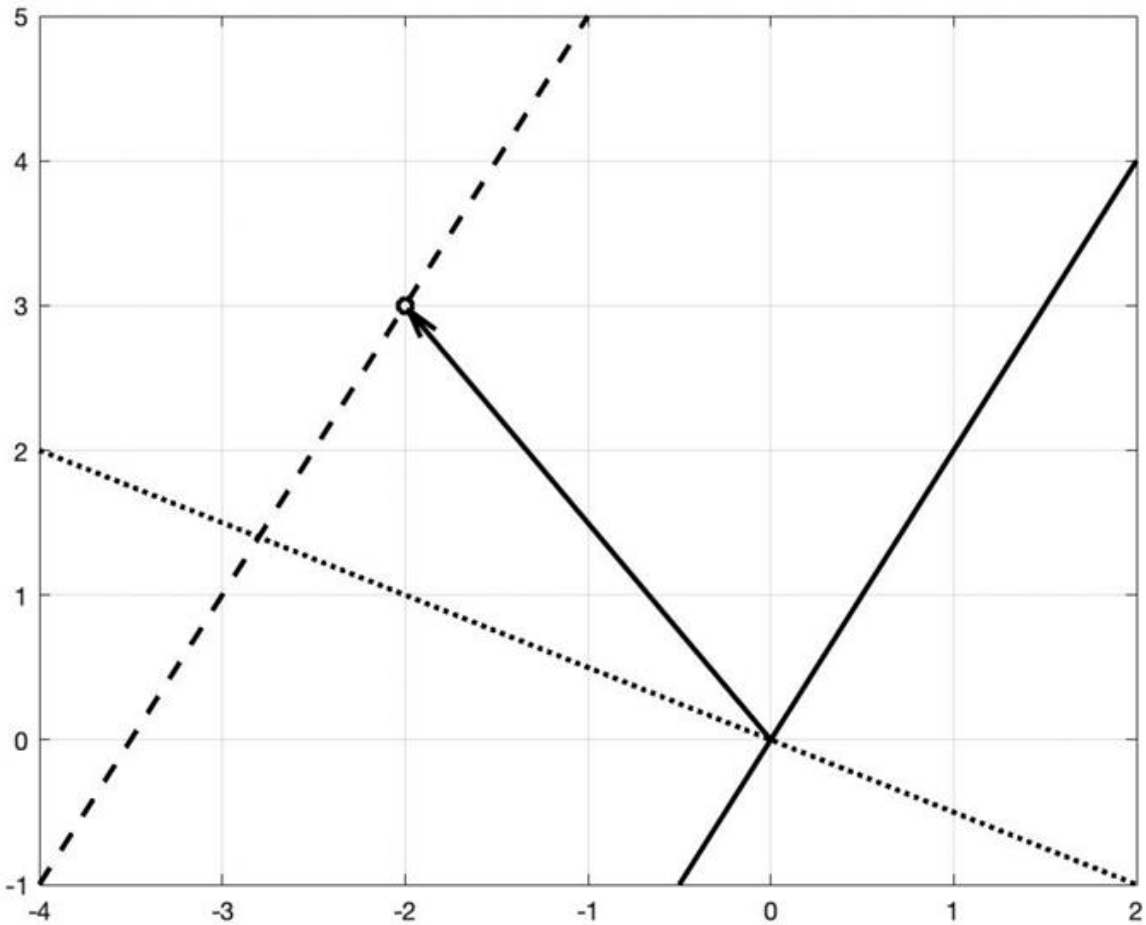


Figure 3.12 The projected vector used to calculate the distance from the hyperplane to the data point; the arrow indicates the direction of the unit vector to the data point in question. The hyperplane is displayed as a dark line and the normal is rendered as a dotted line.

The calculation proceeds by the following steps:

1. Select a point on the hyperplane.
2. Find the normal, **w**, at that point.
3. Ensure that the normal is converted to a unit vector, **u**.
4. Generate vector **p**, which is the projection of the data point vector **a** onto the direction of the unit vector, **u**.

$$\mathrm{a} \cdot p = (u \cdot a) * u \tag{3.38}$$

5. We want the magnitude of **p** for the closest data point in one class to be the same for the magnitude of **p** for the closest point in the second

class.

In calculating the vector **p**, we calculated the distance of a point from the hyperplane, (***u*** · ***a***). The dot product produces a dimensionless number (scalar), which gives us the magnitude of the distance to the hyperplane.

Now, if we define a normal vector (**w**) and a vector **x**, where **x**=(x1, x2,…) and *b* is the intersection:

$$(\boldsymbol{w} \cdot \boldsymbol{x}) + b = 0 \tag{3.39}$$

This equation defines the hyperplane. But, for points that are NOT on the hyperplane, the expression will not equal zero:

$$(\boldsymbol{w} \cdot \boldsymbol{x}) + b \neq 0 \tag{3.40}$$

The dot product, representing distance, can be positive or negative, depending on which side of the hyperplane a point lies. Now, if we define *y* as:

$$y = 1 \text{ if } (\boldsymbol{w} \cdot \boldsymbol{x}) + b > 0 \text{ and} \tag{3.41}$$

$$y = -1 \text{ if } (\boldsymbol{w} \cdot \boldsymbol{x}) + b < 0$$

All points are correctly classified if:

$$y(\boldsymbol{w} \cdot \boldsymbol{x}) + b > 0 \tag{3.42}$$

For each hyperplane, the equation to be optimised (*Margin*) is given by:

$$\text{Margin} = \min_{w,b} y\left(\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\right) \cdot x + \frac{b}{\|\boldsymbol{w}\|} \tag{3.43}$$

The normal, **w**, has been scaled by its magnitude, $\|\boldsymbol{w}\|$, to make the equation scale invariant.

If the equation is not scaled, the magnitude of the normal vector would greatly affect the results; the only factor should be the direction of the normal vector. This is achieved by converting the normal vector to a unit

normal vector. With an equation that is scale invariant, multiple hyperplanes with different normal vectors can be compared.

Margin finds the distance of the closest point for each hyperplane, and this is the value that we want to maximise across all the hyperplanes.

$$\max \text{Margin},$$

$$\text{such that}: y\left(\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\right) \cdot x + \frac{b}{\|\boldsymbol{w}\|} \geq \text{Margin} \tag{3.44}$$

As real data are noisy, data won't always be entirely separable. Therefore, SVM algorithms can include hard and soft margins. Hard margins mean all data need to be correctly classified, whereas soft margins allow for some data points to be misclassified in order to improve generability.

Solutions to SVM algorithms are unique, as a result of it being a convex optimisation problem. Additional information regarding Support Vector Machines can be found in ref. 46.

Final Thoughts

One of the key advantages of SVM is that one very quickly arrives at a classification model. But there are considerations about applying it to large amounts of data (more than 10 000 data points) as the approach – like other canonical ML methods – does not always scale with the amount of data. This is why Deep Learning is used when the amount of data is vast. SVMs have been used in drug discovery for property prediction, assessing synthetic accessibility, and for exploring activity cliffs – a few examples can be found in ref. 47*a*–*d*.

3.6 Summary

Phew. That was a lot of maths. Congratulations if you have continued to read – you have cemented your reputation as a medcheminformatician who knows their stuff!

Now that you have been introduced to some popular ML algorithms/methods/tools, you may be thinking, “what is the best method?” Well, cross-validation does not only apply to datasets (remember to leave

some out!) but also to ML approaches – in a nutshell, try them all and see what the models look like. The success of a ML approach depends on the inherent structure and values of the dataset. Some approaches will not be applicable at all, depending on your modelling needs (classifying or regression) or whether your data are numerical or not.

Reportedly, SVM and ensemble Decision Tree approaches have been found to be more accurate for classifications in general;[48] however, there is no one method that works best for all problems.

The good (and incredibly relieving) news is that most commercial software packages have implementations of these algorithms with sensible starting parameters, so you can simply run ML models on your data with a few button clicks without worrying about the underlying science or the nuts and bolts of the approaches. However, it is our view (and the view of most computational scientists) that mindless button clicking is not a good approach to modelling. Awareness of the underlying algorithms and their strengths and limitations will allow you to select the most appropriate method for the data you are attempting to model and ultimately help avoid the trials, tribulations, and endless aggravation when the method you selected produces incomprehensible results.

Acknowledgements

The authors are supported by the CRUK Manchester Institute Drug Discovery Unit (C5759/A17098) and are immensely grateful that our DDU Head, Prof. Caroline Springer, championed our work. We also thank Dr Simon Pearce and Dr Angelo Pugliese for their helpful suggestions and review of the material.

References

1. https://deepmind.com/(last accessed September 2019).
2. (a) J. De Fauw, J. R. Ledsam, B. Romera-Paredes, S. Nikolov, N. Tomasev, S. Blackwell, H. Askham, X. Glorot, B. O'Donoghue, D. Visentin, G. van den Driessche, B. Lakshminarayanan, C. Meyer, F.

Mackinder, S. Bouton, K. Ayoub, R. Chopra, D. King, A. Karthikesalingam, C. O. Hughes, R. Raine, J. Hughes, D. A. Sim, C. Egan, A. Tufail, H. Montgomery, D. Hassabis, G. Rees, T. Back, P. T. Khaw, M. Suleyman, J. Cornebise and P. A. Keane, *Nat. Med.*, 2018, **24**(9), 1342; (b) J. J. Titano, M. Badgeley, J. Schefflein, M. Pain, A. Su, M. Cai, N. Swinburne, J. Zech, J. Kim, J. Bederson, J. Mocco, B. Drayer, J. Lehar, S. Cho, A. Costa and E. K. Oermann, *Nat. Med.*, 2018, **24**(9), 1337; (c) C. J. Lynch and C. Liston, *Nat. Med.*, 2018, **24**(9), 1304.

3. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel and D. Hassabis, *Nature*, 2016, **529**, 484.
4. M. H. S. Segler, M. Preuss and M. P. Waller, *Nature*, 2018, **555**, 604–610.
5. According to a recent PR Newswire article, the AI in healthcare market was worth USD$667M (2016), and project to reach close to USD$8B by 2022. https://www.prnewswire.com/news-releases/global-artificial-intelligence-in-healthcare-market-2017-2022-market-is-expected-to-reach-usd-79888-million-at-a-cagr-of-5268—research-and-markets-300464679.html.
6. UK-based BenevolentAI recently raised $115M for AI-enabled drug development; Exscientia has signed deals with Celgene, GSK and Sanofi; and celebrated a recent success milestone in the GSK COPD collaboration delivering an in vivo active lead, through 5 cycles and only 85 compounds tested. Further, the UK launched the AI Sector Deal, a £1B fund to make the UK a global leader. https://www.gov.uk/government/publications/artificial-intelligence-sector-deal/ai-sector-deal; https://benevolent.ai/news/announcements/benevolentai-raises-115m-for-ai-enabled-drug-development; https://www.exscientia.co.uk/news/2019/3/20/celgene-and-exscientia-enter-3-year-ai-drug-discovery-collaboration-focused-on-accelerating-drug-discovery-in-oncology-and-autoimmunity;

https://www.exscientia.co.uk/news/2019/4/3/exscientia-achieves-discovery-milestone-in-gsk-collaboration.

7. (a) C. J. Hansch, *J. Comput.-Aided Mol. Des.*, 2011, **25**(6), 495; (b) C. J. Hansch, P. P. Maloney, T. Fujita and R. M. Muir, *Nature*, 1962, **194**, 178.
8. S. M. Free and J. W. Wilson, *J. Med. Chem.*, 1964, **7**(4), 395.
9. S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang and E. E. Bolton, *Nucleic Acids Res.*, 2019, **47**(D1), D1102.
10. A. Gaulton, A. Hersey, M. Nowotka, A. P. Bento, J. Chambers, D. Mendez, P. Mutowo, F. Atkinson, L. J. Bellis, E. Cibrián-Uhalte, M. Davies, N. Dedman, A. Karlsson, M. P. Magariños, J. P. Overington, G. Papadatos, I. Smit and A. R. Leach, *Nucleic Acids Res.*, 2017, **45**(D1), D945.
11. (a) P. Tiikkainen, L. Bellis, Y. Light and L. Franke, *J. Chem. Inf. Model.*, 2013, **53**, 2499; (b) I. Cortes-Ciriano, A. Bender and T. E. Malliavin, *J. Chem. Inf. Model.*, 2015, **55**(7), 1413; (c) A. Hersey, J. Chambers, L. Bellis, A. P. Bento, A. Gaulton and J. P. Overington, *Drug Discovery Today*, 2015, **14**, 17; (d) G. Papadatos, A. Gaulton, A. Hersey and J. P. Overington, *J. Comp.-Aided Mol. Design*, 2015, **29**(9), 885; (e) K. Mansouri, C. M. Grulke, A. M. Richard, R. S. Judson and A. J. Williams, *SAR QSAR Environ Res.*, 2016, **27**(11), 911.
12. Y.-C. Lo, S. E. Rensi, W. Torng and R. B. Altman, *Drug Discovery Today*, 2018, **23**(8), 1538.
13. S. Wold, K. Esbensen and P. Geladi, *Chemom. Intell. Lab. Syst.*, 1987, **2**, 37.
14. (a) ROCS 3.3.1.2: OpenEye Scientific Software, Santa Fe, NM; (b) P. C. D. Hawkins, A. G. Skillman and A. Nicholls, *J. Med. Chem.*, 2007, **50**, 74.
15. T. Cheeseright, M. Mackey, S. Rose and A. Vinter, *J. Chem. Inf. Model.*, 2006, **46**(2), 665.
16. (a) Y. Xu, J. Ma, A. Liaw, R. P. Sheridan and V. Svetnik, *J. Chem. Inf. Model.*, 2017, **57**, 2490; (b) https://www.kaggle.com/c/MerckActivity.
17. G. E. Dahl, N. Jaitly and R. Salakhutdinov, 2014, https://arxiv.org/abs/1406.1231.
18. Y. Ogata and M. Tsuchida, *Ind. Eng. Chem.*, 1957, **49**(3), 415.

19. (a) S. Sagarika, C. Adhikari, M. Kuanar and B. K. Mishra, *Curr. Comput Aided Drug Des.*, 2016, **12**(3), 181; (b) A. Mauri, V. Consonni and R. Todeschini, *Handbook of Computational Chemistry*, ed. J. Leszczynski *et al.*, Springer International Publishing, Switzerland, **ch. 49**, 2017, pp. 2066–2093.
20. (a) R. D. Cramer III, D. E. Patterson and J. D. Bunce, *J. Am. Chem. Soc.*, 1988, **110**(18), 5959; (b) M. Clark, R. D. Cramer III, D. M. Jones, D. R. Patterson and P. E. Simeroth, *Tetrahedron Comput. Methodol.*, 1990, **3**(1), 47.
21. (a) R. Guha and E. Willighagen, *Curr. Top. Med. Chem.*, 2012, **12**(18), 1946; (b) Y.-C. Lo, S. E. Rensi, W. Torng and R. B. Altman, *Drug Discovery Today*, 2018, **23**(8), 1538; (c) D. Stumpfe and J. Bajorath, *Wiley Interdiscip. Rev.: Comput. Mol. Sci*, 2011, 260.
22. (a) T. T. Tanimoto, *An Elementary Mathematical Theory of Classification and Prediction in Internal IBM Technical Report*, 1957; (b) D. J. Rogers and T. T. Tanimoto, *Science*, 1960, **132**(3434), 1115; (c) P. Willett, J. M. Barnard and G. M. Downs, *J. Chem. Inf. Comput. Sci.*, 1998, **38**, 983; (d) G. Maggiora, M. Vogt, D. Stumpfe and J. Bajorath, *J. Med. Chem.*, 2013, **57**, 3186.
23. (a) H. Eckert and J. Bajorath, *Drug Discovery Today*, 2007, **12**, 225; (b) Y. C. Martin, J. L. Kofron and L. M. Traphagen, *J. Med. Chem.*, 2002, **45**, 4350.
24. J. W. Godden, L. Xue and J. Bajorath, *J. Chem. Inf. Comput. Sci.*, 2000, **40**, 163.
25. A. Tversky, *Psychol. Rev.*, 1977, **84**, 327.
26. L. R. Dice, *Ecology*, 1945, **26**, 297.
27. A. Golbraikh and A. Tropsha, *J. Mol. Graph. Model.*, 2002, **20**, 269.
28. M. A. Kendall, *Biometrika*, 1938, **30**(1–2), 81.
29. (a) M. A. Khanfar and M. O. Taha, *J. Chem. Inf. Model.*, 2013, **53**, 2587; (b) T. S. Chitre, K. D. Asgaonkar, S. M. Patel, S. Kumar, V. M. Khedkar and D. R. Garud, *Comp. Bio. & Chem.*, 2017, **68**, 211.
30. J. Liu, G. Patlewicz, A. J. Williams, R. S. Thomas and I. Shah, *Chem. Res. Toxicol.*, 2017, **30**(11), 2046.
31. S. Chavan, A. Abdelaziz, J. G. Wiklander and I. A. Nichols, *J. Comp.-Aided Mol. Design*, 2016, **30**(3), 229.
32. K.-C. Chou, *J. Theo. Biol.*, 2011, **273**(1), 236.

33. O. Sutton, *Introduction to k Nearest Neighbour Classification and Condensed Nearest Neighbour Data Reduction*, University of Leicester, 2012. Available: http://www.math.le.ac.uk/people/ag153/homepage/KNN/OliverKNN_Talk.pdf.
34. (a) R. Tibshirani, *J. Roy Stat. Soc. Series B*, 1996, **58**(1), 267; (b) R. Tibshirani, *Stat. Med.*, 1997, **16**(4), 385.
35. S. Reid, R. Tibshirani and J. Friedman, *J. Statistica Sinica*, 2016, **26**, 35.
36. H. Wold, Estimation of principal components and related models by iterative least squares, in *Multivariate Analysis*, ed. P. R. Krishnaiah, Academic Press, New York, 1996, pp. 391–420.
37. A. E. Stott, S. Kanna, D. P. Mandic and W. T. Pike, An online NIPALS algorithm for Partial Least Squares. International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2017, DOI: 10.1109/ICASSP.2017.7952943.
38. R. D. Cramer III, *Perspect. Drug Discovery Des.*, 1993, **1**, 269.
39. J. R. Quinlan, *Mach. Learn.*, 1986, **1**, 81.
40. (a) J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kauffmann, San Mateo, CA, 1992; (b) S. L. Salzberg, *Mach. Learn.*, 1994, **16**, 235.
41. L. Breiman, *Mach. Learn.*, 1996, **24**(2), 123.
42. Y. Freund and R. E. Shapire, *J. Comput. Syst. Sci.*, 1997, **55**(1), 119.
43. (a) G. Cano, J. Garcia-Rodriguez, A. Garcia-Garcia, H. Perez-Sanchez, J. A. Benediktsson, A. Thapa and A. Barr, *Expert Syst. App*, 2017, **72**, 151; (b) E. Deconick, M. H. Zhang, D. Coomans and Y. Van der Heyden, *J. Chem. Inf. Model.*, 2006, **46**, 1410; (c) M. P. Gleeson, N. J. Waters, S. W. Paine and A. M. Davis, *J. Med. Chem.*, 2006, **49**(6), 1953; (d) C. Lamanna, M. Bellini, A. Padova, G. Westerberg and L. Maccari, *J. Med. Chem.*, 2008, **51**(10), 2891; (e) I. Olier, N. Sadawi, G. R. Bickerton, J. Vanschoren, C. Grosan, L. Soldatova and R. D. King, *Mach. Learn.*, 2018, **107**(1), 285; (f) P. Zane, H. Gieschen, E. Kersten, N. Mathias, C. Ollier, P. Johansson, A. Van den Bergh, S. Van Hemelryck, A. Reichel, A. Rotgeri, K. Schäfer, A. Müllertz and P. Langguth, *Eur. J. Pharm. Biopharm.*, 2019, **142**, 222.

44. E. J. Martin, V. R. Polyakov, X.-W. Zhu, L. Tian, P. Mukherjee and X. Liu, *J. Chem. Inf. Model.*, 2019, DOI: 10.1021/acs.jcim.9b00375.
45. C. Cortes and V. N. Vapnikm, *Mach. Learn.*, 1995, **20**(3), 273.
46. https://shuzhanfan.github.io/2018/05/understanding-mathematics-behind-support-vector-machines/.
47. (a) D. Horvath, G. Marcou, A. Varnek, S. Kayastha, A. de la Vega de Leon and J. Bajorath, *J. Chem. Inf. Model.*, 2016, **56**(9), 1631–1640; (b) Y. Podolyan, M. A. Walters and G. Karypis, *J. Chem. Inf. Model.*, 2010, **50**(6), 979; (c) T. Cheng, Q. Li, Y. Wang and S. H. Bryant, *J. Chem. Inf. Model.*, 2011, **51**(2), 229; (d) V. G. Maltarollo, T. Kronenberger, G. Z. Espinoza, P. R. Oliveira and K. M. Honorio, *Expert Opin. Drug Discovery*, 2018, **14**(1), 23.
48. M. Fernandez-Delgado, E. Cernadas and S. Barro, *J. Mach. Learn. Res.*, 2014, **15**, 3133.

CHAPTER 4

Machine Learning for Nonadiabatic Molecular Dynamics

JULIA WESTERMAYR(0000-0002-6531-0742)[a] AND PHILIPP MARQUETAND(0000-0002-8711-1533)*[a,b,c]

[a] University of Vienna, Faculty of Chemistry, Institute of Theoretical ChemistryWähringer Str. 171090 ViennaAustria
[b] Vienna Research Platform on Accelerating Photoreaction Discovery, University of ViennaWähringer Str. 171090 ViennaAustria
[c] University of Vienna, Faculty of Chemistry, Data Science @ Uni ViennaWähringer Str. 291090 ViennaAustria
*Email: philipp.marquetand@univie.ac.at

4.1 Introduction

Nonadiabatic molecular dynamics simulations (NAMD) go beyond Born–Oppenheimer molecular dynamics by including two or more electronic states that are coupled. By doing so, many fundamental processes that are at the origin of life and death on earth, such as photosynthesis or DNA photodamage, can be studied.[1–3] To name only a few applications, getting to know the relation between a molecular structure and its photochemical properties can help to design new materials for photovoltaics[4,5] or drugs for phototherapy.[6]

Despite the importance of NAMD simulations, they are limited by the expense of the underlying quantum chemical computation of the high-dimensional potential energy surfaces (PESs), *i.e.*, the energetic states for corresponding molecular configurations that are involved in the targeted reaction. This computation is the most expensive part of a NAMD simulation and hampers its application for long time scales. With contemporary approaches, up to a couple of picoseconds can be simulated and a compromise between accuracy and computational efficiency usually has to be made.[7–9] To speed up simulations and make long time scales as well as accurate PESs accessible, the expensive quantum chemical calculations have to be replaced by more efficient methods. For the electronic ground state force fields exist,[10–13] but these introduce a lot of approximations leading to less accurate potentials. Recently, with the rise of machine learning (ML), ML potentials have been successfully developed for the electronic ground state[14–39] that allow an accurate representation of the PES and the investigation of chemical reactions, which is not possible with standard force fields. However, no such force fields for excited states are available, and only a few studies have tackled the development of ML potentials for excited-state properties.[40–50] The reason for the slow development of ML potentials for the excited state to carry out dynamics simulations compared with the ML potentials for the ground state is two-fold. On one side, not just one PES must be accurately described; more need to be taken into account. Further, not only the energies and corresponding forces are necessary to carry out accurate NAMD simulations, but also couplings between different states; this leads to additional challenges for the regression with ML, as will be explained below.

Almost all approaches in quantum chemistry are based on the Born–Oppenheimer approximation that allows one to separate the degrees of freedom of the electrons and the nuclei. The Born–Oppenheimer approximation is valid for many geometries of the electronic ground state, in particular for the equilibrium of most molecules. However, it breaks down whenever two or more states are strongly coupled and the electrons and the nuclei rearrange on an ultra-short time scale.[51–53] Nevertheless, the use of this approximation is crucial for efficient computations of NAMD. It

allows for mixed quantum-classical dynamics, where the motion of the nuclei on the PESs made up by the electrons is treated classically, while the electronic motion follows the time-dependent Schrödinger equation. The underlying elements that couple different PESs show singularities in regions where PESs are close to each other and are almost vanishing elsewhere. In the conformational space, where states of the same spin multiplicity get into proximity, nonadiabatic couplings (NACs) are important, while spin–orbit couplings play a key role for transitions between states of different spin multiplicities. These critical regions are called conical intersections or simply state crossings, respectively, and do not only pose an obstacle to quantum chemistry but consequently also to the fitting of the PESs.[44–46,50]

The numerical difficulties that arise due to the singularities of NACs at conical intersections lead to other approaches that focus on the diabatization of adiabatic potentials.[48,54–57] Using diabatic potentials is favourable, as the off-diagonal coupling elements of the Hamiltonian are smoothly varying properties. Another advantage is that effects due to the Berry (or geometric) phase vanish and do not influence dynamics simulations. The importance of the Berry phase depends on the dynamics method and the chemical system under study. While the Berry phase can be safely neglected in some cases,[50,58] it plays a vital role in others, leading to path-dependent transition probabilities in the vicinity of conical intersections.[59] For the latter cases, a diabatic approach is desirable, but finding meaningful diabatic potentials is often a tedious task.[48] The output of a quantum chemistry calculation is given in the adiabatic picture, as the diabatic wavefunction is not an eigenfunction of the electronic Hamiltonian. Hence, the diabatic potentials are not unique and it often turns out to be infeasible to find diabatic potentials, especially for larger and more complex molecules. One reason is that the number of electronic states within a certain energy range often increases with the number of atoms in the molecule, therefore increasing the computational effort to provide all desired electronic states. Recently, some approaches were developed to improve the diabatization process by using ML models.[47–49] In large molecules, *i.e.*, in the range of 10s to 100s of atoms, more extensive approximations,[47,60] such as the linear vibronic coupling model,[61] should be considered. As their discussion goes beyond

the scope of this chapter, the reader is referred to ref. 54 and 61–64 for more details. Although excited-state dynamics are favourable in the diabatic picture, due to the aforementioned problems and the fact that the computation of diabatic potentials is still a challenging task, on-the-fly NAMD in the adiabatic representation remains often the method of choice.

Within NAMD simulations, different approximations exist to account for nonadiabatic transitions between different adiabatic states. The surface-hopping methodology is often a good compromise between accuracy and efficiency because it computes the PESs and corresponding properties on-the-fly at geometries obtained after classical propagation of the nuclei.[65,66] This method has been the method of choice in recent ML-driven NAMD simulations[44–46,50] and allows for an on-the-fly exploration of the PESs with ML models.[44,50] Transitions, or the so-called hops, between different PESs are determined stochastically. Different algorithms[58,66,67] to compute the hopping probability and its direction are available with the most popular one being based on Tully's fewest switching algorithm.[68] In this case, the hopping probability is dependent on the couplings between adjacent states.[65] Another commonly used algorithm is the Zhu–Nakamura approximation,[58] that omits the computation of any couplings. For more information on hopping algorithms, the reader is referred to ref. 68–77.

Here, we discuss several ML approaches for the computation of NAMD simulations. In the following sections, different algorithms, *i.e.*, a linear model (LR), kernel-ridge regression (KRR), support vector regression (SVR), and artificial neural networks (NNs), will be compared and their advantages and disadvantages will be listed. It is further shown how to efficiently generate a training set by keeping in mind that the number of costly quantum chemical reference calculations should be small. The problem of wave function phases will be discussed and its impact on couplings, and in general any excited-state properties corresponding to off-diagonal elements in matrix notation, such as transition dipole moments. As an example, the methylenimmonium cation is selected, where the training set is taken from ref. 50 and contains energies, corresponding forces, NACs as well as permanent and transition dipole moments for three singlet states. However, the latter property will not be discussed in the course of this chapter.

4.2 Methods

4.2.1 Machine Learning (ML) Models

As ML models, KRR, SVR, and NNs are discussed and compared with LR as a baseline model. The focus of this chapter is on NNs. In case of LR, SVR, and KRR, each electronic state is independently modelled, so for three singlet states of the methylenimmonium cation containing six atoms, the respective state energies must be determined in three separate fits for one molecular geometry, the gradients (if not fitted as energy derivatives) as single values 18 times, and the NACs as single values 162 times. By using the relations of NAC vectors between two states, *i* and *j*, where $NAC_{ij}=-NAC_{ji}$ as well as $NAC_{ii}=0$, the parameters can be reduced to 54. As NNs can relate all energetic states to one molecular geometry at once, the main focus will be on this model, which we call a multi-state model. The gradients are treated as derivatives of the PESs[20] and NACs are treated together in one additional NN.

4.2.1.1 Linear Model (LR)

The central goal of supervised ML, and in general of regression, is to fit a function to find the best possible relation between an input, *i.e.*, the molecular geometry, and its corresponding output,[78] in our case the energies and corresponding forces for each state, as well as NACs between each set of states. In the simplest case, the relationship can be approximated by a linear function,

$$Y = b + w \cdot X, \tag{4.1}$$

where Y is the output property, X is the data to be fitted, w are the coefficients, and b is a constant bias. Ordinary least squares regression is used for the linear model as implemented in scikit-learn.[79] To fit the PESs, a training set is used that contains X as well as corresponding Ys. The learning process is based on finding the parameters, w and b, where the error between a predicted property, Y^p, by the model and the original value, Y, for a given input, is minimal.[80]

The corresponding loss function reads

$$L = \sum_{i}^{N} \left(Y_i^p - Y_i\right)^2 \tag{4.2}$$

with i running over all data points N in the training set.[81] In the first step, the parameters w and b are estimated or randomly chosen. They get optimized during a process termed training, in which the loss function is minimized,

$$\min_{w} \sum_{i}^{N} \left(\langle w, Y_i^p \rangle - Y_i\right)^2. \tag{4.3}$$

The resulting problem is thus a minimization problem that can be solved by following the gradient of the function with respect to the weights.

4.2.1.2 Kernel Ridge Regression (KRR)

Kernel ridge regression (KRR) is derived from ridge regression instead of LR in combination with the kernel trick. Ridge regression adds a penalty term to the loss function, that is minimized according to:

$$\min_{w} \sum_{i}^{N} \left(\langle w_i, Y_i^p \rangle - Y_i\right)^2 + \lambda \|w\|^2. \tag{4.4}$$

λ is a regularizer, also known as L2 penalty, that is added to the weights and used to control overfitting.[78] As the noise in the training set made up by quantum chemistry is supposed to be small when calculations are converged, λ is also very small in these cases.[82] The kernel trick is used to map a training set that has no linear relation between inputs and outputs into a higher-dimensional space, where it is linearly separable. This leads to a non-linear model with the non-linearity being determined by the type of the used kernel, K.[78] In most cases, a Gaussian kernel or Laplacian kernel is chosen, centred on each compound (molecule), X_i, in the training set. The prediction of a property, Y^p, is then computed using eqn (4.5). The

regression coefficients are again found *via* linear regression according to eqn (4.6).

$$Y^p = \sum_i^N w_i K(X, X_i) \tag{4.5}$$

$$w = (K + \lambda 1)^{-1} Y \tag{4.6}$$

For KRR, the QML toolkit[83] in combination with the FCHL representation[82] was used relying on a Gaussian kernel function and treating gradients as energy derivatives. Note that a new version of the FCHL representation[84] already exists, allowing for more efficient computations. KRR is often the method of choice for quantum chemical ML problems. The advantage of this method is that it can fit non-linear data sets, while still profiting from simplicity and ease of use.[82]

4.2.1.3 Support Vector Regression (SVR)

Support vector machines can be used for classification as well as regression problems. Here, the focus is on the latter case, also known as support vector regression (SVR). Like KRR, SVR makes use of the kernel trick. Input data are mapped into a higher-dimensional feature space, where a linear model can be constructed. However, instead of a minimum loss function, an ϵ-intensive loss function, L_ϵ, is employed, where ϵ is a margin of tolerance:[85]

$$L_\epsilon = \begin{cases} 0 \text{ if } \left|Y_i^p - Y_i\right| \leq \epsilon \\ \left|Y_i^p - Y_i\right| - \epsilon \text{ otherwise} \end{cases} \tag{4.7}$$

The ϵ-insensitive region (where L_ϵ=0) is often referred to as ϵ-*tube*, where the optimization process is based on finding the tube with most of the training points inside. The support vectors represent the instances outside this tube, which have the most influence in the training process.[86] Hence, the loss function is only affected by data points that are predicted with a larger error than ±ϵ.[87] The function to be minimized is then a regularized error function:

$$\min_{w} C \sum_{i}^{N} L_{\epsilon} + \frac{1}{2} \|w\|^2 \tag{4.8}$$

where C is a regularization parameter written in front of the error term.[88]

Common kernel types, or basis functions, that are placed on each training point, are linear kernels, radial basis functions, and polynomials. For the results given here, a radial basis function is used as implemented in scikit-learn.[79] SVR has the advantage of having the capability of good generalization while achieving high prediction accuracy. However, as with the previous models that rely on basis functions that are centred on the training compounds, the depth of the model is directly dependent on the size of the training set.[86]

4.2.1.4 Neural Networks (NNs)

In contrast to LR, KRR, and SVR, neural networks (NNs) have their basic concepts from the information processing in the human brain. As can be seen from Figure 4.1, an NN consists of several layers, with an input layer, that contains the input, X, several hidden layers, each comprised of a finite number of nodes, and an output layer that gives the desired property, Y. Similar to the neurons in our brain that are connected by synapses, the nodes i and k in the hidden layers j and l are connected by weights, $w^{kl}{}_{ij}$, that are the fitting parameters during training. The nested functional form of the NN with two hidden layers and three nodes per hidden layer (as given in Figure 4.1) is presented in eqn (4.9).

$$Y^p = f_1^3 \left\{ b_1^3 + \sum_{l=1}^{3} w_{1k}^{23} f_k^2 \left[b_k^2 + \sum_{j=1}^{3} w_{jk}^{12} f_j^1 \left(b_j^1 + \sum_{i=1}^{2} w_{ij}^{01} X_i \right) \right] \right\} \tag{4.9}$$

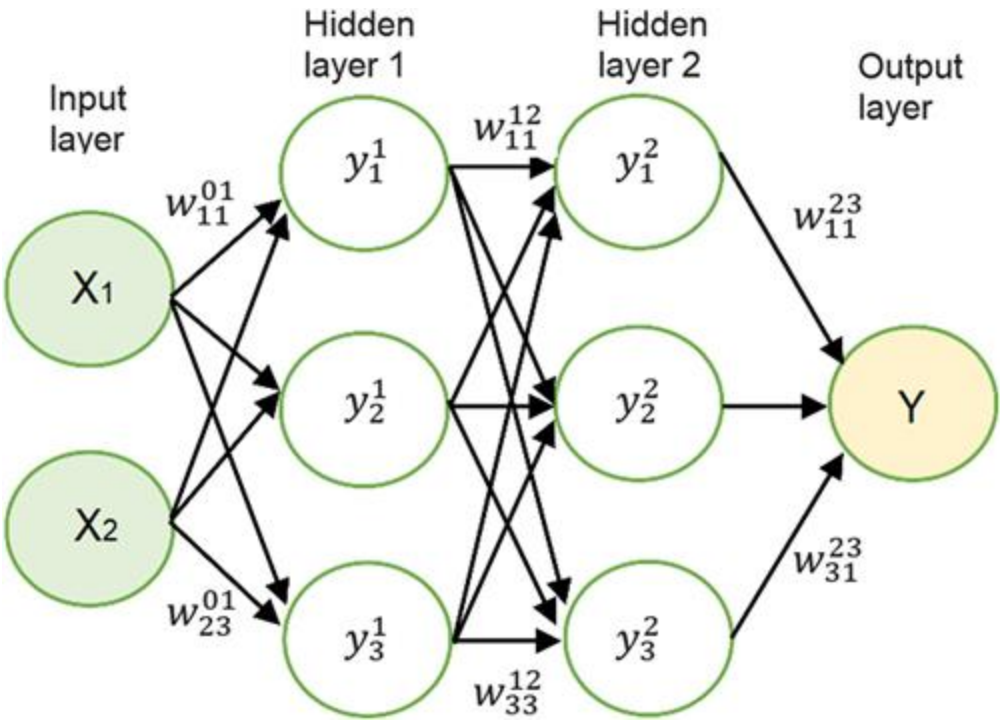


Figure 4.1 A small multi-layer feed-forward NN with an input layer that contains atomic coordinates, X={X_i}, two hidden layers with 3 nodes, *y*, per hidden layer, and an output layer that yields a property, Y. A bias that is connected to each node in the hidden layers and output layer is not illustrated here.

As can be seen from eqn (4.9), the NN model can be referred to as a set of functional transformations. First, the linear combinations of the inputs are generated, $\sum_{i=1}^{2} w_{ij}^{01} X_i + b_j^1$. A non-linear basis function, *f*, is then employed and used to transform this expression to yield the second layer of the network. Frequently used basis functions are sigmoidal functions, tangent hyperbolic, or a shifted softplus function, such as $\ln(0.5e^x+0.5)$, that was used for the results presented here. By a series of such transformations, the NN model can be constructed in its simple multi-layer feed-forward (MLFF) architecture. The bias values, b^j_i, are used to achieve more flexibility by allowing basis functions to shift. In the last step, the desired properties are obtained by using a linear basis function. Hence, if no hidden layers are defined, the MLFF NN is equivalent to LR in its simplest form.[88,89] As the number of hidden layers and nodes per hidden layer

defines the depth of the network, the complexity and the depth of the models are independent of the training set size.

To find the best possible relation between X and Y, the loss function to be minimized is the sum of mean squared errors between NN predictions and reference values, as given in eqn (4.3). Several algorithms exist, with Adam (adaptive moment estimation)[90] being one of the most popular stochastic gradient descent optimization algorithms. To prevent the model from overfitting, different techniques can be applied, such as the inclusion of a L1 and L2 regularization, soft weight sharing, dropout or an early stopping mechanism.[91] The latter can be applied by splitting the whole training set into a training set and a validation set, whereas the validation set is excluded from training, but included for prediction. This means that after each epoch, *i.e.*, when the weights get updated, the error on the validation set is computed and recorded. Once this error increases, the model is assumed to overfit and the training needs to be stopped.[89] The NN results here were obtained using theano.[92]

4.2.1.5 Training Process

Compared with LR, KRR, and SVR, NNs contain a lot more hyperparameters that need to be optimized in order to achieve successful training and to fit the weights to accurately predict the desired property. Hence not only the type of basis function, its width and the regularization terms have to be optimized, as is the case for KRR and SVR, but also the number of nodes per layer, the number of layers and several more hyperparameters. One commonly-used way to search the space of hyperparameters is *via* random grid search.[93] For some implementations and further details, see for example ref. 9 or 50. It is further advantageous to scale inputs and outputs of the training set by subtracting the mean and dividing by the standard deviation. We used such a scaling for our NNs.

As already mentioned, the depth of the NN does not depend on the number of data points, and a deep NN can be built from any training set size. For practical reasons and memory-efficient computations, it is thus favourable to split the training set into the so-called mini batches. The batch size depends on the problem and no rule of thumb on its optimal value can be given. A batch is thus a subset of the training set, which delivers a

combined gradient for the optimization of the weights *via* gradient descent. Each time a batch is used for training, the weight parameters get updated. After one pass through the complete training set, an epoch is finished; for practical reasons, an early-stopping mechanism is recommended to prevent overfitting.[93] After one epoch, the weights can be adapted, and also the learning rate, which defines the magnitude of change of the weights after each epoch and is one of the most critical hyperparameters. A large initial learning rate prevents the NN model from getting stuck at a local minimum. It can then be decreased during the optimization process. The number of hidden layers and nodes per hidden layer determines the depth of the NN. Usually, up to a certain number of hidden layers and nodes per hidden layer, the NN gets more accurate, but converges towards an optimal value in terms of accuracy and computational costs. In addition, the L1 and L2 regularization rates have to be selected and searched accordingly.

Owing to this more complex architecture, NNs are more difficult to apply in practice than LR, KRR, and SVR. However, an NN is more flexible and can fit more complex data. It is also straightforward to relate one molecular input to a multi-dimensional output. In detail, one model can predict the values of each energetic state as well as the corresponding gradients in one step. In this case, the training improves if the forces, F, are included in the loss function, L. Another hyperparameter, η, that controls the influence of forces on the loss function, as described in eqn (4.10), has to be added.

$$L = \frac{1}{N}\sum_{i}^{N}\left(E_i^p - E_i\right)^2 - \frac{\eta}{N}\sum_{i}^{N}\frac{1}{3M}\sum_{\alpha}^{3M}\left(F_i^p - F_i\right)^2 \tag{4.10}$$

The same accounts for NACs, where one model can be built to match an input to all the NAC vectors between different states of the same spin multiplicity. The corresponding models are thus multi-state models for energies and gradients and multi-value models for NACs. LR, KRR, and SVR, are single-state models by definition. To make multi-state predictions possible, an explicit encoding of the energetic state is necessary, *i.e.*, a separate kernel for this state must be defined to predict the energies of all states with one model.

To achieve energy conservation during dynamics simulations, the forces should be treated as derivatives from energy potentials. This can be done for KRR by using atomistic Gaussian kernels in combination with the FCHL representation,[34,84] which will be explained in the following and is implemented in the QML toolkit.[83]

4.2.1.6 Descriptors

To represent a molecule suitably for an ML model, a rotationally and translationally invariant descriptor must be provided that contains all the relevant information regarding a molecular geometry to fit a complex property. Compared to previous ML potentials for the electronic ground state, where a universal model is aimed for, the excited-state potentials are molecule-specific. Hence, it is sufficient to present the molecule as a whole to the ML models. This MLFF NN contrasts to high-dimensional NNs,[94] where each atom type is represented by one NN, such as given in ref. 15, 20, 89, and 95–97 for example to construct ML potentials for the electronic ground state. The target in our approach is to fit complex excited-state properties and aim for long time scale photodynamics simulations. Several types of descriptors are compared: The inverse distance matrix is the simplest descriptor applied, which contains only information about the distances of the atoms. The Coulomb matrix[98] additionally contains atomic charges, and a polynomial descriptor is applied, generated *via* cross-products of all entries of the inverse distance matrix.[50] For KRR, the FCHL representation is applied,[82] whereas LR and SVR are trained to relate the inverse distance matrix to a desired property. In the following, the descriptors will be explained in more detail with the test molecule, methylenimmonium cation, $CH_2NH_2^+$, taken as the reference.

4.2.1.7 Distance-matrix Based Descriptors

The inverse distance matrix, *D*, will be abbreviated as inv.D. in this chapter. Its entries

$$D_{ij} = \frac{1}{|R_i - R_j|} \tag{4.11}$$

with R_i being the position of atom i and R_j the position of atom j are very similar to those of the Coulomb matrix,[98]

$$C_{ij} = \begin{cases} \dfrac{Z_i Z_j}{|R_i - R_j|} & \text{if } \delta_{ij} = 0 \\ \dfrac{1}{2} Z_i^{2.4} & \text{if } \delta_{ij} = 1. \end{cases} \tag{4.12}$$

In the latter, the atomic charges of atoms i and j, Z_i and Z_j, are additionally included. When comparing matrices, the inverse distance matrix contains 15 features for the 6-atom molecule $CH_2NH_2^+$, *i.e.*, the off-diagonal elements of the upper or lower triangular matrix. The diagonal elements of this symmetric matrix are not defined. The Coulomb matrix additionally contains constant values for the diagonal matrix elements resulting in 21 input features as a molecular descriptor for $CH_2NH_2^+$. Considering the aforementioned scaling of inputs, both descriptors are equivalent and should yield comparable results. Therefore, another descriptor is computed, consisting of polynomials of the inverse distances, abbreviated as poly.D. in the following. The dimension of this descriptor is squared compared to inv.D., thus allowing for a more accurate representation of the molecule.

4.2.1.8 FCHL: Faber-Christensen-Huang-Lilienfeld Representation

The FCHL (Faber-Christensen-Huang-Lilienfeld) representation[34,84] has the advantage over distance-matrix based descriptors that it is also permutation invariant. This invariance is achieved by describing the molecule atom-wise in its chemical and structural environment. To this aim, a set of M-body expansions, $A_M(I)$, is defined for each atom, I. Here, only the first three terms will be included, due to minor advances when accounting for higher body terms.[34] Similar to the distance-based descriptors, the FCHL representation contains information about the interatomic distances (second-order expansion, $A_2(I)$), but additionally includes information about the chemical compositions (first-order expansion, $A_1(I)$) and angular

distributions (third-order expansion, $A_3(I)$). The representation up to the third-order expansion reads:

$$A_3(I) = N\left(x^{(1)}\right) \sum_{i \neq I} N\left(x_{iI}^{(2)}\right) \sum_{j \neq i,I} N\left(x_{ijI}^{(3)}\right) \xi_3\left(d_{iI}, d_{jI}, \theta_{ij}^{I}\right) \tag{4.13}$$

with the first-order expansion

$$N\left(x^{(1)}\right) = A_1(I) = e^{-\frac{(P_I - x_1)^2}{2\sigma_P^2} - \frac{(G_I - x_2)^2}{2\sigma_G^2}} \tag{4.14}$$

and $\boldsymbol{x}^{(1)}=\{P_i, \sigma_P; G_i, \sigma_G\}$ accounting for stoichiometry. The corresponding higher terms are similarly generated with $\boldsymbol{x}^{(2)}_{ij}=\{d_{iI}, \sigma_d; P_i, \sigma_P; G_i, \sigma_G\}$ and $\boldsymbol{x}^{(3)}_{ijI}=\{\theta^I_{ij}, \sigma_\theta; P_j, \sigma_P; G_j, \sigma_G\}$ and are similar to the atom-centred symmetry functions for NNs of Behler[99] or their weighted variants.[97,100]

The advantage of this descriptor compared with the previously mentioned distance-based descriptors is that it treats the molecule atom-wise and thus includes more information about the molecule, *i.e.*, the atomic composition and angular distribution. The disadvantage is a higher computational expense, which is optimized in a new version, called FCHL19.[84] The previous version FCHL18, was used to obtain the results presented here.

4.2.2 Training Set Generation

A training set that is applicable for NAMD should contain a molecular geometry and the energies of all energetic states that are included in dynamics simulations, corresponding gradients as well as couplings between the states. Spin–orbit couplings are important between states of different spin-multiplicities, whereas NACs account for transitions between states of same spin-multiplicity. Here, the test system has three active singlet states and thus NACs will be described. Notably, the accuracy of the ML potentials solely relies on the quantum chemistry reference method if ML models are trained appropriately. Hence it is important to choose this method thoughtfully. To keep the number of expensive reference computations minimal, cost-effective generation of the training set is

required.[50] Nevertheless, this small training set should cover the relevant conformational space of a molecule that is visited during dynamics simulations. Therefore, it is advisable to build such a training set by starting with an initial, small set that is generated manually and to expand it in a meaningful way *via* adaptive sampling.[15,20,50,89,101,102]

4.2.2.1 Initial Training Set

As a starting point for generating a training set for NAMD simulations, the equilibrium geometry of a molecule should be included and taken as a reference point for all subsequent calculations to compute wave function overlaps. Hence, the information of the wave function of this geometry should be saved for further pre-processing (see Section 4.2.3). In principle, every sampling method can be used for computing the rest of the data points for the initial training set. For small molecules, which do not have many degrees of freedom, a good guess is to carry out scans along each normal mode coordinate. An even better idea is to take points from the optimization of critical points like excited-state minima, conical intersections, and state crossing points. For larger systems, some other approaches like Wigner sampling[103] or sampling *via* molecular dynamics simulations,[104,105] such as umbrella sampling,[106] trajectory-guided sampling,[107] enhanced sampling[108] or metadynamics,[109] *e.g.* with the semiempirical tight-binding based quantum chemistry method GFN2-xTB[110] can be employed, to name only a few recent developments. In any case, it is advantageous to consider critical regions of the PESs or reaction coordinates that might be visited during photodynamics simulations in advance. For example, if it is known from literature or supposed by chemical intuition that photo-excitation can lead to hydrogen dissociation or carbon–carbon bond cleavage, appropriate reaction coordinates should be scanned. As soon as the initial training set is thought to be large enough (so far, we have used approximately 1000 points), ML models can be trained and dynamics simulations can be carried out using adaptive sampling for excited states,[50] as will be explained below.

4.2.2.2 Adaptive Sampling for Excited States

In general, an ML model used to replace quantum chemistry in NAMD simulations is, simply said, interpolation between data points. Therefore, it can be as accurate as the reference method while being extremely fast at the same time in interpolative regions of the PESs. However, it fails dramatically in extrapolative regions.[78] Hence, it is not sufficient to sample a conformational subspace of a molecule in advance, but it is necessary to check the reliability of ML predictions at any point in time along a dynamics run. If a region is detected to be untrustworthy, it is either undersampled or not included in the training set at all. The conformation, where the untrustworthiness was detected, should be added to the training set and the ML models should be re-trained from this expanded training set in order to continue ML-NAMD simulations accurately and on long time scales. The method of choice to expand the training set with unknown conformations on-the-fly and check the accuracy of ML predictions is in most cases a process termed adaptive sampling,[15,20,89,101,102] which is well-established for ground state dynamics, and modified for excited-state dynamics.[50] The process is exemplified in Figure 4.2.

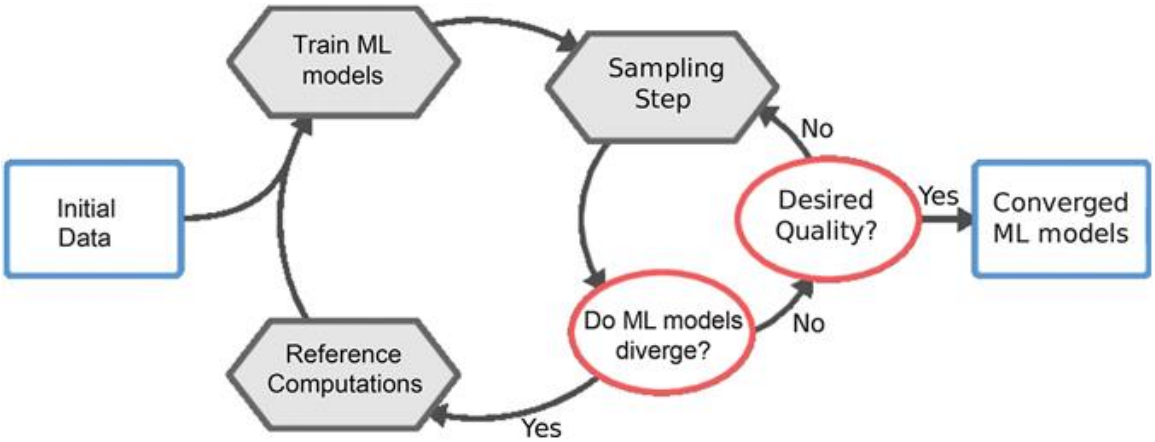


Figure 4.2 Scheme of the adaptive sampling procedure adapted from ref. 32.

In more detail, adaptive sampling for excited states starts with a NAMD simulation, where the excited-state properties, $\tilde{Y}^p_j$, (PESs, forces, and NACs), are taken as the mean, Y^p, of predictions from at least two ML models, J:

$$\bar{Y}^p = \frac{1}{J}\sum_{j=1}^{J} \tilde{Y}_j^p . \tag{4.15}$$

Those ML models are independently trained from the initial training set but differ slightly in their initial hyperparameters or starting weights. At each time step, the RMSE between the ML model forecasts, $Y^p{}_\sigma$, is computed and compared with a predefined threshold, ε^p, which is set for each excited-state property. The thresholds are defined manually and should be in the range of the validation error of the ML models. A separate threshold for each property is used. Whenever one of these thresholds is exceeded, such that

$$Y_\sigma^p = \sqrt{\frac{1}{J-1}\sum_{j=1}^{J}\left(\tilde{Y}_j^p - \bar{Y}^p\right)^2} \geq \varepsilon^p , \tag{4.16}$$

the prediction is termed untrustworthy, and the molecular geometry is presumed to be part of an unexplored conformational space. This geometry is then used as an input to carry out an additional quantum chemistry reference calculation for each property, no matter which property's threshold is exceeded. The training set is expanded with this additional data point and the ML models are retrained from the expanded training set. After one such cycle, the thresholds are adapted by multiplication with a predefined factor (*e.g.* 0.95) to reduce the magnitude of the thresholds and sample the conformational space comprehensively for the training set generation. The dynamics simulations with an ensemble of newly-trained ML models can be continued from the unknown configuration, employing the adaptive sampling procedure until the training set covers the relevant space for NAMD.[20,50,89]

For practical reasons and efficient training set generation, it is favourable to start not one trajectory with an ensemble of ML models, but approximately 100 trajectories, as is usually done within NAMD. The ML models should then be retrained only after each trajectory reaches an unexplored conformational region of the PESs and the corresponding data points are all added to the training set. Hence the training of the ML models

is carried out only once for all trajectories and not for each of the aborted trajectories separately. In this way, the computational cost for the training of the ML models is reduced.

Furthermore, it is advisable to recalculate the thresholds according to the error of ML models on the training set and to restart the dynamics from the beginning when the size of the training set increases by a factor of about two. Further, the hyperparameters of the ML models should be scanned and optimized again, in order to find the best fit for the expanded training set. This procedure is advisable, because the trajectories will probably not lead to the exact same conformational region, but might explore different subspaces of the PESs.[32,50]

In general, there is no guide on how large the training set should be for NAMDs. It is rather a matter of the size and flexibility of the targeted molecule as well as the complexity and diversity of the photo-induced processes that can take place. If different reaction channels are accessible after photo-excitation, far more data points might be needed than for a rigid molecule, where most trajectories lead towards the same reaction channel. Another important factor is the number of active states that are included in NAMD. By checking the convergence behaviour of the sampling procedure, an indication can be obtained on how close the training set is to its optimal size.[50]

For all those reference computations, more states might be considered than in the actual ML-NAMD, and this fact should also be kept in mind when choosing the reference method. This is especially important for excited-state properties that result from wave functions of two different electronic states, such as NACs that couple two different states or transition dipole moments. Their sign, *i.e.*, if the coupling value, C_{ij}, is $+C_{ij}$ or $-C_{ij}$, is dependent on the relative phase of the wave function of each electronic state. For very similar geometries, the couplings can arbitrarily switch their sign, leading to inconsistencies in the potentials formed. This makes it difficult for an ML model to find a relation between C_{ij} values and a molecular geometry. To allow for meaningful learning of off-diagonal elements in terms of matrix notation, a further preprocessing, termed phase correction, is carried out such that almost all such inconsistencies are removed, and a proper learning behaviour of the ML models can be

achieved. To this aim, it is necessary to compute wave function overlaps between every geometry in the training set and the reference geometry defined at the beginning of the training set generation, that still contains all the information about the wave function.[50,111,112] This contrasts with the generation of a training set for Born–Oppenheimer molecular dynamics and thus will be explained in the following.

4.2.3 Wave Function Phases

The problem of phase inconsistencies throughout a set of quantum chemical reference computations can be exemplified with molecular orbitals of the methylenimmonium cation (Figure 4.3 taken from ref. 50). As can be seen in panel A of Figure 4.3, molecular geometries are the input of a quantum chemical reference computation. The interpolation coordinate here is the elongation along the C–N bond. In panel B, the molecular orbitals obtained from each quantum chemical calculation are given for the S_1 and S_2 states. As shown, they can switch their sign, illustrated by their brightness – that is either dark or light. The same accounts for the electronic wave function, which is a valid eigenfunction of the electronic Hamiltonian. If it is multiplied by a phase factor, it is still a valid eigenfunction. Hence, the wave function is not uniquely defined. As a consequence, the coupling vectors between two states, NAC_{ij}, and in general any off-diagonal element in terms of matrix notation, C_{ij}, resulting from the wave functions of those two states, Ψ_i and Ψ_j,

$$C_{ij} = \langle \Psi_i | \hat{O} | \Psi_j \rangle, \tag{4.17}$$

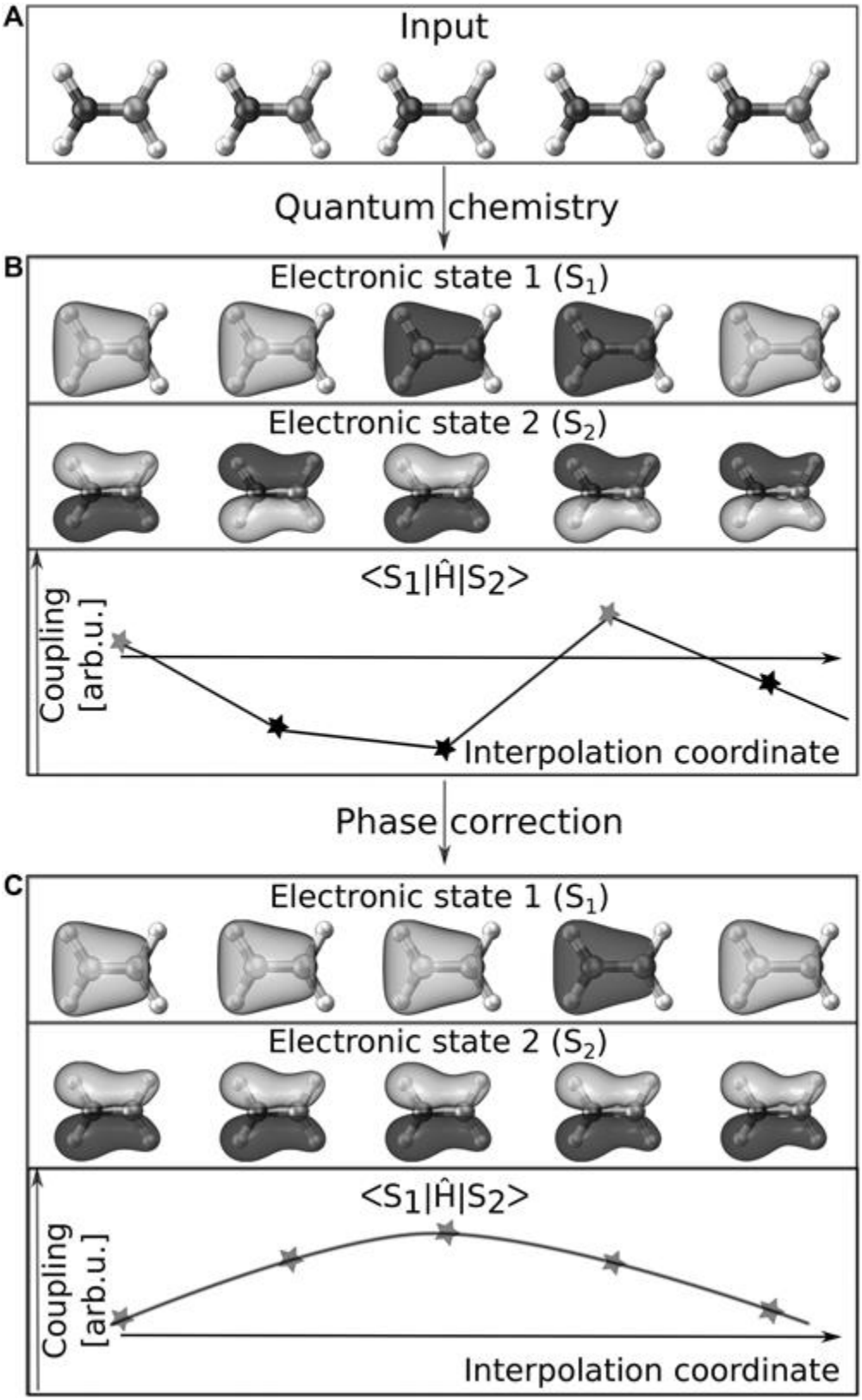

A
Input
Quantum chemistry
B
Electronic state 1 (S_1)
Electronic state 2 (S_2)
$\langle S_1|\hat{H}|S_2\rangle$
Coupling [arb.u.]
Interpolation coordinate
Phase correction
C
Electronic state 1 (S_1)
Electronic state 2 (S_2)
$\langle S_1|\hat{H}|S_2\rangle$
Coupling [arb.u.]
Interpolation coordinate

Figure 4.3 A set of quantum chemical calculations of the methylenimmonium cation, $CH_2NH_2^+$, along the C–N bond. The molecular conformations, which are the input of a quantum chemical calculation, are given in panel A, the orbitals of the computed two electronic states, the S_1 and S_2 state as well as the corresponding off-diagonal matrix elements between those two states, $\langle S_1|\hat{H}|S_2\rangle$, are given in panel B. As can be seen, the sign of those values arbitrarily switches. Those sign jumps can be removed by applying a phase correction algorithm. Results are given for those elements in panel C. Reproduced from ref. 50 with permission from the Royal Society of Chemistry.

might randomly change sign along an interpolation coordinate. As a result, these elements show discontinuous curves, as can be seen in the bottom of panel B. To remove such phase jumps and make correct learning of the ML models possible, a virtual global phase correction has to be applied.[50] This process allows one to correct the couplings so that smooth curves can be obtained, as given in Figure 4.3 in panel C. This phase correction algorithm, discussed in more detail below, is applied only to ensure a global phase in most parts of the configurational space that is covered within the training set in order to make it learnable and applicable to an ML model.

It is worth mentioning that phase jumps resulting from the Berry phase cannot be removed, see ref. 59 for more details. However, it is assumed that those remain only in a small subspace of the training set and are thus negligible. This is further validated by theories, such as the Zhu Nakamura theory,[58,67,80,113–115] that do not track any phase along a NAMD simulation and still yield accurate results.

4.2.4 Phase Correction Algorithm

To carry out phase correction, the phase of each electronic state must be tracked from a pre-defined reference geometry onward. This can be done by computing wave function overlaps,[74,111,112] $\boldsymbol{S}$, between two adjacent molecular geometries, k and l, as implemented for example in the WFoverlap code:[111]

$$\boldsymbol{S} = \langle \Psi_k | \Psi_l \rangle. \tag{4.18}$$

The dimension of this matrix is $N^{\text{states}} \times N^{\text{states}}$, with N^{states} being the number of active states included in the calculations. If the two configurations k and l are very similar, the overlaps are obtained as diagonal

elements with values close to ±1. If the wave function overlap of a state is very close to +1, no phase change occurred, while a value close to −1 indicates that the phase of the wave function switched. A phase vector, $\boldsymbol{p}$, can then be defined that contains a value of +1 or −1 for each electronic state, which indicates the change of the phase with respect to the previous geometry. Apart from critical regions of the PESs, $\boldsymbol{p}$ usually corresponds to the diagonal matrix elements of $\boldsymbol{S}$.

In regions of the PESs where adiabatic potential energy curves have an avoided crossing and NACs become large, the off-diagonal elements of $\boldsymbol{S}$ might be larger than the corresponding diagonal elements. Then, the sign of $\boldsymbol{p}$ for each electronic state is obtained from the largest absolute value of a row of $\boldsymbol{S}$.

During adaptive sampling, there will be cases when the two configurations, *i.e.*, the one that should be included in the training set and the reference geometry, are not similar enough to obtain overlaps larger than an absolute value of 0.5. In such cases, no information about a phase change can be obtained and interpolation between the two geometries in combination with iterative computation of wave function overlaps is necessary. The entries of $\boldsymbol{p}$ can be obtained using the following equation:

$$p_i = sgn\left(\max\left(\left|S_{ij}\right|\right)\right) \ \forall \ \left|S_{ij}\right| \geq 0.5;\ i,j = 1, 2, \ldots, N^{\text{states}}. \tag{4.19}$$

If interpolation is necessary during phase correction, the phase vector of the final geometry, $\boldsymbol{p}_n$, is computed as the product of all previous phase vectors, $\boldsymbol{p}_0$ to $\boldsymbol{p}n-1$:

$$\boldsymbol{p}_n = \prod_{\beta=0}^{n-1} \boldsymbol{p}_\beta. \tag{4.20}$$

If the interpolation steps, β, become very large, it is advisable to store the wave functions of different geometries and the corresponding overlap calculations, *i.e.*, $\boldsymbol{p}$. If a new geometry has to be added to the training set, the root mean square deviations (RMSDs) of this geometry to all other geometries, of which wave function information is available, can be computed and the phases can be tracked back to the geometry with the

lowest RMSD. This intermediate calculation can reduce the number of interpolation steps significantly and hence leads to less additional computational cost.[50]

Special care must be taken when molecules are treated that contain many states lying energetically close to each other. So-called *intruder states* can become a source of error for phase correction. These states are so high in energy at the reference geometry that they are not computed, but possess a significantly lower energy at another geometry during a dynamics simulation such that they should have been included. In such cases, the relevant state for tracking the phase is not part of the active states of the geometry that has to be added to the training set and the result of the wave function overlap computation will not provide any information about a phase change of this state. Therefore, it is favourable to include more energetic states in the calculations from the beginning. In the targeted NAMD simulations, those additional states will not be accounted for and hence the forces and couplings that involve those states do not need to be computed. Therefore, computations are still affordable.

When the phase vectors of the configurations in the training set are calculated, the training set can be pre-processed. This should be done during adaptive sampling on-the-fly. For the initial training set, it can be done at any time before training. Phase correction can be employed using eqn (4.21) for the Hamiltonian matrix, $\boldsymbol{H}$ containing *e.g.* SOCs, and eqn (4.22) for the NAC vectors between a set of states, $\mathbf{NAC}_{ij}$. Each matrix element of $\boldsymbol{H}$ is multiplied with the corresponding phase values of each state. In contrast, all values of a NAC vector between two states, $\mathbf{NAC}_{ij}$, (3 vectors accounting for x, y, and z direction with N^{atoms} entries (N^{atoms} is the number of atoms of the molecule)), is multiplied with the corresponding phase values, p_i and p_j, of those states.

$$H_{ij} = H_{ij} \cdot p_i \cdot p_j;\ 1,\ 2,\ \ldots N^{\text{states}} \tag{4.21}$$

$$\mathbf{NAC}_{ij} = \mathbf{NAC}_{ij} \cdot p_j \cdot p_j \tag{4.22}$$

In contrast to NAMD simulations, ground-state dynamics simulations do not need a pre-processing of the training set and thus also no phase

correction. The reason is that energies and gradients of any electronic state are computed as diagonal elements from the wave function of the same state, $\langle \Psi_i | \hat{O} | \Psi_i \rangle$, where the phase factor enters twice and hence cancels out.

4.2.5 Surface Hopping Dynamics

To carry out NAMD simulations, the Surface Hopping including ARbitrary Couplings (SHARC) method[115] was used for the results represented here. SHARC is an extended version of Tully's fewest switches surface hopping method.[68] In surface hopping methods, the nuclei move on the adiabatic PESs that are made up by the electrons and are computed with quantum chemistry. The PESs are coupled by spin–orbit couplings or NACs, that directly influence the hopping probability of a molecule. A hop is a nonadiabatic transition from one electronic state to another. This event is computed stochastically by comparing a random number between 0 and 1 to the computed hopping probability. As surface hopping is a stochastic approach, an ensemble of trajectories is computed to approximate the motion of a wave packet. The results can be displayed by population plots, showing the population transfer between the electronic states along time.

In order to execute ML surface hopping molecular dynamics, the pySHARC Python wrapper for the SHARC dynamics driver[116] can be used, that, compared with the conventional NAMD program SHARC, omits the file I/O and allows for direct communication of ML models and the driver. This reduces the program runtime of SHARC drastically.[50] Usually, conventional NAMD codes are not limited by the duration of file I/O, as the quantum chemical calculations of the electronic properties are generally the time-limiting step in such simulations. However, when they are replaced by ML models that are much faster than quantum chemistry, the file I/O can become the time-limiting step. Moreover, as long dynamics simulations are aimed for, one has to keep in mind that the amount of produced data increases substantially and that the analysis of the outcomes might take longer than the production runs. In these cases, ML models can also be used to efficiently analyze the data and explore new concepts.[9] Further, it is advisable to save only each 10th–100th time step, when the dynamics do not change much and to save every time step, when transitions and

important processes take place. For example, when a molecule is trapped for a long time in an excited state, it not necessary to save each time step of each trajectory, whereas in critical regions of the PESs, where many different processes can take place on a very short time scale, it is important to store the information of every time step for subsequent analysis.

The training set that is used to produce all the results mentioned in this chapter is taken from ref. 50 and comprises a conformational subspace of the methylenimmonium cation, $CH_2NH_2^+$, that is obtained from normal mode scans and adaptive sampling using the SHARC program and two NNs. It contains three active singlet states and was computed with the multi-reference configuration interaction method that accounts for single and double excitations (MR-CISD). The double-zeta basis set aug-cc-pVDZ was used; in total the training set contains 4000 phase corrected data points, where each data point contains the molecular geometry in *xyz*-format, energies of three singlet states, corresponding gradients, and NACs. Permanent and transition dipole moments are also included but are not treated here. To compare NAMD based on ML with NAMD based on quantum chemistry, reference dynamics based on MR-CISD was carried out after excitation to the second excited singlet state for 100 fs using a time step of 0.5 fs. The so-obtained single points were not part of the training set for the ML models and cannot be used as such, as no phase correction was carried out for these points. For more details on reference dynamics, see ref. 50.

4.3 Example: Methylenimmonium Cation

As already introduced, many processes that are important for nature and life depend on light, such as photosynthesis or the ability to see. Concerning the latter, retinal is the molecule in the protein rhodopsin that plays a key role in vision. The photoreactive part is the chromophore retinal, which belongs to the family of protonated Schiff bases. The smallest member of this family is the methylenimmonium cation, $CH_2NH_2^+$, which undergoes ultrafast switches after photo-excitation. The nonadiabatic transition from the first

excited singlet state back to the ground state leads to a rotation of the molecule along the HC–N–H dihedral angle.[116–118]

In the following, the photo-induced processes of this test molecule are used to investigate the ability of ML models to reproduce NAMD. The ML models are NNs taken from ref. 50. Here, the methylenimmonium cation is used as a test system, because transitions from the second excited singlet state back to the ground state are occurring on an ultrafast time scale, which are challenging to reproduce. More importantly, this ultrafast nature makes the dynamics computationally accessible for simulations based on the quantum chemistry reference method for comparison.[50]

4.3.1 ML Surface Hopping Dynamics

Figure 4.4 shows the population schemes obtained after executing surface hopping dynamics with different methods. Panel A shows results from two slightly different quantum chemistry methods. QC1 refers to the reference quantum chemistry method MR-CISD(6,4)/aug-cc-pVDZ (dashed lines), while QC2 indicates the same method, but with another double-zeta bases set, MR-CISD(6,4)/631++G** (dotted lines). The slightly different quantum chemistry methods are used to demonstrate which population schemes can be regarded as similar. As can be seen from panel A, the two quantum chemistry methods QC1 and QC2 result in very similar population schemes, which are obtained from dynamics simulations of 90 trajectories propagated for 100 fs. Initially, 100% of the population is in the second excited singlet state, S_2, from which fast population transfer to the first excited singlet state, S_1, and back to the ground state, S_0, takes place. This dynamics can be successfully reproduced with NNs trained on the training set of 4000 data points computed with the QC1 method. The results are directly compared with each other in panel B. As can be seen, the population curves obtained from NN computations (continuous lines) are much smoother than those obtained with QC1 (dashed lines). This is because not only 90 trajectories (resulting from 1000 initial conditions from Wigner sampling) were propagated with NNs, but 3846 trajectories (resulting from 20.000 initial conditions from Wigner sampling). This enlarged statistics is the first advantage of NNs and is possible due to the NNs' low computational costs. The overall computational time of 3846

trajectories with NNs (≈19 000 min) is still about a factor of 800 lower than the overall computational time of 90 trajectories obtained from QC1 calculations (≈15.5 Mio. min.) when using the same central processing unit. Comparing panels A and B further shows that the agreement of NNs and QC1 is akin to the agreement of QC1 and QC2.

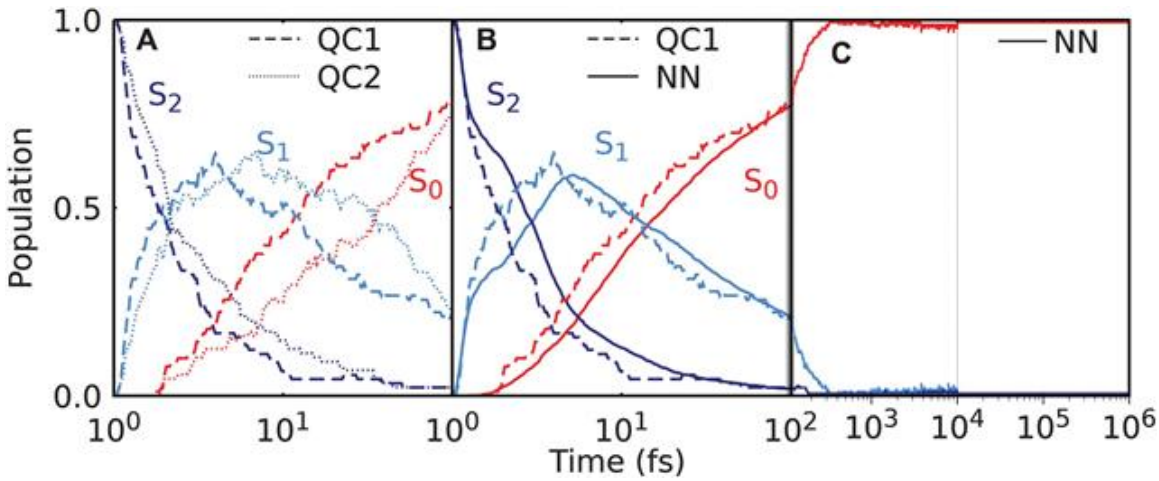


Figure 4.4 Surface hopping dynamics of $CH_2NH_2^+$ after excitation to the S_2 state. (A) Dynamics of 90 trajectories propagated for 100 fs with the QC1 and QC2 method are compared with (B) NNs, which are obtained from 3846 trajectories. (C) Long excited-state dynamics from NNs were obtained from 200 trajectories propagated for 10 ps and 2 trajectories propagated for 1 ns.[50]

The goal of executing long NAMD simulations that are not feasible with the QC1 method anymore is possible with two NNs by employing the aforementioned sampling scheme to ensure accurate potentials for energies, forces, and NACs along the dynamics run. In contrast to the adaptive sampling scheme described above, an iterative sampling scheme is employed, *i.e.*, the threshold for comparison of the different NNs is not reduced anymore, but is fixed in the beginning and kept constant. An ensemble of 200 trajectories was used to produce the population schemes up to 10 ps (10^4 fs) and 2 trajectories were further propagated for 1 ns (10^6 fs). As can be seen, the population is transferred to the electronic ground state after around 300 fs and stays there for the rest of the simulation time. Hops that still take place, can be regarded as noise and are due to remaining kinetic energy. In 6 hours, a dynamics simulation of 10 ps could be

achieved, while it took around 59 days to compute the ns simulation. A simulation for 1 ns would have taken approximately 19 years with the QC1 method.[50]

To make long time scale NAMD simulations possible, it is favourable to include important processes, such as hydrogen atom dissociation or rotations of different molecular groups, into the training set from the beginning. This can be done to reduce the computational costs from additional quantum chemistry calculations, which become necessary if such previously-mentioned events take place during the long dynamics simulations. In the case of hydrogen dissociation, for example, the trajectory might be aborted after each time step the corresponding H-bond gets elongated and no profit would be gained from using NNs instead of quantum chemistry. Indeed, the dynamics would even take longer than the reference dynamics in such cases, as the NNs have to be re-trained after each time step and the phase correction procedure might need additional interpolation steps to properly track the phases of the wave function of each state.

4.3.2 Energy Conservation

Importantly, forces need to be computed as derivatives of NN potentials to preserve energy conservation. To check the energy conservation along NN dynamics, the mean and standard deviation of the total energy along the trajectories is given in Table 4.1 as taken from ref. 50. As can be seen, the total energy is as well conserved with QC1 as with NNs. It is evident that energy conservation becomes slightly worse when hops are forbidden and when trajectories have to stay in the S_2 state for the whole simulation time. This is demonstrated by the larger standard deviation – a trend that is obtained with both methods separately. By reducing the time step of classical nuclei propagation from 0.5 fs to 0.05 fs, the standard deviation lowers considerably, even though hops are forbidden.[50]

Table 4.1 Computation of the MAE and standard deviation (Std.) of the total energy averaged over 90 trajectories (time step for classical propagation of 0.5 fs) and 50 trajectories (time step

for classical propagation of 0.05 fs) propagated for 100 fs with the QC1 method and NNs. Values taken from ref. 50.

Method	Time step [fs]	MAE [eV]	Std. [eV]	Hops allowed?
QC1	0.5	10.63	0.047	Yes
QC1	0.5	10.77	0.059	No
QC1	0.05	10.73	0.011	No
NN	0.5	10.72	0.052	Yes
NN	0.5	10.73	0.061	No
NN	0.05	10.80	0.017	No

4.3.3 Further Tools of ML Models

As shown in the previous section, the NNs can be trained to reproduce excited-state energies, forces, and NACs accurately. Hence, not only NAMD simulations can be carried out, but also optimizations of critical regions, such as state minima or conical intersections. The latter in particular are the target of many quantum chemical studies, and therefore, will also be discussed here.

Conical intersections are places of the PESs, where most of the transitions from one state to another are assumed to take place. The specific geometries at which the hops take place in a surface hopping molecular dynamics simulation are called hopping geometries. At the conical intersections of two coupled states, the NACs show a singularity and are very large. Further, the first derivative of such an avoided state crossing is discontinuous. Therefore, conical intersections pose a challenge to quantum chemistry as well as to ML models. Optimization runs from hopping geometries as starting guesses can be done to find such critical points and to additionally provide a measure of the quality of NN potentials for energies, gradients, and NACs. Figure 4.5, taken from ref. 50, illustrates that NN potentials are accurate enough to find the correct minimum energy conical intersections. It shows the scatter plots of the optimized geometries and the hopping geometries of QC1, QC2, and NNs and further gives the distribution of the training set geometries along two coordinates that are important for each minimum energy conical intersection. Panel A gives the plot of the S_1/S_0 state crossing, which is characterized by a rotation of the

molecule along its dihedral angle (see Figure 4.6 first row), and panel B shows the scatter plot of the S_2/S_1 state crossing that leads to an elongated C–N bond in combination with a slight bipyramidalization of the molecule (see Figure 4.6 second row).

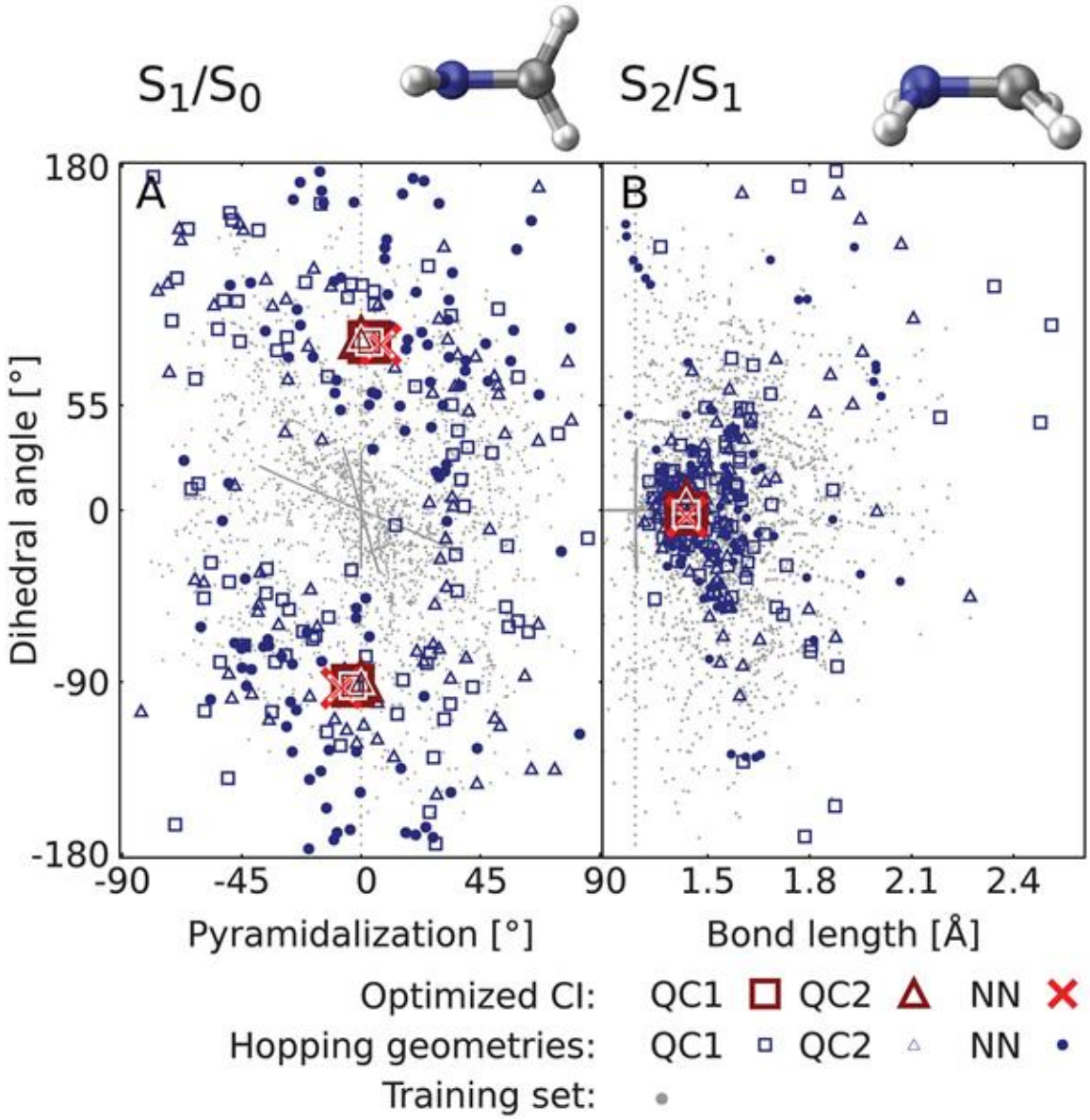


Figure 4.5 The distribution of the hopping geometries around the minimum energy conical intersections obtained with QC1, QC2, and NNs along with the geometries in the training set are shown in panel A and B for the S_1/S_2 conical intersection and the S_2/S_1 conical intersection, respectively. The geometry of the minimum energy conical

intersection of QC1 is depicted on top. The definition of the dihedral and pyramidalization angle used to produce the scatter plots is given in panel C. Reproduced from ref. 50 with permission from the Royal Society of Chemistry.

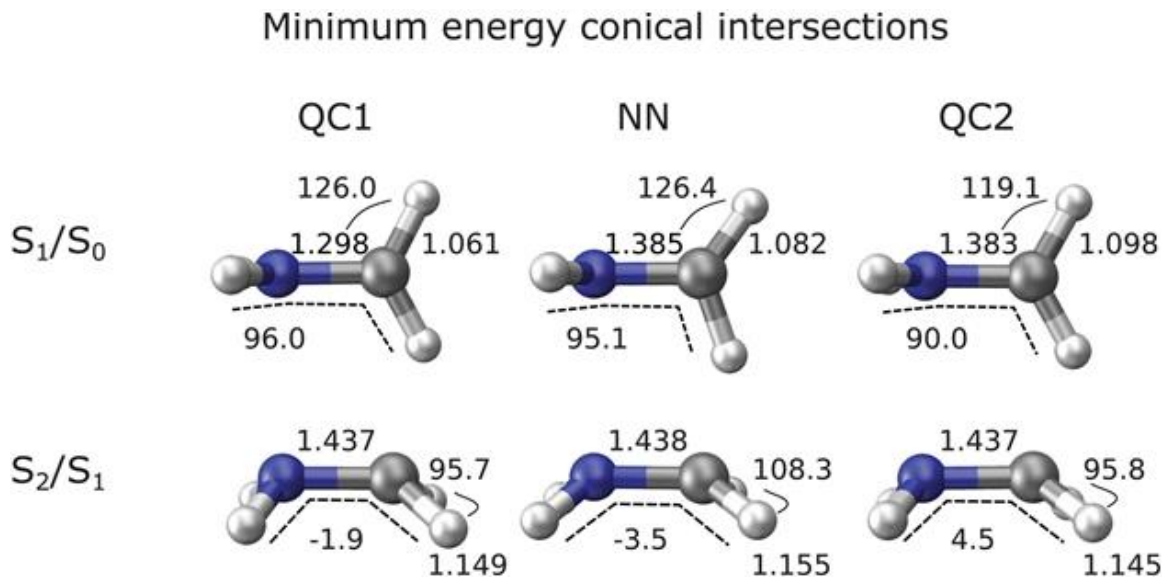


Figure 4.6 For each method, we could find two different conical intersections. QC1 (MR-CISD/aug-cc-pVDZ) is used as the reference method to train neural networks (NNs) and compared with conical intersections obtained with QC2 (MR-CISD/6-31++G**). For the two conical intersections, the bond length between the nitrogen (dark grey) and the carbon (light grey) atom is shown, as well as the bond length between the carbon and one hydrogen atom. Values are given in Angström. A dihedral angle between four atoms marked with the dashed line is given, as well as an angle between the carbon and a hydrogen atom (S_1/S_0 CI) and between two hydrogen atoms (S_2/S_1 CI). Reproduced from ref. 50 with permission from the Royal Society of Chemistry.

As can be seen in the scatter plots, the minimum energy conical intersections found with each method are located very close to each other, which can be also verified in detail from the geometries given in Figure 4.6. In the S_1/S_0 conical intersection (panel A), the hopping geometries are more widely distributed around the minimum energy conical intersection than in the S_2/S_1 state crossing (panel B). This indicates that the potential around the former state crossing is smoother, allowing hops from configurations that are further away from the minimum energy conical intersection. In the latter case, the PESs have a steeper slope and the hopping geometries are more similar to the minimum energy conical intersection. This trend is

obtained with each method separately and demonstrates the high accuracy of the ML potentials for energies, gradients, and NACs.

This assumption can be further tested by potential energy scans around each conical intersection. The scans were independently performed with the QC1 and NN method and the corresponding three-dimensional plots are shown in Figure 4.7. As can be seen, the NN potentials are slightly smoother than the QC1 potentials (compare panels B, A for the S_2/S_1 state crossing and panels D, C for the S_1/S_0 state crossing) and show larger energy gaps between the two approaching surfaces. These larger gaps result from the fact that the ML potentials need to be differentiable at any point in space. Analysis of the conducted potential energy scans around the minimum energy conical intersections further proves the shallower PESs around the S_1/S_0 conical intersection compared with the S_2/S_1 conical intersection and hence explains the wider distribution of hopping geometries around the S_1/S_0 state crossing. Because hops occur further away from the minimum energy conical intersection, the training set (obtained *via* adaptive sampling) does not cover the S_1/S_0 conical intersection as densely as the S_2/S_1 conical intersection. The error of the NNs to reproduce the former conical intersection is thus also larger.[50]

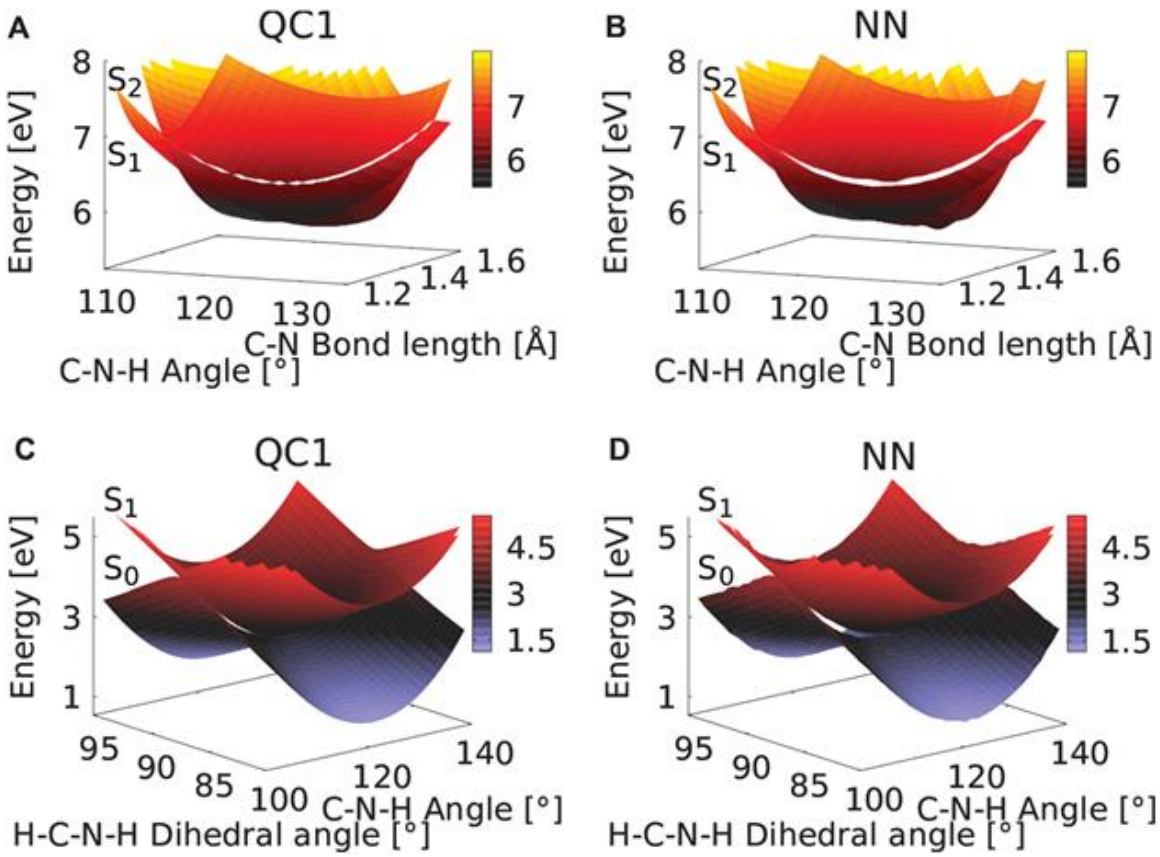


Figure 4.7 The minimum energy conical intersections computed with the QC1 method were used as a centre to compute potential energy scans with the QC1 and NN method around them. The scans around the S_2/S_1 state crossing are given in panel A and B for QC1 and NN, respectively, while the scans around the S_1/S_0 minimum energy conical intersection are given in panel C and D for QC1 and NN, respectively. Reproduced from ref. 50 with permission from the Royal Society of Chemistry.

In addition to the scans around the minimum energy conical intersections, a scan was carried out along a reaction coordinate that contains both of the state crossings. The obtained potential energy curves are computed with QC1, NNs with different representations, LR, SVR, and KRR and are given in Figure 4.8.

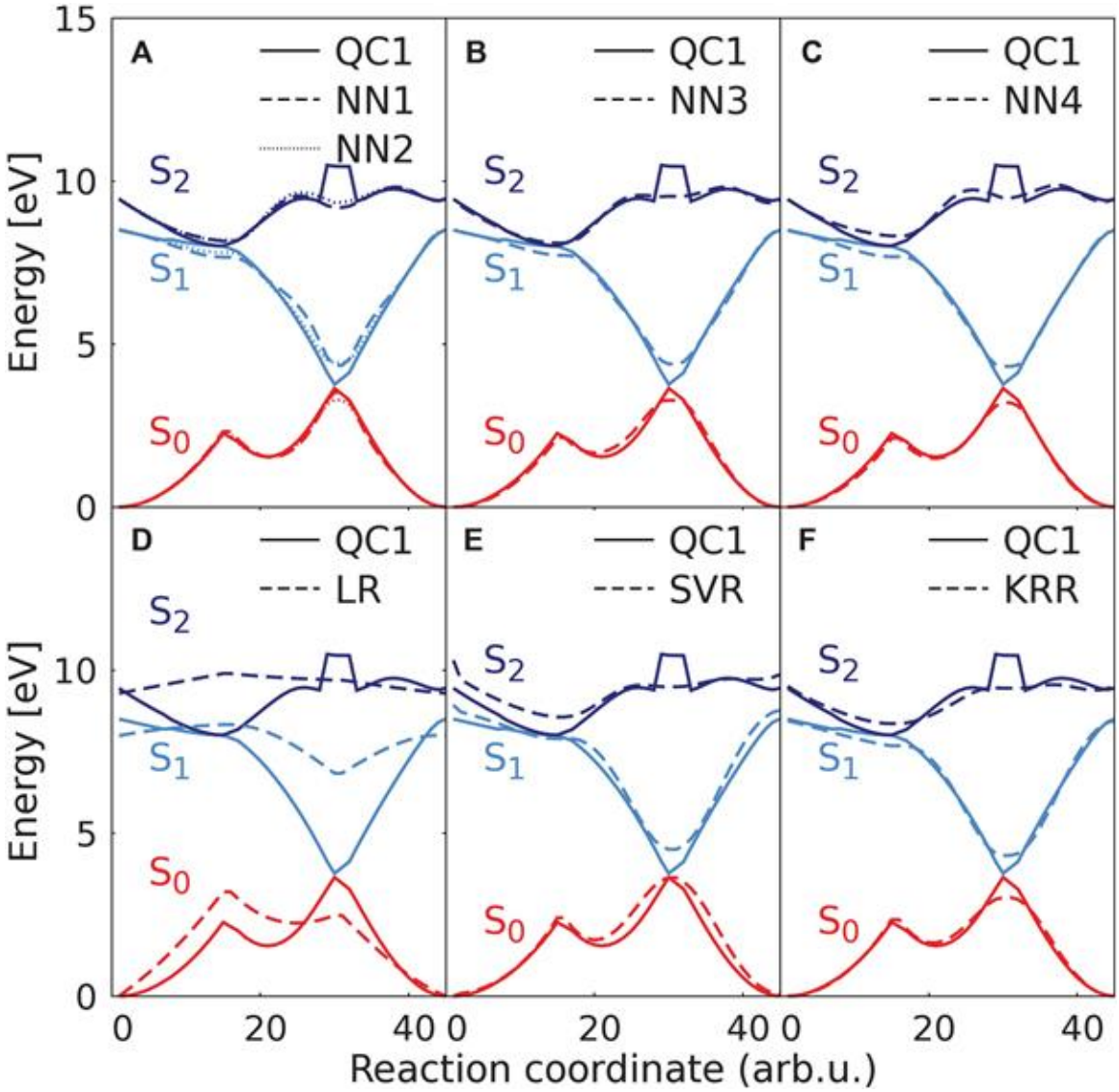


Figure 4.8 Potential energy curves along a reaction coordinate that contains two avoided crossings. One avoided crossing is between the S_2 and S_1 states, and the other one is between the S_1 and S_0 states. The potential energy curves are computed with the QC1 method (continuous line in each panel) and are compared with different ML models (dashed and dotted lines). (A) NN1 and NN2 indicate predictions with the inverse distance matrix using slightly different network parameters, such as it is done in adaptive sampling. (B) NN3 gives results from NNs using the Coulomb matrix as a descriptor and (C) the poly.D. matrix as a descriptor. (E) Curves obtained with LR, (F) SVR, and the inv.D. matrix, and (G) KRR using the FCHL representation that treats gradients as derivatives and provide the most sophisticated representation used here.[34,50]

The NNs employed for the dynamics shown above (NN1/NN2 with inv.D.) are compared with additional NN models with advanced descriptors and different types of regressors. Panel B and C show NN potential energy curves with the Coulomb matrix (NN3) and the poly.D. matrix (NN4) as a representation, respectively. These lead to similarly accurate results and are comparable to the results obtained with the inv.D. descriptor in panel A. The two avoided crossings between different states are not captured as accurately as the rest of the potentials with each method and larger energy gaps are obtained in these regions compared with the QC1 method.

However, a comparison of the NN results with the result of the baseline model, LR, clearly indicates the higher prediction accuracy of NNs. LR is not able to reproduce the shape of the S_2 state at all, and the avoided crossings show a very large energy gap. SVR in panel E is more accurate than LR, but less accurate than NNs with the same descriptor (inv.D. – compare to panel A). KRR given in panel F yields comparable results to NNs. Besides the distinct regressor, the kernel methods, SVR and KRR, differ in the used representation. As the FCHL[82,84] representation could be used for KRR compared with the inv.D. matrix for SVR, more information about the molecule is encoded for KRR, which finally leads to more accurate potential energy curves of KRR. The kernel size depends on the number of data points, therefore a better descriptor can lead to higher prediction accuracy. Nonetheless, the most accurate results are obtained with the NN models. There, no significant improvement can be obtained by using a better descriptor, because the NNs can compensate deficiencies of the descriptor by a deeper network architecture.

As Figure 4.8 indicates, the potential energy curve of the S_2 state shows an artificial jump in the energy that is located near the avoided crossing between the S_1 and S_0 state. This erroneous discontinuity in the quantum chemical potential energy curve is not reproduced by the ML models. This ability of ML models to provide smooth interpolation between data points is even able to correct errors in this case. Moreover, as is clear in panel A, the two slightly different NNs with the inv.D. matrix as a descriptor for molecules (NN1 and NN2) yield very similar potential energy curves along this reaction coordinate. This means that when NN1 and NN2 are used for adaptive sampling, no quantum chemistry reference calculation is required

in such a region of larger artificial error of the NN models with respect to the reference method. The trend of larger error for the S_2 state is also revealed by computing the state-specific MAE on a separate test set with 770 data points obtained from scans along the linear combinations of normal modes (see Table 4.2). The MAE for the S_2 state is approximately doubled for energies and gradients compared with the MAE obtained for the S_0 state. The MAE of the S_1 state is close to the MAE of the S_0 state for energies and lies between the MAE of the S_0 and S_2 states for gradients.

Table 4.2 The MAE of each electronic state separately on a test set containing 770 data points computed with NN1 and NN2 from Figure 4.8 with the inv.D. matrix as a descriptor for molecules. MAEs are taken from ref. 50.

State	MAE energy [H]	MAE gradients [H/Bohr]
S_0	0.00176	0.00444
S_1	0.00200	0.00670
S_2	0.00335	0.00893

For the sake of completeness, the MAEs obtained with each ML model on the same test set of 770 data points on energies, gradients, and NACs are given in Table 4.3. Referring to the NN models, the Coulomb matrix and the inv.D. matrix have similar MAEs for each property, which is expected due to the similarity of the descriptors. It can, however, be observed that the poly.D. matrix yields a slightly lower error and thus higher accuracy. The input vector, *i.e.*, the number of input features to represent a molecule, is squared for the poly.D. matrix compared with the inv.D. matrix. As expected, the LR model yields the largest MAEs for each property. The same trend as it is assumed from potential energy curves in Figure 4.8 can be obtained for the SVR model, which achieves higher accuracy than the LR model and lower accuracy than the NN models and KRR.

Table 4.3 A test set of 770 data points of the methylenimmonium cation was used to compute the MAE on energies [H], gradients [H/Bohr], and NACs [a.u.] averaged over all states for LR,

SVR, KRR, and NNs with the FCHL, inv.D., poly.D. and the Coulomb matrix as molecular descriptors. Most values are taken from ref. 50.

Model	MAE energy	MAE gradients	MAE NACs
NN/inv.D. (NN1/NN2)	0.00237	0.00669	0.328
NN/Coulomb (NN3)	0.00238	0.00690	0.314
NN/poly.D. (NN4)	0.00197	0.00617	0.335
LR/inv.D.	0.09240	0.13902	0.471
SVR/inv.D.	0.00618	0.01169	0.382
KRR/FCHL	0.00300	0.00866	0.351

The MAEs obtained from KRR are slightly larger than those obtained from NNs but smaller than those from SVR and LR. This trend is again in agreement with the accuracy of the potential energy curves given in Figure 4.8.

For all properties, the LR baseline model yields the least accurate results. Using the inv.D. matrix as a descriptor is sufficient for NNs, but it leads to larger errors on energies, gradients, and NACs using SVR. In fact, the MAE on energies is three times as large as it is for NNs, the MAE on gradients is doubled compared with the MAE on gradients obtained with NNs, and the MAE on NACs is also larger. In case of KRR in combination with the FCHL representation, a significant improvement compared with SVR can be obtained. Still, KRR is slightly worse than NNs, which might be because NNs can be regarded as a *multi-state* model and one molecular structure can be related to all energy levels at once. In contrast, KRR can only relate one molecular geometry to one output value. This gives the NN models more information about the relationship between a geometry and its excited-state properties.

4.4 Conclusion and Outlook

ML can be successfully applied to compute excited-state energies, gradients, and NACs. With these properties at hand, much longer time scales than with quantum chemistry-based dynamics simulations are

reachable, due to the low computational costs of ML models and the use of at least two ML models on-the-fly to ensure accurate ML predictions. Further, not only long dynamics simulations can be carried out, but also many more trajectories can be computed than it is usually the case with quantum chemistry methods. Hence, better statistics can be obtained, and rare reaction channels can be analyzed.

To conserve the energies along ML trajectories, the gradients should be derived from ML PESs. As a consequence, the ML PESs are smoother than those obtained with quantum chemistry, and the discontinuities occurring at crossing points of the PESs can never be reproduced completely. This drawback, in turn, becomes an advantage, when erroneous energy jumps arise in the quantum chemistry training data due to the shortcomings of the chosen underlying *ab initio* method. Such wrong discontinuities are then corrected automatically in the smooth ML PESs.

By comparing different ML models and representations of molecules, it can be shown that it is advisable to treat all energy levels that are included in NAMD in one ML model. NNs are employed in this way, and we find that NN potentials are more accurate than the potentials obtained from LR, SVR, or KRR, which can relate the molecular input only to one energy level, when employed straightforwardly. By using a more intricate representation of the molecule, the accuracy of KRR can catch up with that of NNs, while using a better representation for NNs does not improve their performance significantly.

If only energies and gradients are necessary, as is the case for Born–Oppenheimer dynamics, a very efficient training set generation can be achieved from a small initial training set that can include any configurations obtained from manually conducted scans or a sampling method. To enlarge the training set in an efficient way, it is beneficial to start with an adaptive sampling approach as soon as the initial training set is large enough (≈1000 data points) to enable molecular dynamics simulations with preliminary, crude ML models. In this way, the relevant conformational space for photodynamics can be sampled. However, if transition dipole moments or couplings between different states are required, it is important to apply a phase correction procedure from one reference geometry to every data point that needs to be added to the training set.

To conclude, using the automatic sampling technique in combination with the phase correction algorithm to build up and expand a training set for NAMD, new possibilities arise to study the excited-state dynamics of complex chemical systems at nanosecond time scales. With the high accuracy of expensive quantum chemistry methods and the low computational costs of ML models at the same time, this setup can be used to calculate NAMD that are not feasible with conventional *ab initio* methods. Thus, ML-based dynamics simulations allow exploration of new reaction channels and pathways of molecules that are relevant in several research fields, such as medicine, biology, chemistry, and materials design.

References

1. T. Schultz, E. Samoylova, W. Radloff, I. V. Hertel, A. L. Sobolewski and W. Domcke, *Science*, 2004, **306**(5702), 1765.
2. W. J. Schreier, T. E. Schrader, F. O. Koller, P. Gilch, C. E. Crespo Hernández, V. N. Swaminathan, T. Charell, W. Zinth and B. Kohler, *Science*, 2007, **315**, 625.
3. C. Rauer, J. J. Nogueira, P. Marquetand and L. González, *J. Am. Chem. Soc.*, 2016, **138**, 15911.
4. S. Mathew, A. Yella, P. Gao, R. Humphry-Baker, B. F. E. Curchod, N. Ashari-Astani, I. Tavernelli, U. Rothlisberger, M. K. Nazeeruddin and M. Gräatzel, *Nat. Chem.*, 2014, **6**, 242.
5. A. P. Bart´ok, S. De, C. Poelking, N. Bernstein, J. R. Kermode, G. Cs ´anyi and M. Ceriotti, *Sci. Adv.*, 2017, **3**(12), e1701816.
6. I. Ahmad, S. Ahmed, Z. Anwar, M. A. Sheraz and M. Sikorski, *Int. J. Photoenergy*, 2016, **2016**, 1.
7. S. Mai, P. Marquetand and L. González, *J. Phys. Chem. Lett.*, 2016, **7**, 1978.
8. S. Mai, M. Richter, P. Marquetand and L. González, *Chem. Phys.*, 2017, **482**, 9.
9. F. Häse, I. F. Galván, A. Aspuru-Guzik, R. Lindh and M. Vacher, *Chem. Sci.*, 2019, **10**, 2298.
10. B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan and M. Karplus, *J. Comput. Chem.*, 1983, **4**, 187.

11. A. D. Mackerell, *J. Comput. Chem.*, 2004, **25**, 1584.
12. R. Salomon-Ferrer, D. A. Case and R. C. Walker, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2013, **3**(2), 198.
13. M. Christen, P. H. Hünenberger, D. Bakowies, R. Baron, R. Bürgi, D. P. Geerke, T. N. Heinz, M. A. Kastenholz, V. Kräutler, C. Oostenbrink, C. Peter, D. Trzesniak and W. F. van Gunsteren, *J. Comput. Chem.*, 2005, **26**(16), 1719.
14. A. P. Bartók, M. C. Payne, R. Kondor and G. Csányi, *Phys. Rev. Lett.*, 2010, **104**, 136403.
15. Z. Li, J. R. Kermode and A. De Vita, *Phys. Rev. Lett.*, 2015, **114**, 096405.
16. M. Gastegger and P. Marquetand, *J. Chem. Theory Comput.*, 2015, **11**(5), 2187.
17. M. Rupp, R. Ramakrishnan and O. A. von Lilienfeld, *J. Phys. Chem. Lett.*, 2015, **6**(16), 3309.
18. J. Behler, *J. Chem. Phys.*, 2016, **145**(17), 170901.
19. M. Gastegger, C. Kauffmann, J. Behler and P. Marquetand, *J. Chem. Phys.*, 2016, **144**(9), 194110.
20. M. Gastegger, J. Behler and P. Marquetand, *Chem. Sci.*, 2017, **8**, 6924.
21. V. L. Deringer and G. Csányi, *Phys. Rev. B*, 2017, **95**, 094203.
22. V. Botu, R. Batra, J. Chapman and R. Ramprasad, *J. Phys. Chem. C*, 2017, **121**(1), 511.
23. J. S. Smith, O. Isayev and A. E. Roitberg, *Chem. Sci.*, 2017, **8**, 3192.
24. J. Behler, *Angew. Chem., Int. Ed.*, 2017, **56**(42), 12828.
25. H. Zong, G. Pilania, X. Ding, G. J. Ackland and T. Lookman, *Npj Comput. Mater.*, 2018, **4**, 48.
26. A. P. Bartók, J. Kermode, N. Bernstein and G. Csányi, *Phys. Rev. X*, 2018, **8**, 041048.
27. R. Xia and S. Kais, *Nat. Commun.*, 2018, **9**, 4195.
28. S. Chmiela, H. E. Sauceda, K.-R. Müller and A. Tkatchenko, *Nat. Commun.*, 2018, **9**, 3887.
29. G. Imbalzano, A. Anelli, D. Giofré, S. Klees, J. Behler and M. Ceriotti, *J. Chem. Phys.*, 2018, **148**(24), 241730.
30. L. Zhang, J. Han, H. Wang, W. A. Saidi, R. Car and E. Weinan, in *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, USA, 2018, p. 4441.

31. L. Zhang, J. Han, H. Wang, R. Car and E. Weinan, *Phys. Rev. Lett.*, 2018, **120**, 143001.
32. M. Gastegger and P. Marquetand, *arXiv e-prints*, 2018, arXiv:1812.07676.
33. H. Chan, B. Narayanan, M. J. Cherukara, F. G. Sen, K. Sasikumar, S. K. Gray, M. K. Y. Chan and S. K. R. S. Sankaranarayanan, *J. Phys. Chem. C*, 2019, **123**(12), 6941.
34. A. S. Christensen, F. A. Faber and O. A. von Lilienfeld, *J. Chem. Phys.*, 2019, **150**(6), 064105.
35. H. Wang and W. Yang, *J. Chem. Theory Comput.*, 2019, **15**(2), 1409.
36. S. Chmiela, H. E. Sauceda, I. Poltavsky, K.-R. Müller and A. Tkatchenko, *Comput. Phys. Commun.*, 2019, **240**, 38.
37. G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. VogtMaranto and L. Zdeborová, *Rev. Mod. Phys.*, 2019, **91**, 045002.
38. R. V. Krems, *Phys. Chem. Chem. Phys.*, 2019, **21**, 13392.
39. V. L. Deringer, M. A. Caro and G. Csányi, *Adv. Mater.*, 2019, 1902765.
40. J. Behler, K. Reuter and M. Scheffler, *Phys. Rev. B*, 2008, **77**, 115421.
41. C. Carbogno, J. Behler, K. Reuter and A. Groß, *Phys. Rev. B*, 2010, **81**, 035410.
42. F. Häse, S. Valleau, E. Pyzer-Knapp and A. Aspuru-Guzik, *Chem. Sci.*, 2016, **7**, 5139.
43. F. Liu, L. Du, D. Zhang and J. Gao, *Sci. Rep.*, 2017, **7**(8737), 1.
44. D. Hu, Y. Xie, X. Li, L. Li and Z. Lan, *J. Phys. Chem. Lett.*, 2018, **9**(11), 2725.
45. P. O. Dral, M. Barbatti and W. Thiel, *J. Phys. Chem. Lett.*, 2018, **9**, 5660.
46. W.-K. Chen, X.-Y. Liu, W.-H. Fang, P. O. Dral and G. Cui, *J. Phys. Chem. Lett.*, 2018, **9**(23), 6702.
47. D. M. G. Williams and W. Eisfeld, *J. Chem. Phys.*, 2018, **149**(20), 204106.
48. C. Xie, X. Zhu, D. R. Yarkony and H. Guo, *J. Chem. Phys.*, 2018, **149**(14), 144107.
49. Y. Guan, D. H. Zhang, H. Guo and D. R. Yarkony, *Phys. Chem. Chem. Phys.*, 2019, **21**, 14205.
50. J. Westermayr, M. Gastegger, M. F. S. J. Menger, S. Mai, L. González and P. Marquetand, *Chem. Sci.*, 2019, **10**, 8100.

51. N. L. Doltsinis, P. R. L. Markwick, H. Nieber and H. Langer, in *Radiation Induced Molecular Phenomena in Nucleic Acids*, ed. M. K. Shukla and J. Leszczynski, Springer Netherlands, 2008, p. 265.
52. E. Tapavicza, G. D. Bellchambers, J. C. Vincent and F. Furche, *Phys. Chem. Chem. Phys.*, 2013, **15**, 18336.
53. P. Marquetand, J. J. Nogueira, S. Mai, F. Plasser and L. González, *Molecules*, 2017, **22**(1), 49.
54. A. W. Jasper, B. K. Kendrick, C. A. Mead and D. G. Truhlar, in *Modern Trends in Chemical Reaction Dynamics*, ed. X. Yang and K. Liu, World Scientific, **vol. 14**, 2004, p. 329.
55. D. R. Yarkony, *J. Chem. Phys.*, 2005, **123**(20), 204101.
56. X. Zhu and D. R. Yarkony, *J. Chem. Phys.*, 2016, **144**(4), 044104.
57. N. Wittenbrink, F. Venghaus, D. Williams and W. Eisfeld, *J. Chem. Phys.*, 2016, **145**(8), 184108.
58. T. Ishida, S. Nanbu and H. Nakamura, *Int. Rev. Phys. Chem.*, 2017, **36**(2), 229.
59. I. G. Ryabinkin, L. Joubert-Doriol and A. F. Izmaylov, *Acc. Chem. Res.*, 2017, **50**(7), 1785.
60. S. Gómez, M. Heindl, A. Szabadi and L. González, *J. Phys. Chem. A*, 2019, **123**(38), 8321.
61. H. Köppel, W. Domcke and L. S. Cederbaum, in *Conical Intersections*, ed. W. Domcke, D. R. Yarkony and H. Köppel, World Scientific, **vol. 15**, 2004, p. 323.
62. D. R. Yarkony, in *Conical Intersections*, ed. W. Domcke, D. R. Yarkony and H. Köppel, World Scientific, **vol. 15**, 2004, p. 41.
63. G. A. Worth and L. S. Cederbaum, *Annu. Rev. Phys. Chem.*, 2004, **55**, 127.
64. F. Plasser, S. Gómez, M. F. S. J. Menger, S. Mai and L. González, *Phys. Chem. Chem. Phys.*, 2019, **21**, 57.
65. M. Richter, P. Marquetand, J. González-Vázquez, I. Sola and L. González, *J. Chem. Theory Comput.*, 2011, **7**(5), 1253.
66. E. Fabiano, T. Keal and W. Thiel, *Chem. Phys.*, 2008, **349**(1–3), 334.
67. P. Oloyede, G. Mil'nikov and H. Nakamura, *J. Chem. Phys.*, 2006, **124**(14), 144110.
68. J. C. Tully, Molecular dynamics with electronic transitions, *J. Chem. Phys.*, 1990, **93**(2), 1061.

69. L. Zhu, V. Kleiman, X. Li, S. P. Lu, K. Trentelman and R. J. Gordon, *Phys. Rev. Lett.*, 1995, **75**(13), 2598.
70. C. Zhu, H. Kamisaka and H. Nakamura, *J. Chem. Phys.*, 2002, **116**(8), 3234.
71. N. L. Doltsinis, in *Computational Nanoscience: Do It Yourself!*, John von Neumann Institute for Computing, Jülich, **vol. 31**, 2006, p. 389.
72. G. Granucci and M. Persico, *J. Chem. Phys.*, 2007, **126**(13), 134114.
73. J. P. Malhado, M. J. Bearpark and J. T. Hynes, *Front. Chem.*, 2014, **2**(97), 97.
74. S. Mai, P. Marquetand and L. González, *Int. J. Quantum Chem.*, 2015, **115**, 1215.
75. L. Wang, A. Akimov and O. V. Prezhdo, *J. Phys. Chem. Lett.*, 2016, **7**(11), 2100.
76. J. E. Subotnik, A. Jain, B. Landry, A. Petit, W. Ouyang and N. Bellonzi, *Annu. Rev. Phys. Chem.*, 2016, **67**(1), 387.
77. S. Mai, P. Marquetand and L. González, *WIREs Comput. Mol. Sci.*, 2018, **8**(6), e1370.
78. M. Rupp, *Int. J. Quantum Chem.*, 2015, **115**(16), 1058.
79. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, *J. Mach. Learn. Res.*, 2011, **12**, 2825.
80. D. Uysal and H. A. Güvenir, *Knowl. Eng. Rev.*, 1999, **14**(4), 319.
81. E. I. Im, *Working Paper*, Working Papers 199611, 1996, University of Hawaii at Manoa, Department of Economics.
82. F. A. Faber, A. S. Christensen, B. Huang and O. A. von Lilienfeld, *J. Chem. Phys.*, 2018, **148**(24), 241717.
83. A. Christensen, F. Faber, B. Huang, L. Bratholm, A. Tkatchenko, K. Müller and O. Lilienfeld, https://github.com/qmlcode/qml, 2017.
84. A. S. Christensen, L. A. Bratholm, F. A. Faber, D. R. Glowacki and O. A. von Lilienfeld, *J. Chem. Phys*, 2020, **152**(4), 044107.
85. V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.
86. M. Awad and R. Khanna, in *Efficient Learning Machines*, ed. J. Pepper, S. Weiss and P. Hauke, Apress, Berkeley, CA, 2015, p. 67.
87. A. J. Smola and B. Schölkopf, *Stat. Comput.*, 2004, **14**, 199.

88. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.
89. J. Behler, *Int. J. Quantum Chem.*, 2015, **115**, 1032.
90. D. P. Kingma and J. Ba, in *ICLR 2015*, 2014, arXiv:1412.6980.
91. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, *J. Mach. Learn. Res.*, 2014, **15**, 1929.
92. Theano Development Team, *arXiv e-prints*, 2016, arXiv:1605.02688.
93. I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*, MIT Press, 2016.
94. J. Behler and M. Parrinello, *Phys. Rev. Lett.*, 2007, **98**, 146401.
95. H. Wang and W. Yang, *J. Phys. Chem. Lett.*, 2018, **9**(12), 3232.
96. Y. Huang, J. Kang, W. A. Goddard and L.-W. Wang, *Phys. Rev. B*, 2019, **99**, 064103.
97. M. Gastegger, L. Schwiedrzik, M. Bittermann, F. Berzsenyi and P. Marquetand, *J. Chem. Phys.*, 2018, **148**(24), 241709.
98. M. Rupp, A. Tkatchenko, K.-R. Müller and O. A. von Lilienfeld, *Phys. Rev. Lett.*, 2012, **108**, 058301.
99. J. Behler, *J. Chem. Phys.*, 2011, **134**, 074106.
100. J. E. Herr, K. Koh, K. Yao and J. Parkhill, *J. Chem. Phys.*, 2019, **151**(8), 084103.
101. V. Botu and R. Ramprasad, *Int. J. Quantum Chem.*, 2015, **115**(16), 1074.
102. J. S. Smith, B. Nebgen, N. Lubbers, O. Isayev and A. E. Roitberg, *J. Chem. Phys.*, 2018, **148**(24), 241733.
103. E. Wigner, *Phys. Rev.*, 1932, **40**, 749.
104. R. E. Bruccoleri and M. Karplus, *Biopolymers*, 1990, **29**(14), 1847.
105. T. Maximova, R. Moffatt, B. Ma, R. Nussinov and A. Shehu, *PLoS Comput. Biol.*, 2016, **12**, e1004619.
106. J. Kästner, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2011, **1**(6), 932.
107. G. Tao, *Theor. Chem. Acc.*, 2019, **138**, 34.
108. Y. I. Yang, Q. Shao, J. Zhang, L. Yang and Y. Q. Gao, *J. Chem. Phys.*, 2019, **151**(7), 070902.
109. J. E. Herr, K. Yao, R. McIntyre, D. W. Toth and J. Parkhill, *J. Chem. Phys.*, 2018, **148**(24), 241710.
110. S. Grimme, *J. Chem. Theory Comput.*, 2019, **15**(5), 2847.

111. F. Plasser, M. Ruckenbauer, S. Mai, M. Oppel, P. Marquetand and L. González, *J. Chem. Theory Comput.*, 2016, **12**, 1207.
112. A. V. Akimov, *J. Phys. Chem. Lett.*, 2018, **9**(20), 6096.
113. T. Ishida, S. Nanbu and H. Nakamura, *J. Phys. Chem. A*, 2009, **113**(16), 4356.
114. A.-H. Gao, B. Li, P.-Y. Zhang and K.-L. Han, *J. Chem. Phys.*, 2012, **137**(20), 204305.
115. S. Mai, M. Richter, M. Ruckenbauer, M. Oppel, P. Marquetand and L. González, *sharc-md.org*, 2018.
116. G. J. M. Dormans, G. C. Groenenboom, W. C. A. Van Dorst and H. M. Buck, *J. Am. Chem. Soc.*, 1988, **110**(5), 1406.
117. M. Barbatti, A. J. A. Aquino and H. Lischka, *Mol. Phys.*, 2006, **104**(5–7), 1053.
118. J. Herbst, K. Heyne and R. Diller, *Science*, 2002, **297**(5582), 822.

CHAPTER 5

Machine Learning in Science – A Role for Mechanical Sympathy?

HUGH M. CARTWRIGHT(0000-0003-0949-2657)

Department of Chemistry, Oxford University, OX1 3QZ, UK
Email: Hugh.Cartwright@chem.ox.ac.uk

5.1 Introduction

The phrase “Mechanical Sympathy” has an interesting origin: it was coined by Jackie Stewart, a British racing car driver. You don't need to be an engineer to enter motor racing, he argued, but knowing about engines can make you a better driver. In other words, to get the best out of something, don't be just a passive consumer.[1] As a three-time winner of Formula 1 motor racing championships, perhaps he had a point.

Might “Mechanical Sympathy” be helpful in Artificial Intelligence? Here, we argue that it could.

5.1.1 A Little History

The earliest scientific applications of Artificial Intelligence (AI) were Expert Systems (ESs). Incapable of independent learning, the ES was a crude tool by today's standards, requiring pump-priming with rules:

> “**IF** the pH is greater than 7, **THEN** the solution is alkaline”

> "**IF** there is strong infrared absorption near 1760 cm^{-1}, **THEN** suspect a carbonyl group in the molecule"

Human experts were required to determine which rules should be included in the knowledge base. Despite being time-consuming to construct, in a few fields, such as the interpretation of mass spectral fragmentation patterns, their performance hinted that computers might match, and eventually surpass, human scientific reasoning.[2]

The subsequent development of Artificial Neural Networks (ANNs), on which this chapter focusses, was a key step in the evolution of AI; in contrast with the ES, these needed no instruction. Despite this clear step forward, computational chemists were initially unimpressed; they continued to concentrate on quantum mechanics and crystallography, using algorithms so greedy for CPU cycles that they sucked up half of all computer time in many Universities in the 1960s and 1970s.

It proved impossible to ignore AI indefinitely though. Like a computational Swiss Army knife, AI evolved to encompass a multitude of tools. Today's "narrow AI" can tackle only one type of problem at a time – spectral interpretation or robotic control perhaps – but its range of application is broad. Intrigued by the promise of fast, intelligent interpretation of data, chemists have been adopting AI enthusiastically.[3–7]

What could possibly go wrong?

Quite a lot, in fact.

5.1.2 Challenges

This chapter addresses the challenges that await physical and life scientists when they stick a toe into the warm water of AI. Though intended primarily for those who have limited prior experience with AI in science, it may be helpful also for those with greater familiarity.

The challenges discussed below have the potential to seriously compromise an AI application, which sounds alarming. Would it not in that case be safer for scientists to set AI aside and limit themselves to "conventional" methods of analysis? The answer is undoubtedly no. AI is becoming pervasive in society for a good reason – it works – and it promises to be equally successful in science.

So, this chapter is not a warning sign (“Use AI at your own risk”). Instead, it is an information booth (“Look within”). Practical issues do arise when scientists use AI for the first time; here are suggestions of what to look out for.

But this seems perverse; why are suggestions necessary? After all, if AI is so “intelligent”, why should its use present difficulties? For a clue to the reasons, consider the following:

```
↑1 ω∨.∧3 4=+/,¯1 0 1∘.(⌽∘⍉)⍣2⊂ω
```

You may be surprised to learn that this is a complete, functioning computer program.[8] Written in the wonderfully eccentric language apl (**A** **P**rogramming **L**anguage), it executes an example of John Horton Conway's iconic Game of Life,[9] which formed the basis of Cellular Automata.

It is not hard to see why apl programs earned a reputation for brevity and incomprehensibility. You could run them from a command line, but would need a “real” programmer to create them or explain how they worked.

For a period in the late 1960s and into the 1970s, apl prospered in science, but it faced a formidable competitor. Though much less compact, programs written in FORTRAN were easy to understand, avoiding the opaqueness that characterised apl; furthermore, FORTRAN programming was in the skill set of many physical and life scientists. Organisations such as QCPE[10] offered FORTRAN programs (weighty boxes of punched cards delivered – free of charge! – around the world), which non-computer scientists could use unmodified. Organic chemists became instant near-experts in Molecular Orbital calculations.

As FORTRAN flourished, apl withered; the few available AI programs were underpowered, limited in scope, and quirky, so provided little competition. Fast forward to 2020, FORTRAN is out of favour, and apl has been relegated to a niche role. By contrast, AI is immeasurably more powerful than in its earlier incarnations and, with the benefit of new software, it is easier to use than the arcane programs of a decade ago.

If powerful and rather more friendly AI software has left apl and FORTRAN in the dust, where is the problem? Paradoxically, the danger is ease of use: the more user-friendly the software becomes, the more

tempting it is to just feed in one's data and “see what happens”. And even when running the software is not necessarily simple, it is often still possible to use whatever default parameters are set within the software, and then accept the results if they seem “reasonable”. Most AI programs will generate output, no matter how absurd the input, but confidence in that output requires an understanding of the limitations of the software that we use.

We need to know what is going on under the AI hood.

A bit of Mechanical Sympathy would be perfect.

Let's dive in.

5.2 Problems and Solutions

5.2.1 How Many Samples do I Need to Train an AI?

The favourite tool of nuclear physicists is the Large Hadron Collider at CERN in Switzerland,[11] which can generate 10^{17} bytes of data (enough to fill 100 000 1Tb hard discs) in about three minutes. So much data are produced that, after real-time filtering, nearly all are thrown away;[12] only the most promising are retained for detailed analysis.

Chemists rarely generate data at such speed, but large-scale databases (including PubChem,[13] the Inorganic Crystal Structures Database,[14] Reaxys, SciFinder and ChEMBL[15]) do exist; some are central to applications discussed in this book.

How many samples are needed for training? It might seem that the more data we use, the better the outcome. But it is not quite that simple.

Comment When ANNs first found their way into science, some researchers used their entire experimental dataset for training; the quality of the network was then tested using samples from the same set.[16] This might have seemed a good idea at the time, but you probably already realise that it was not. Double-dipping into the dataset is akin to testing a pianist who has learnt sight-reading using half a dozen pieces of music, by asking them to sight-read one of the pieces they already know. Training and testing are separate tasks: samples used during training cannot be re-used for testing.

Some samples in the dataset must, therefore, be held back for testing; how many? Experience suggests that a division of the dataset into 75% for training and 25% for testing often works well; if a validation step (Section 5.2.5) is used, a 70 : 20 : 10 split for training, validation, and testing is reasonable. However, these splits are based on experience, not theory, so these are not immutable.

How large should the database itself be? The more complex and numerous the relationships between features and observables are (see Box 5.1), the more data are required from which the network can learn. In simple problems, such as determining the concentrations of three coloured dyes from the absorption spectra of multiple solutions that contain them, only a few underlying relationships link concentration to absorption (one per dye, if Beer's law holds) and a small database is sufficient; but most problems for which an ANN might be considered are more complicated and require far more data.

Box 5.1 A Little Terminology.

The role of an ANN is to find *relationships* that link *features* (also known as descriptors, predictors, attributes, feature vectors or fingerprints) to *observables* (also known as targets, ground truth or labels). For example, if the aim was to find the composition of novel solvents for organic materials, the training data might consist of various known solvents whose features included their molecular weight, molecular dipole, synthetic feasibility, pH, number of carbon atoms *etc.*, while the observables could be the solubility of different organic solids in those solvents.

Training takes place in a series of *cycles* (or epochs), during each of which the network is shown the entire training set once.

The difference between the known values of the observables and those generated by the network is the *test* or *validation error* (or loss function).

The search for efficient synthetic routes (discussed elsewhere in this book) is typical of a complex problem in which an ANN may be of use. The

yield of a chemical reaction depends on the identity of reagents, presence of catalysts, solvent, temperature and pressure, reaction phase and other factors, so there are many relationships that contribute to the links between features and observables. A suitable database will be very large.

For some complex problems, the database may, in fact, be so large that training becomes unreasonably lengthy, even after a portion of the database has been reserved for validation and testing. It may then be necessary to use Data Reduction (Section 5.2.4) to slim down the database prior to training.

At the other extreme, some databases are too small, a problem that is more common in chemistry than in many other sciences. To fill in the gaps, additional data might be generated through further experiments, by lifting information from unstructured sources such as journal publications using data mining or by accessing previously unused databases. Google Scholar,[17] a powerful tool for searching scientific publications, has recently been complemented by Google Dataset Search.[18] This executes rapid searches by keyword of many thousands of commercial, non-commercial, government and other websites, a process that can be helpful in locating previously unrecognised databases.

Another potential option when the database is small is Transfer Learning,[19,20] commonly used in image recognition. We first train a network using a large pool of data related to the topic in which we are interested. The trained network then becomes the starting point for a shorter period of training, using samples from the small database. The purpose of this secondary training is to tweak the previously determined network weights; in essence a network trained to solve one problem is tuned to solve a second. The procedure is efficient, making use of databases that might otherwise be too small for effective learning, and can also be beneficial if a database is unbalanced. Neural Turing machines and imitation learning provide further ways to make effective use of small datasets.

A final comment on dataset size: if the dataset is small, you might think that one way to use it to best advantage would be to:

(i) train the network, reserving as usual a portion of the dataset for testing;

(ii) then return the test data to the dataset and use those samples to finish training.

Superficially this is attractive; it seems to resemble Transfer Learning, mentioned above. The 20–30% of the data that the network was not shown during training might contain useful information, so why not give the network this extra information to "top up" its learning?

The problem is that a final polishing of the network using the test set *could* lead to a better model, but we cannot be confident that it *will* do so, as there is no way to test the updated ANN. If there is bias in the test data, for example, the model may degrade rather than improve: we have no means of detecting this. Equally, without an independent check of performance, we might terminate training too early, or allow it to run for too long, leading to an under- or over-trained model. Avoid feeding test data back into the dataset to try to improve the model; it is never wise.

5.2.2 We Are Combining Databases; Do Difficulties Await?

Babies enter a rich, chaotic world and immediately start to make sense of sounds, shapes, lights, and tastes. It would save time if AIs could emulate this,[19] learning from random, unrelated pieces of data; but that's not how AIs work. The difference between a successful application and a failure may be the care with which data are curated. What does that involve?

Comment Without proper data curation, our approach is "hope for the best". This is a poor strategy; in fact, it is not a strategy at all. If we just throw data into a Machine Learning (ML) pot to see what happens, we should prepare to be disappointed, especially if the data are complex, recorded over long periods, subject to significant noise, or derived from multiple sources.

Data collection is just the first step in curation. Not all data are in accessible, well-maintained databases; some might be buried in journals, presentations, figures, tables, lab notebooks, poorly documented experiments, or even diaries. Tools such as ChemDataExtractor[21] can interrogate unstructured sources, but the data that they yield are often in a format incompatible with data obtained elsewhere. Data must be classified and authenticated, labelled, checked for relevance and unreadable entries,

possibly reformatted, and archived in a form suitable for future use – all these are curation tasks.

Information can also be biased (see Box 5.2), intentionally or inadvertently;[22,23] whatever the origin, bias can slow learning and diminish its effectiveness. Data from cheap, quick and easy experiments can be more generously represented in databases than data from more challenging, but information-rich, experiments. As one would anticipate, researchers contribute results mainly from their own field, so this too leads to data being plentiful in some areas, weaker in others. Bias may also arise when data from commercial sources are partially redacted, to prevent competitors from learning about areas that might be commercially sensitive. Even failed reactions could be revealing, so these might be omitted or obfuscated. These effects work against the generation of a successful ML model, which requires that data cover all areas of the search space roughly evenly.

Box 5.2 Bias.

Both the composition of a database and the way that samples are chosen from it can bias learning. Here are some examples:

- Overrepresentation within the database of data that are easy to obtain, while data that are more challenging to collect, but nevertheless potentially valuable, are poorly represented.
- Overrepresentation within the database of data from one's own specialised field compared to other relevant fields.
- Deliberate withholding of pertinent data from the database for commercial reasons, or to prevent others from gaining a research advantage.
- Selection of only a portion of the dataset for learning by excluding samples that might not support what you wish to prove.
- Weighting all samples equally, even though some are known or suspected to be less reliable.
- Selection of samples from a dataset in a non-random fashion; for example, without sufficient evidence to justify the choice, choosing

only data from particular years, or data only from research groups you are familiar with or who work in particular countries.
- Selection of databases without adequate justification for using some and omitting others.

Curation is like digging the garden in the spring, a tedious but unavoidable task before the more gratifying one of harvesting. The process of actually feeding data into an ML application must wait upon curation, a task that may take far longer than training, but one that is nonetheless vital.

5.2.3 We Use a Chemical Reactions Database to Predict Synthetic Routes, but We Are Making Little Progress. What Might Be Going Wrong?

Given high-quality, diverse data, we feel ready to start our ML. But even well-curated databases can contain holes.

Comment Marquee science projects, such as predicting the composition of materials with pre-specified properties,[7] creating a room temperature superconductor,[24] or the computational design of synthetic routes,[25] are technically challenging, industrially important, and commercially valuable.

For such projects, a large dataset is essential. This is the so-called Curse of Dimensionality:[26] as the number of independent features required for effective learning grows, the number of samples needed for training increases roughly exponentially with the number of features. Consider the design of synthetic routes: from Organic 101 you may recall what products to expect when two organic chemicals are mixed, but in planning a viable, high-yield synthetic route this level of knowledge is completely inadequate. Machine-readable, fully contextualised data with “above the arrow” information such as solvents, catalysts, temperature, time, the order of addition of reagents *etc.* are essential. It is not uncommon to find that some of this critical data are not reported for reactions in the literature, severely limiting the value of the data.

Negative data are not as troublesome as unreported data, though; in fact, negative data can be valuable. An ANN cannot learn from data it never sees, but it can utilise negative results.[27] Reactions that fail, have a low

yield, or are extremely slow are far less likely to appear in the literature than successful reactions,[28] but the knowledge that a reaction is not viable allows us to rule out proposed synthetic routes that involve it. If data on failed reactions are missing, it may be possible to estimate them using maximum likelihood methods,[25] or an external analytical model. Alternatively, speculative data for failed reactions may be created for reactions that could be considered to be plausible, but can reasonably be supposed to be failures.[29,30]

5.2.4 Our Database Is so Large That Training Will Take Forever. What Can We Do?

The perfect dataset is comprehensive yet compact, noise-free, machine-readable, evenly spread across the search space, and in a consistent format. Real datasets contain noise, redundancies, duplicates and errors, and may be in a mix of formats (data in both metric and imperial units perhaps, or in numerical and text form). Even worse, some databases may be so bloated that training becomes virtually impossible.

Comment Curation is not a sharply demarcated process; it overlaps with preprocessing, during which we massage information in the database to improve the prospects for learning.

A simple and common type of preprocessing is the scaling of data. If features to be fed into an ANN have very different magnitudes, the larger values may dominate learning, so that features whose values are always small act as little more than noise during the learning process. In principle, the ANN itself is capable of scaling down large values so that they do not dominate learning. However, when large values feed into a sigmoidal activation function, there is only a feeble dependence of output on input, as the activation function plateaus at large input values. Learning is then slow and can be ineffective. Scaling reduces this effect by bringing all feature values into approximately the same range.

Scaling is straightforward, but a more critical challenge arises when a database is so large that it is almost unusable. Some of the largest databases in chemistry comprise molecular data; Reaxys, for example, contains data on more than 100 million compounds. Each molecule may be characterised by hundreds of features, including molecular weight, logP values, number

of chiral centres, molecular polarity, and molecular volume. The inclusion of very large numbers of compounds and numerous features in a database provides great flexibility; data can be added to the database without needing to know which features might be of value in some future application, so a comprehensive database encourages wide applicability. Unfortunately, it can also make the database unwieldy.

The molecular features that are key in an application may lie alongside hundreds of other features, which are irrelevant or redundant, so some way of either identifying those key features or cutting back the dataset can be essential. Features of low information content can be identified using a Genetic Algorithm,[31] and then discarded before training begins. Alternatively, Principal Component Analysis (PCA)[32] can be used to create a new set of linearly independent "components", each of which represents a blend of the dominant features of the data and is associated with an eigenvalue which quantifies the variability in the data that the component encapsulates. The eigenvalues can be used to determine how many components should be retained: components with large eigenvalues are used, while those with small eigenvalues are discarded. (You will recognise that there may be a grey area between the components that we deem to have "large" eigenvalues and so are kept, and the components that are unlucky enough to have "small" eigenvalues, so are candidates for removal.)

As a side benefit, this culling of components with small eigenvalues can help reduce noise. If the signal-to-noise ratio in the dataset is high, most noise should be concentrated in components with small eigenvalues, so removal of these will lessen their influence, while leaving the meaningful features largely untouched. The PCA procedure does unavoidably blend a degree of noise into the sample features though, so some noise is retained in the large eigenvalue components; some information also is lost along with noise by discarding components with low eigenvalues. In addition, the components that are retained do not have a simple physical interpretation.

Nevertheless, data reduction is valuable; it improves efficiency by reducing the size of the dataset, so training should be faster. It may also limit distortion arising from misleading or erroneous data, as the overall effect of a small number of faulty samples within a large dataset may be no greater than the background noise, so will largely be concentrated in the

components with small eigenvalues. Finally, a reduction in the number of features diminishes the chances of overfitting and leads to a simpler model. Although the choice of how many components to keep is a matter of judgement, both experience and the eigenvalues themselves provide useful guidance.

5.2.5 How Do I Set the Parameters?

AIs process data, not meaning; whether they predict the composition of a superconductor or the probability that the Yankees will win the World Series next year, their mode of action is roughly the same. Flexibility and ease of use are among their attractions, but the latter can be a trap: if you train a network “blind”, using the default parameters, sub-optimal or even meaningless results are likely. How should the parameters be set for the best chance of success?

Comment Successful AI applications require more than just high-quality data. The values of several parameters (known as hyperparameters) affect how well an ANN learns, and as these parameters interact in complex ways, it can be difficult to determine the optimum value for every one of them.

Of the parameters in an ANN, the connection weights, which comprise the network's memory, are the simplest to deal with. In fact, we do not need to deal with them at all, for these are under the control of the network itself, and are updated during training.

Although the numbers of input and output nodes are nominally under user control, in practice these numbers are determined from the dataset. One input node is required for every feature (such as each wavelength in an IR spectrum at which the absorption has been measured), and an output node for every observable that the network will predict (the presence or absence of absorption peaks associated with different normal modes of vibration, for example).

The most crucial of the remaining parameters, all of which can be set by the user, are those that define the network geometry. The number of hidden layers and number of nodes per layer determine how many connection weights there are, and this in turn determines how complex a problem the network can solve. Deep Neural Networks (DNNs) have many layers, so these can tackle more complex problems, but in the neural network world,

bigger is not necessarily better. Large networks are prone to overfitting, in which not just relationships but individual samples from the dataset and noise are learnt, diluting network generality. Choices also must be made of the type of activation function to be used, the learning rate and momentum factor and how the data will be split between training, test and validation sets (see Box 5.3).

Box 5.3 Validation.

It might seem that training is completely independent of the test set. The test set is used for just a single purpose: to measure the quality of the trained network; it is never used to update network weights.

However, even though testing appears to be “at arm's length” from training, this hides a problem. Training and testing are not truly independent because *the test set determines when training should be halted*. In this way, the performance of the completed model depends on the composition of the test set. If a different test set were used, training might have continued longer, or be brought to a halt earlier, thereby influencing the extent of learning.

To address this issue, it is common to use a validation set. Like the test set, validation samples are picked at random from the dataset. As the network is trained, its performance is monitored using the validation set, not the test set. When the validation error reaches a minimum, training is stopped and at that point the network is shown the test set; it is the error on this test set that we use as a measure of the network's quality.

Provided that neither the network nor the database is very large, this process can be repeated with different values for the hyperparameters, and different validation sets. One can then choose the values for the hyperparameters that gave the best overall performance.

With several parameters to be set, how should we proceed? First, we should be aware that for many straightforward problems a single hidden layer in the ANN is sufficient. Small networks train rapidly and are good generalisers, so, even though choices must still be made about geometry and learning parameters, a trial-and-error approach is usually feasible.

Small networks are also less likely to learn noise in the data or learn specific examples from it because all the connection weights are needed to describe relationships in the data. A small network with one hidden layer is therefore often an appropriate starting point.

When the data are more complicated, more than a single hidden layer may be required. Although network design is then more difficult, we are not completely in the dark. If PCA has been used for data reduction, the number of principal components that have been retained will be known, and this can be used as a guide to network size. Nevertheless, although the number of parameters to be set is not large, they interact in ways that can be difficult to predict, so a logical approach is helpful. Evolutionary methods, in particular the Genetic Algorithm,[31] have proved to be a popular tool for finding networks of high quality. Their principal disadvantage is that they must be effectively integrated into the ANN, as it is rarely feasible to run a Genetic Algorithm optimisation of a neural network by hand.

A further option is offered by expanding or pruning networks.[33] In the first method, a network that is likely to be too small is trained, and a test set used to assess its performance. A single node is then added to a hidden layer, and the newly enlarged network trained from scratch and re-assessed. Further nodes are added and training repeated until performance (as judged by the test set) shows the network starting to learn noise and examples. The size of the “ideal” network is in this way determined.

We might instead consider an inverse procedure, in which we start with a network thought to be too large, and then repeatedly train, assess, and prune the network, removing one node at a time, until performance begins to deteriorate noticeably. Of the two methods the former is more practical, because the networks being tested are smaller than the optimum, and therefore train rapidly, while in a pruning scheme the discarded networks are all too large, so train more slowly.

You may have spotted a difficulty though: for networks of two or more layers, we must decide which layer is altered each time the geometry is updated. Although in deep networks individual layers may be used to recognise different features of the data, this is rarely feasible in two- or three-layer networks, so an element of trial-and-error is almost unavoidable unless an evolutionary algorithm is used.

5.2.6 How Has My AI Learnt Something That Is Not There?

You have written an ANN program to recognise cats. Keen to avoid bias, you train it with an equal number of images of black cats, white cats and ginger cats, then test the trained network on some fresh pictures of cats and dogs. The ANN consistently identifies dogs as non-cats – that's encouraging – but strangely it sometimes correctly identifies an image as a white cat, but later misidentifies *exactly the same image* as a black cat. What is going on?

Comment This sounds like a minor, tolerable irritation; most AI tools are not correct 100% of the time,[19] so we must be prepared to accept the odd mistake. But there may be deeper problems.

ANNs used on time-based data, such as the progression of illness in an acutely unwell patient or the online control of a chemical flowshop, typically have a feedback (or recurrent) structure. This gives them a temporal memory; in other words, they can learn sequences. Our cat problem has arisen because the ANN has learned from both the training data *and from how the data were presented*. In our eagerness not to bias training, we have shown it a white cat, a black cat, a ginger cat, a white one, a black one, a ginger one, a white one, … and so on. The algorithm has unwittingly been fed extra information:

"if you see a white cat, the next image will be a black cat"

and has learnt from it.

In time series, a knowledge of how one data point relates to its neighbours is vital information, but if you never analyse such series you might feel that you can ignore the issue of presentation.

But there is a fundamental problem here. Databases are frequently organised in a logical fashion (or even in an illogical, but nevertheless ordered, fashion); molecules might appear in the order of increasing molecular weight, all chemical reactions involving amino acids might be collected together, drugs might be listed in the order of decreasing effectiveness, or all data derived from research groups in a particular country might be grouped together. One might note in passing that even if the database as a whole does not display this degree of ordering, some level of organisation is quite common. Data uploaded to a database are often

unintentionally biased – an incoming chunk of data might all be from the same research group perhaps, or all relate to the same group of reactions or the same drug testing program – the database may then contain regions that are not representative of the whole unless the entire database is periodically randomised. If samples are not drawn from the dataset in a truly random fashion, learning can be affected; this is a potential problem even for a standard feedforward algorithm. The solution is simple: just select samples genuinely at random during training.

5.2.7 My Trained AI Tested Beautifully; Now It Produces Rubbish

When a creative person "thinks outside the box" we may be impressed by the result. When an AI does the same, we can expect trouble.

Your PhD student, whose project was to use AI to predict phase diagrams, has given up Chemistry to become a merchant banker. Figure 5.1 shows data from her most recent experiment,[34] which was almost finished when she left to make her fortune.

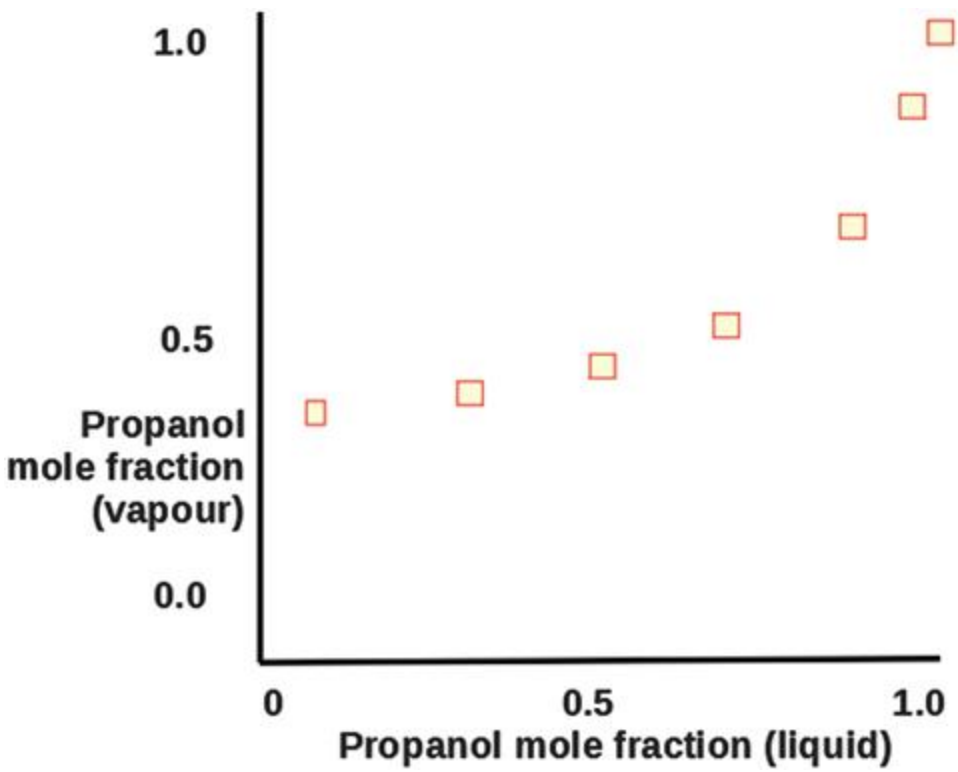


Figure 5.1 The mole fraction of propanol in the vapour phase above a propanol-water mixture as a function of composition. Points at low propanol mole fraction (<0.1M) are missing.

You'd like the complete diagram, so you plug in the experimental data that she has already measured and use her AI to predict the rest of the figure. Are you feeling confident about the result?

Comment Before we tackle your student's problem, take a look at Figure 5.2, which shows how height varies with age for a group of schoolchildren.

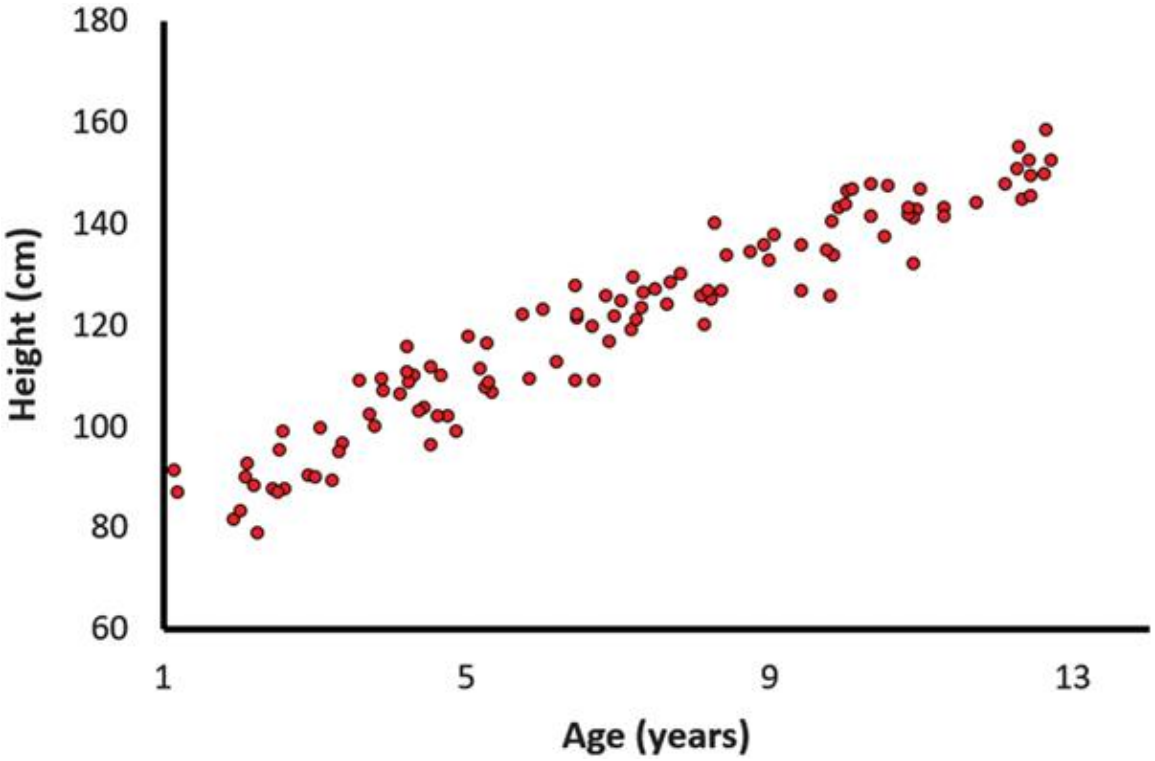


Figure 5.2 Variation of height with age for 100 children.

Let's train an ANN with the data, and then ask it how tall a typical seven-year-old child is. “About 118 cm” it responds.

This seems right; how about a typical 75-year-old? The ANN is confident once again: 580 cm (about 19 feet) is its estimate. Of course, the mistake is ours, not that of the ANN. The network has done as instructed, learning how age (the feature) determines height (the observable). However, even though the trained network answers our queries, the reliability of the answers that it provides diminishes the further away the question takes it

from the boundaries of the dataset. Its estimate of the height of a 14-year-old might be reasonable, as it has seen data relating to children close to that age, but a 75-year-old is so far from any data that it has been trained on that its conclusions are meaningless.

Returning to our scientific example, the full set of experimental data is shown in Figure 5.3.

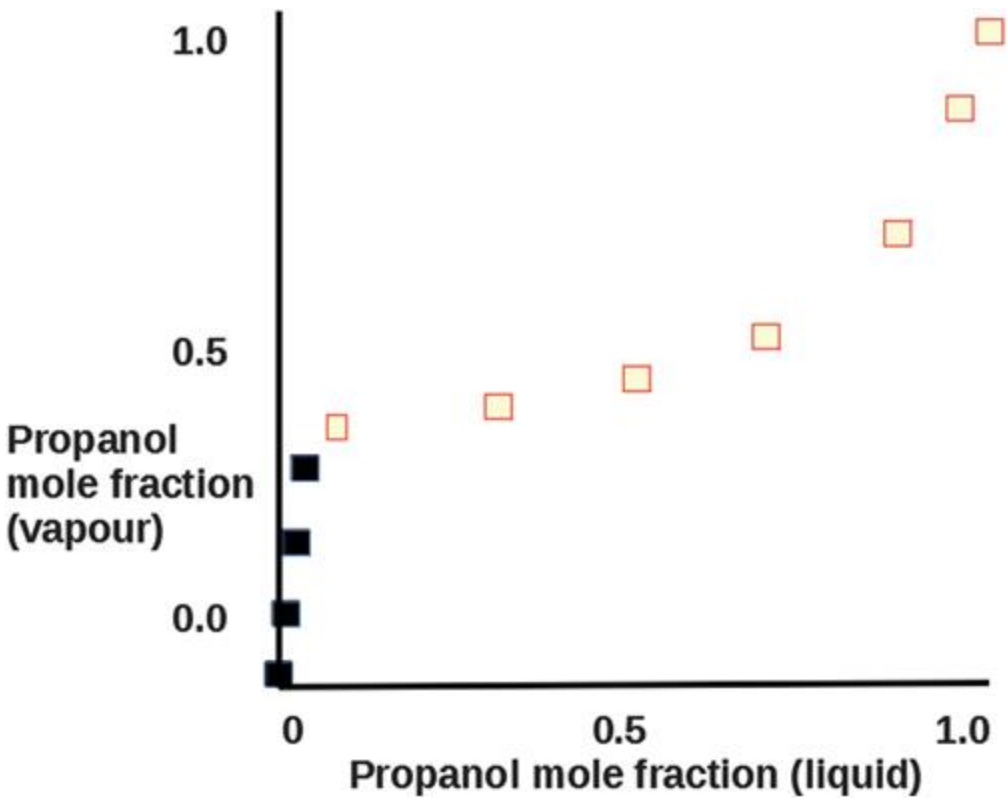


Figure 5.3 The mole fraction of propanol in the vapour phase above a propanol-water mixture as a function of composition. All points, including those shown at low propanol mole fraction, are experimental data.

It is clear that we cannot reasonably expect the AI to adequately predict the four points in black in Figure 5.3. if the only data that it had access to during training was the points shown in Figure 5.1. Confidence in the ANN's responses to questions that lie outside the bounds defined by the information in the database must be low, an observation that may be forgotten when working with very large databases, as we may not even know precisely what the limits are. We conclude that we should be cautious

about accepting the output of an ANN unless we have evidence that the data from which it has learnt suitably cover the range of samples in which we are interested.

5.2.8 My AI Solved the Challenges in Box 5.4; How Did It Manage It?

Box 5.4 What is the next number in each of these series?

Series A: 1, 0, 1, 0, 1, 1, 8, ….
Series B: 6, 6, 7, 9, 8, 6, …

[Not sure of the answers? They follow later.]

If our AI solved the problems, did it use brilliance or luck? There is no use in asking for an explanation; AIs don't show their working.

Comment Explanation is fundamental; in the absence of solid justification, predictions are persuasive only when they are consistently right. Without evidence, a correct prediction might be a stab in the dark, or reliable just within some limited domain.

This is not to say that inexplicable science is valueless. Quantum theory does not provide an explanation for wave-particle duality, yet still is part of the bedrock of science. Nevertheless, if we are offered a startling conclusion without explanation by an AI we might reasonably be cautious.

Though the “knowledge” of an ANN is encoded in its connection weights, an individual weight does not “mean” anything. The weights are not even unambiguous: ANNs of identical geometry trained on the same data can converge on completely different sets of weights and yet still yield similar predictions. It is the combination of weights with network geometry that relates features to observables, but the mathematical relationships encoded within the trained network are far from constituting a coherent explanation.

It is difficult to discern what small networks are telling us with their connection weights, so we can expect that deep networks will present even greater challenges. Their multiple layers combine deep power with deep impenetrability. The performance of DNNs in challenges such as playing Go and in protein folding are impressive, and we may be prepared to tolerate inadequate explanation because of their power. Nevertheless, in safety-critical tasks, such as control of commercial aircraft or nuclear missiles, we should be wary of using methods whose conclusions we cannot interrogate. Even in non-safety-critical applications in science, we should not feel that DNNs are the gold standard (see Box 5.5). On many types of problems, they perform no better than smaller networks and are even more difficult to interpret.

Box 5.5 Some Limitations of Deep Learning Networks.

- The networks are complex, with many weights, so a large quantity of training data is required.
- Training is lengthy, so substantial computer power is needed.
- In common with other ANNs, DNNs currently have a poor ability to explain their “reasoning”.
- DNNs are unable to integrate prior knowledge; for example, they cannot at present explicitly take into account physical laws such as those that govern the electrostatic interaction between two ions (though they may be able to learn about such laws from the data).
- There is evidence that trained DNNs can sometimes be misled by showing them data that differ only marginally from the data on which they have been trained (for example, using adversarial images).
- They are poor at dealing with hierarchical knowledge, a form of information that humans can readily understand and use.

In passing, we should also note that ANNs are not alone in being difficult to interpret. Support Vector Machines project data into high-dimensional space, of which we can only visualise a small portion. Support Vector Machines are no better at explaining what they have done than are ANNs.

Even in non-safety-critical applications, explanation would be helpful. If a black box model cannot be interpreted, it cannot be manually expanded or improved. We cannot pick apart the ANN model and stitch in additional pieces, nor can an ANN learn how to complete a task and then instruct a human on how to carry out the same task efficiently. Future generations of AI tools will not only be good at doing what we cannot, such as calculating how proteins fold, but also good at what we can, for example driving a car. If we are to make the best use of an AI which has learnt skills that humans are familiar with, it will be essential that we can understand how it reached its decisions and learn from them (even be taught by them).

There are ethical considerations too. There is no "morally correct" way to fold a protein or determine whether a black hole lies at the centre of a galaxy, but we can imagine situations in which the decisions of an AI may be both difficult to predict and hard to justify. Suppose an autonomous car suddenly finds itself in a situation in which a collision is inevitable, an accident that will cause injury to you as the passenger (and to the AI itself), or instead to multiple pedestrians. If the car has no ethical compass – and can we at present imagine how it would learn such a thing? – it may prioritise its own, and your, survival over that of many other humans. We could run trial accidents with virtual cars and people to see how the AI reacts, but would we be comfortable with the idea that, after training in such a virtual world, the AI may have developed its own ethical guidelines and yet still be unable to articulate them?

Critical situations arise in other fields such as the control of chemical plants. As AIs control progressively larger sections of a complex plant, our need to understand how decisions are made will grow stronger. The development of AIs that can explain themselves to humans will be an essential requirement as they become ubiquitous.

5.2.9 My AI Thinks that Basketball Players Cause Global Warming

Figure 5.4 demonstrates that high salaries for players in the NBA (National Basketball Association) cause global warming. Other factors that raise the temperature of the earth include a drop in the number of elephants in Norway, and the increasing cost of rubber boots in Belgium.

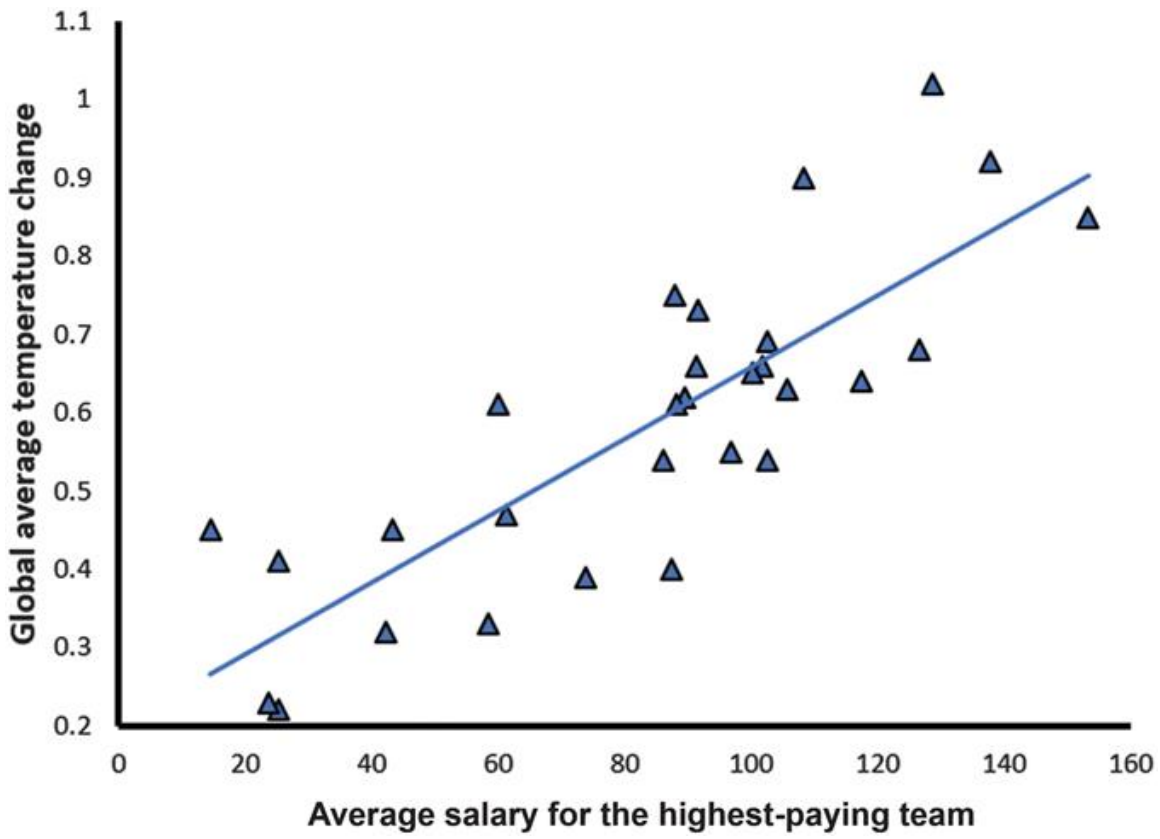


Figure 5.4 The link between the average salary of basketball players and global mean temperature rise.[35,36]

Except, of course, that they don't.

Comment AIs have ability but no intelligence. They can reveal correlations – statistical measures that describe the degree to which two sets of values are related – but ANNs cannot determine whether correlation indicates causation, where the change in one variable is directly responsible for a change in another. Meaningless correlations, for example between the population of Scandinavian elephants and global temperature, constitute the problem of “false discovery”.

False discovery took on particular significance in the middle of the last century. The rate of lung cancer was rising (the illness was so rare at the start of the 20th century that many physicians would see no cases during a lifetime in practice), but the cause of the rise was unknown. In 1950, Bradford Hill and Richard Doll, having previously theorized that exposure to tarmac or car fumes might be responsible, published findings that suggested that smoking cigarettes was to blame instead.[37]

The response of cigarette manufacturers was predictable: outright denial. Lung cancer was not caused by smoking, they argued, but by an unidentified heritable factor that made some people unusually susceptible to lung cancer; the same factor made people prone to addiction to tobacco. The smoking↔lung cancer correlation was there, but the link was this unknown heritable factor, not smoking. Genetics (had it been the real explanation) was what is known as a confounding variable; a factor that affects two completely separate elements – in this case lung cancer rates and tobacco addition – that are not linked causally to one other. Years passed before the companies admitted that their introduction of a confounding factor was not based on any meaningful data and that smoking did indeed cause lung cancer.

As we see with NBA player salaries and global warming, the values of two properties may be correlated without a linking mechanism. Surprisingly, the reverse can also be true: one factor may directly affect another and yet not be correlated with it. An example is a thermostat, which maintains a constant temperature in a house. As the outside temperature falls, the thermostat runs the heating for longer and the heater burns more gas. As the temperature in the house is virtually constant, there is a negligible correlation between the temperature of the thermostat and the amount of gas consumed, even though the thermostat causes the change in gas consumption.

Correlation (also called association) is not causation. But how can we distinguish one from the other? Teasing out the difference can be complex. Bayesian networks,[38] already used in speech recognition, weather forecasting, and other areas, are a potential solution for some problems. These are more open to scrutiny than most types of ML, so in favourable cases, it is easier to disentangle correlation and causation.

There is not enough space here to delve into Bayesian methods, and, though useful, they are by no means an infallible way of detecting causation. The message is that if your ML application seems to suggest that two parameters are correlated, be cautious in assuming that one must cause the change in the other. Scientific input to back up the conclusion of the ANN is necessary.

5.2.10 Results from my AI Suggest that my Data Contain only Noise, but I Think there Might be Something Real present. How can I Tell?

Almost all scientific datasets contain noise. Some noise may be averaged out as the ML learns, but what if that noise is not noise at all and learning is destroying the signal?

Comment As they orbit their star, planets beyond our solar system may pass between the star and the earth, causing a brief drop in the amount of starlight reaching us. Astronomers use the change in intensity during these transits to infer the presence of a planet.[39] As planets are so much smaller than stars the changes are slight, so the alteration in the signal may be swamped by the background variability of the starlight. How can we deal with this?

A potentially useful approach is PCA, which we met in Section 5.2.4. The eigenvalues that a PCA treatment of a dataset yield tell us how much variability in the dataset each eigenvector represents. If the dataset contained nothing but noise, the values of the eigenvalues would be expected to decrease in a roughly monotonic manner. If there are genuine, albeit small, signals in the data, this should be reflected in one or more eigenvalues whose size sets them apart from eigenvalues which just represent noise.

If we know what the signal might look like, instead of training the ANN using raw data, we can train it using examples of the signal. Astronomers are keen to detect amino acids in interstellar space as possible indicators of life elsewhere in the galaxy. Traces of amino acids have been found within meteorites, and so it is reasonable to infer that these molecules might also exist in interstellar space, but they are difficult to detect because their concentration is apparently very low. An ANN can be trained on a set of

laboratory spectra of amino acids and then fed experimental spectra from astronomical observations to look for the very weak signals which might suggest the presence of these molecules. The reverse process, in which an ANN is trained using experimental astronomical observations and then shown the spectrum of a sample amino acid is likely to be less successful because of the incorporation of large amounts of noise into the model. Neither approach is trivial to execute because of the extreme weakness of the astronomical signals (if indeed they are there at all). Nevertheless, AI methods show promise in interpreting weak signals such as these.

5.2.11 My AI Was Fine Until I Found Some Extra Data to Give It; Why Did the Data Make It Worse, Not Better?

Narrow AI is good at learning one thing but struggles to go further.

Comment We train an AI until learning is maximised as measured by performance on a test set. If training is later resumed, using additional, previously unseen data, what was learnt before may be slowly forgotten, as new information begins to overwrite the old (but see Transfer learning in Section 5.2.1); loss of prior learning is especially likely if the new data lie in a very different area to the old. An algorithm trained to recognise dogs will gradually lose that ability if subsequently it is trained with further images that include only other animals.

Deep learning avoids some of these problems, as a DNN can be constructed with constrictions that may force different layers to learn different aspects of the data. Even then, if a trained network is flooded with fresh data that lie in a new area knowledge will degrade. A DNN can play Chess, play GO, or fold a protein, but cannot do all three tasks.

5.2.12 My AI Struggles Even Though It Has Plenty of Data

Being able to calculate the error or loss function is an absolute requirement for an ANN; without access to the values of observables, an ANN cannot learn. Before learning starts, we must define a goal; this may be quite simple to express, recognising fractures in X-rays perhaps, or predicting the composition of ultrahard materials. But the ANN needs a goal in scalar or

vector numerical form, not a general and woolly wish list. Defining goals can be challenging; some problems are ill-formed. What does this mean?

Comment The goal of an ill-formed (or ill-defined) problem is unclear or difficult to quantify. The most common reason is that the overarching goal comprises many sub-goals, with little chance that all of them can be satisfied simultaneously.

Suppose that we decide to use AI to devise a new polymer film, which will act as a temporary skin for people who have suffered serious burns. We require that the product should be effective, non-toxic, retain its properties long enough that the skin underneath it can properly recover. There are other possible criteria too: it should be straightforward to apply so that it could be used in an emergency by untrained personnel; it should not bind to the recovering skin; it should be cheap to produce, have a long shelf life, be easy to synthesise, and promote healing. The list of goals is long, and we might at first believe this to be an advantage: the more goals there are the more precisely the ideal product can be defined. By giving the ANN very specific guidance about these goals surely it can home in on precisely the right formulation? The reality is just the opposite.

There are both practical and ethical problems in using our AI in this example. Recall that an ANN attempts to minimise the difference between its actual output for a sample and the desired output. For all samples in our dataset, we must, therefore, start by determining the observable(s) by combining the many contributing factors, each separately weighted to reflect their relative importance. But “relative importance” is a matter of judgement. How should ease of use of the product be balanced against complexity and safety of the production process? What level of toxicity is acceptable: if one person dies while 99 are saved is that acceptable? What about five deaths, or ten? Suppose that the product would save lives but is so expensive that it would draw funds away from other crucial health services; how do we judge whether the new polymer skin is, overall, beneficial, taking into account the negative effects elsewhere arising from loss of funding? Or – even more problematic – suppose that the polymer skin might work well for people with one particular blood group and not at all for those with a different group; would it be appropriate to fund development and deployment of a product of benefit to only a minority of the population?

Left to its own devices, we cannot expect that the ANN will be able to optimise every observable independently. Instead, it might propose a product that is both extremely effective and extremely costly, or one that is cheap, easy to use, and toxic. Consequently, before we attempt to train the network, we must make these judgements ourselves, possibly on very limited data. Ill-formed problems are not unusual; they are often among the really complex problems that we would most like to solve. As AI increases in power, many more such problems will become amenable to attack. The technical challenges of constructing an ANN, which can tackle such problems are minor compared to the larger questions of how multiple goals should be combined. Even when ethical issues do not complicate the issue, there are no simple answers here.

5.2.13 The Problem: My AI Is Just Odd

AIs can produce curious results, even when they behave as designed.

Comment AI is barely out of its diapers, so we should not be surprised when applications don't always turn out as planned.

Consider what may happen when the great majority of cars are autonomous. The cars must assure the safety of both passengers and pedestrians, so if a pedestrian suddenly steps off the pavement in front of a car, it must stop, and move only when the pedestrian is clear. If another pedestrian then steps into the road, the car must stop again. What will be the likely consequence in cities with many cars? Pedestrians will quickly realise that it is safe to step into the road, no matter how heavy the traffic. Before long, traffic flow will come to a halt. The ultimate in safe roads, perhaps, but the demands for policing of pedestrian behaviour using methods such as face-recognition technology may become overwhelming.

Unexpected behaviours will not be confined to the roads. Consider an AI whose role will be to control an industrial plant in which chemicals flow into an open tank, react, and then leave through an outlet pipe. If the tank overflows, safety will be compromised, so the AI must learn to adjust the rate of liquid inflow to prevent this happening. We train it on a virtual model of the plant, using a loss function which penalises the AI whenever liquid overflows from the tank. It should soon learn how to keep the liquid level within safe limits, but we might not anticipate that it will learn that the

simplest way to do this is to turn off the liquid flow entirely. Safe it is, efficient it is not.

In simple, readily interpretable units such as a chemical flow system, it is possible to spot and correct unwanted behaviour at an early stage. The more control that AIs have, however, the more complicated and inscrutable their behaviour will become; it will then be imperative that any system put in place can be demonstrated unambiguously to behave in a reliable and logical way.

5.2.14 It Is Going Pear-shaped. What Can I Do?

In this final section, we look briefly at a further selection of problems you might encounter with your ANN.

5.2.14.1 The Network Is Not Learning

Check the data. If there is little sign of learning, test the ANN using a well-studied, small dataset, such as the Iris.[40] Learning for this dataset should be nearly complete within at most a few dozen cycles; if it is not, the data may be being read incorrectly, or the network is not working correctly. Could the ANN be reading the data column by column instead of row by row, or *vice versa*? If the data are being read correctly, and you are able to inspect the network weights, feed a single sample through and recheck the weights by hand after backpropagation to see that they have updated as intended. It's tedious, but this is a thorough way to find programming errors.

Check the activation function. Step functions do not work for meaningful problems; if you are using one, change it to a sigmoid function like tanh or ReLU.

Check the data labels. Perhaps samples in the dataset are being read correctly, but are mislabelled, so the observables do not relate to the samples to which they are attached. Check the labels in a portion of the dataset by hand. If they seem correct, train a small network using just this checked set and see if testing shows that learning is improved.

Is there anything there to learn? The data may contain nothing that links input to output. This is more likely if you are using a large dataset and have selected a subset of sample features and/or observables for training. Stop

the training, expand the dataset by around 20%, adding samples that contain only random data, then train again from scratch. If performance is no worse the datasets may contain no information that the network can use.

Check the network geometry. The network may be too large. If so, it may memorise each sample in the dataset individually and not discover any relationships, thereby avoiding learning the rules needed to identify any test samples. Start training with a small network and a modest number of samples. If there is evidence of learning, gradually increase the number of samples to prove that there is data that the network can learn.

5.2.14.2 My Network Learns Very Slowly.

Are you too eager? Training an ANN can be slow. Typical networks require hundreds or thousands of epochs to learn effectively, even more if the dataset is very large. Check to see how the validation error is behaving. If it is improving, and performance on the test set is doing the same, keep going.

The samples are very noisy If the samples are noisy, use a larger dataset, if one is available, to diminish the effect. If the signal is comparable to the noise, learning will be slow and may be ineffective. Try using PCA to shrink the dataset and see if removing the components with the smallest eigenvalues reduces the noise. PCA is not magic though – it becomes progressively less effective at removing noise as the signal to noise ratio decreases and the signal gets swamped in noise.

Is it possible to identify samples within the dataset that are more reliable than the rest (experimental data from a well-respected lab, perhaps)? Try learning from these samples to see whether performance improves. But beware – such a procedure will introduce bias because you are selecting from a favoured portion of the dataset that you believe will serve your purpose best.

Poor links between input and output. The database may contain useful information, but the features you are feeding in may be linked only poorly to the output.

Unbalanced data. The dataset may be biased towards a subset of poor-quality samples. Perhaps they overwhelm other samples whose features have far smaller values; consider whether the entire dataset could benefit from scaling or whether individual samples could be weighted. If there are

many samples of one sort and far fewer samples of others consider whether the dataset could be supplemented, or trimmed without too much loss of information, to even it out. Again, though, be aware of the danger of introducing bias when removing or weighting samples.

5.2.14.3 The Loss Function Diminishes During Training, and the Testing Error also Falls. I Expected that a Rise in the Error on the Test Set Would Indicate When to Stop Training, but This Rise Never Comes

Inappropriate sampling of the database. How are samples drawn from the dataset for training and testing? You already know that these must be taken at random, but the division into the training set and the test set must be made before training begins and then left unchanged. If you instead choose a set of training samples at random, train for one epoch, and test with the samples left over, then draw another set of random samples to train on during the next epoch and use the (new) samples left over the testing you are in effect using the whole dataset for training (see Figure 5.5). As you test with the samples that the network has previously seen, you can expect testing performance to continue to improve with time, even though this improvement is not meaningful.

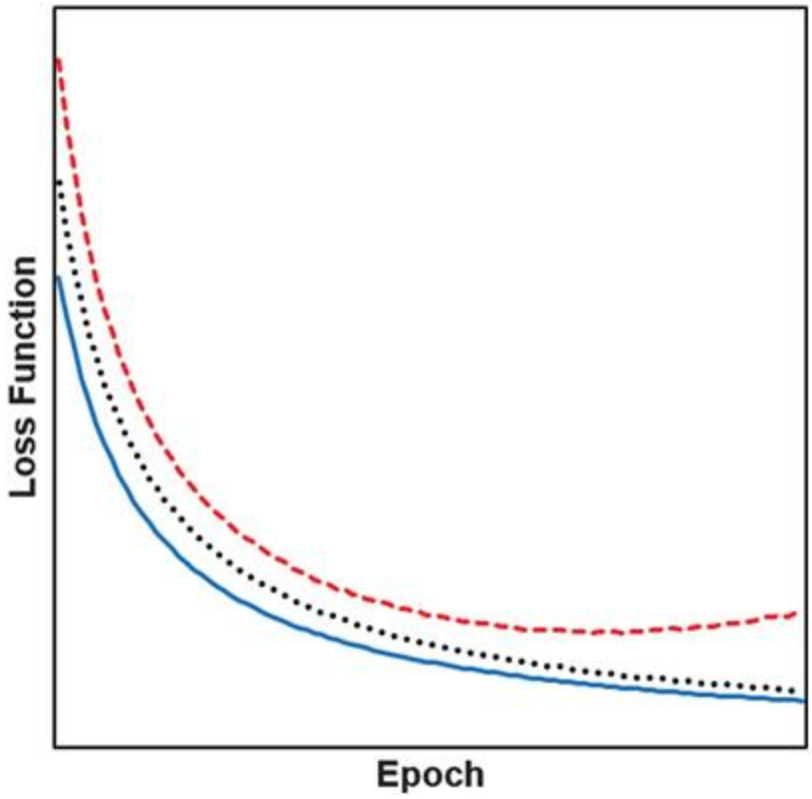


Figure 5.5 The variation of the loss function (solid line) and testing error (dotted line) as the ANN learns. The testing error has continued to decline as learning proceeds, even though it should reach a minimum and then start to rise (dashed line).

5.2.14.4 During 20 Epochs, Performance on the Test Set Improves Very Little, and Then It Starts to Get Worse. Does This Mean That My Network Has Learnt All It Needed to Know in Just 20 Passes Through the Dataset?

Network problems. If training appears to be complete in 20 epochs, consider one of the following: (i) yes, learning is over, but the problem is so straightforward you didn't need an ANN anyway; (ii) you are being too impatient – the problem may be tough and training needs more time; (iii) the network is far too large for the number of samples you have, so the network is learning individual samples (the number of samples should be *substantially more* than the number of connection weights); (iv) training is not working. Perhaps the learning rate is far too high or low. Or the connection weights have been given initial values that are very large, so the

algorithm needs a long time to drag these down into a range at which learning can progress effectively.

Dataset problems. The samples you have used for training and testing may cover different sets of relationships, so what the network learns from the training set is of minimal value in predicting results for the test set. Check whether your dataset has been ordered in some fashion and you are not using random sampling from the entire dataset.

5.2.14.5 I Think That My Network Is Overfitting; What Should I Do?

Consider regularisation. The ideal network encapsulates all the rules in the dataset, but nothing else. The larger the network, the more likely it is that it will learn noise and individual samples, so a reduction in network size is likely to be helpful. Try adding a regularisation term[41] to the cost function in order to penalise complexity. Use the smallest number of descriptors that seem necessary to fully describe the dataset. If you have run PCA on the samples and found that there are just a few components with large eigenvalues and the rest are small, you should not be using a very large number of features.

5.2.14.6 After Training, I Compared the Network Weights with Those My Colleague Got Using the Same Dataset; They Were Completely Different. Who Is Right?

See Section 5.2.8. There is probably nothing wrong. If you and your colleague were asked to propose solutions to the equation $a+b=16$, she might suggest $a=5$, $b=11$, while you offer $a=b=8$. You are both correct because – trivially – an infinite number of solutions exist. Similarly, the ANN can encode relationships in its weights in an unlimited number of ways. You should not expect the connection weights in two networks to be the same, even if network performance is similar.

5.3 Conclusion

The analysis of scientific data can be challenging at any time, but analyses that incorporate AI methods bring with them a slight air of mystery. The connection weights in an ANN are open to inspection, but the understanding that we can draw from those weights is limited. One might compare that with the simplicity of a mathematical procedure such as fitting a line through a set of experimental data points.

This lack of interpretability will continue to be a key issue as the use of ML methods in science expands. While we can manually check the coding of programs to reassure ourselves that the software works as intended, checking that an AI program has reached a justifiable solution is harder. Subsequently being able to understand how it arrived at that conclusion is tougher still.

Fortunately, AI algorithms are evolving. The need for software that can offer some form of explanation is well recognised, and tools that are both smarter and more capable of explanation are under development.[42] It is reasonable to suppose that in due course our algorithms will not be limited to giving a reliable, but explanation-light, analysis of a dataset, but be able to curate datasets, select a suitable ML method, choose optimum values for all the relevant parameters without input from the user, and, once training is over, explain what they have found in terms that we can understand.

In the meantime, as we wait for fully explainable AI, it is important to have an understanding of what goes on under the surface and be able to diagnose the problem when things do not work out. A little Mechanical Sympathy may help ensure that your algorithms have the best chance of generating a meaningful model.

And, in case you are wondering about the solutions to the problems in Section 5.2.8, check out Box 5.6.

Box 5.6 Answers to the Problems in Box 5.4.

In Series A, the numbers are the second digit in the number of days in successive calendar months, starting with August (31) and September (30), and finishing with February (28, in a non-leap year). The missing number is therefore 1, as March has 31 days.

Series B gives the number of letters in successive days of the week (in English), starting with Sunday (6 letters) and Monday (also 6). The missing number is 8, as Saturday has 8 letters.

References

1. https://www.autosport.com/performance/feature/8318/how-mechanical-sympathy-improves-speed; accessed December 2019.
2. E. A. Feigenbaum and B. G. Buchanan, *Artif. Intell.*, 1993, **59**, 233.
3. A. Aspuru-Guzik, M.-H. Baik, S. Balasubramanian, R. Banerjee, S. Bart and N. Borduas-Dedekind, *et al.Nat. Chem.*, 2019, **11**, 286.
4. A. Fabrizio, A. Grisafi, B. Meyer, M. Ceriotti and C. Corminboeuf, *Chem. Sci.*, 2019, **10**, 9424.
5. C. W. Coley, D. A. Thomas III, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen and V. Schultz, *et al.Science*, 2019, 365.
6. R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling and D. Sheberla, *et al.ACS Cent. Sci.*, 2018, **4**, 268.
7. K. T. Butler, D. W. Davies, H. M. Cartwright, O. Isayev and A. Walsh, *Nature*, 2018, **559**, 547.
8. https://dfns.dyalog.com/n_life.htm; accessed December 2019.
9. https://bitstorm.org/gameoflife/; accessed December 2019.
10. D. B. Boyd, Pioneers of Quantum Chemistry, *ACS Symp. Ser.*, 2013, **1122**, 221.
11. https://home.cern/science/accelerators/large-hadron-collider; accessed December 2019.
12. https://home.cern/science/computing/processing-what-record; accessed December 2019.
13. https://pubchem.ncbi.nlm.nih.gov/; accessed December 2019.
14. https://icsd.products.fiz-karlsruhe.de/; accessed December 2019.
15. https://www.ebi.ac.uk/chembl/; accessed December 2019.
16. Early studies which used no independent training set, or where a network contained far more connection weights than the number of

samples, were not uncommon. To protect the innocent, as the saying goes, no examples are given here.
17. https://scholar.google.com/; accessed December 2019.
18. https://toolbox.google.com/datasetsearch; accessed December 2019.
19. Why deep learning Ais are so easy to fool, https://www.nature.com/articles/d41586-019-03013-5; accessed December 2019.
20. G. Koch, R. Zemel, and and R. Salakhutdinov, *Proceedings of ICML 2015 – the 32nd International Conference on Machine Learning, Deep Learning Workshop*, **vol. 2**, 2015.
21. http://chemdataextractor.org/; accessed December 2019.
22. Look out for potential bias in chemical data sets, https://www.nature.com/articles/d41586-019-02670-w; accessed December 2019.
23. J. Sieg, F. Flachsenberg and M. Rarey, *J. Chem. Inf. Model.*, 2019, **59**(3), 947.
24. V. Stanev, C. Oses, A. Gilad Kusne, E. Rodriguez, J. Paglione, S. Curtarolo and I. Takeuchi, *NPJ Comput. Mater.*, 2018, **4**, 29.
25. S. Steiner, J. Wolf, S. Glatzel, A. Andreou, J. M. Granda and G. Keenan, *et al.Science*, 2019, **363**(6423), DOI: 10.1126/science.aav2211.
26. https://en.wikipedia.org/wiki/Curse_of_dimensionality; accessed December 2019.
27. P. Raccuglia, K. C. Elbert, P. D. F. Adler, C. Falk, M. B. Wenny and A. Mollo, *et al.Nature*, 2016, **533**, 73.
28. This is a kind of Survivorship bias, where data about meaningful experiments is widely reported while data about failed experiments is not. One effect of this might be to give the impression that research in a particular field is more successful than it really is, since the number of reported failures is low.
29. M. H. S. Segler and M. P. Waller, *Chem. – Eur. J.*, 2017, **23**, 6118.
30. M. H. S. Segler, M. Preuss and M. P. Waller, *Nature*, 2018, **555**, 604.
31. M. Mitchell, *An Introduction to Genetic Algorithms*., MIT Press Cambridge MA, USA, 1998.
32. I. T. Jolliffe, *Principal Component Analysis*, Springer, New York, ISBN 978-0-387-22440-4, 2002.

33. S. Curteanu and H. M. Cartwright, *J. Chemometrics*, 2011, **25**(10), 527.
34. A. Wakisaka, K. Matsuura, M. Uranaga, T. Sekimoto and M. Takahashi, *J. Mol. Liq.*, 2011, **160**, 103; A. Wakisaka and T. Iwakami, *J. Molec. Liquids*, 2014, **189**, 44.
35. https://hoopshype.com/salaries/; accessed December 2019.
36. https://data.giss.nasa.gov/gistemp/graphs/graph_data/Global_Mean_Estimates_based_on_Land_and_Ocean_Data/graph.txt; accessed December 2019.
37. R. Doll and A. B. Hill, *Br. Med. J.*, 1950, **2**(4682), 739.
38. J. Pearl and D. Mackenzie, *The Book of Why: The New Science of Cause and Effect*, Basic Books, New York, 2018.
39. https://exoplanets.nasa.gov/alien-worlds/ways-to-find-a-planet/; accessed December 2019, https://mcdonaldobservatory.org/news/releases/20171214; accessed December 2019.
40. https://www.kaggle.com/arshid/iris-flower-dataset; accessed December 2019.
41. D. C. Elton, Z. Boukouvalas, M. D. Fuge and P. W. Chunga, *Mol. Syst. Des. Eng.*, 2019, **4**, 828.
42. https://cra.org/ccc/visioning/visioning-activities/2018-activities/artificial-intelligence-roadmap/#roadmapdoc; accessed December 2019.

CHAPTER 6

A Prediction of Future States: AI-powered Chemical Innovation for Defense Applications

TYLER STUKENBROEKER(0000-0002-7889-7183)* AND
JONATHAN CLAUSEN(0000-0002-1628-7136)

Strategic Analysis, Inc., Arlington, Virginia, USA
*Email: tstukenbroeker@sainc.com

6.1 Introduction

The U.S. Department of Defense (DoD) is a unique consumer of chemicals in several ways. First, as an employer of 2.9 million soldiers, civilians, and reservists, it has an unmatched need for pharmaceuticals, adhesives, fuels, coatings, and other chemical products. Second, many of these products are utilized outside of normal supply chains, whether at a forward operating base in Afghanistan or on an aircraft carrier in the Pacific. Finally, the use case for some of these products can be dynamic and unconventional. Missions may require items to be customized on the fly to meet the needs of a warfighter. For instance, a medicine may need to be produced without the use of refrigeration, or perhaps an obscurant is required that will effectively mask a specific environment.

Traditional chemical discovery and manufacturing capabilities rely on manual human planning that requires specialized knowledge and training.

To conduct chemical synthesis using an ever-expanding toolbox of reactions, reagents, and methods, a highly trained chemist or engineer is required. This is not feasible for many military settings and situations, as the need may arise on short notice, in a remote area with little infrastructure, or in the absence of expert personnel. These constraints are even more restrictive when novel chemicals are required—to the point that developing new compounds for defense occurs in a relatively small number of laboratories where discoveries may take years or decades to reach the warfighter. The ability to automate chemical planning and discovery would have a dramatic impact on DoD's ability to carry out its mission.

The Defense Advanced Research Projects Agency (DARPA) makes pivotal investments in breakthrough technologies for national security.[1] This has given rise to ground-breaking defense and civilian technologies such as GPS, self-driving cars, night vision, and of course, the internet. DARPA recognized that the same type of large-scale, collaborative effort which led to these breakthroughs was needed to address chemical automation, and launched the Make-It program in 2015 to automate the synthesis of any molecule.[2] The program drew from the DARPA Battlefield Medicine program,[3] which built a continuous flow chemical system for the on-demand production of 15 different active pharmaceutical ingredient (API) molecules. Make-It sought to push this to the extreme: hardware that could be automatically configured to carry out any organic synthesis route that could be done in a laboratory or chemical plant. However, it became clear that this represented only part of the challenge—more daunting was the development of artificial intelligence (AI) algorithms that could devise the best route and conditions for the synthesis of organic molecules.[4] Make-It revealed the challenges of applying AI to the molecular domain. A new DARPA program launched in 2019, Accelerated Molecular Discovery (AMD),[5] seeks to address this by integrating AI models with experimental systems for autonomous molecular design. Other U.S. government agencies, such as the National Institutes for Standards and Technology, the National Institutes of Health, and National Laboratories, have also recognized the potential for integrating experts from the AI community with those in materials[6] and therapeutics[7] discovery, suggesting that though currently on the vanguard, such collaborations may soon become routine.

This chapter will discuss progress in computer-aided synthesis planning (Section 6.1), challenges acquiring and working with chemical data for machine learning (Section 6.2), potential solutions to these challenges (Section 6.3), and their integration into the research workflow (Section 6.4). It is not intended to be a comprehensive review but rather a survey of the emerging challenges and opportunities (particularly those of the DoD) through the lens of recent research. *The views and opinions expressed in this article are those of the authors and do not reflect the policy or position of DARPA, the DoD, or the US Government*.

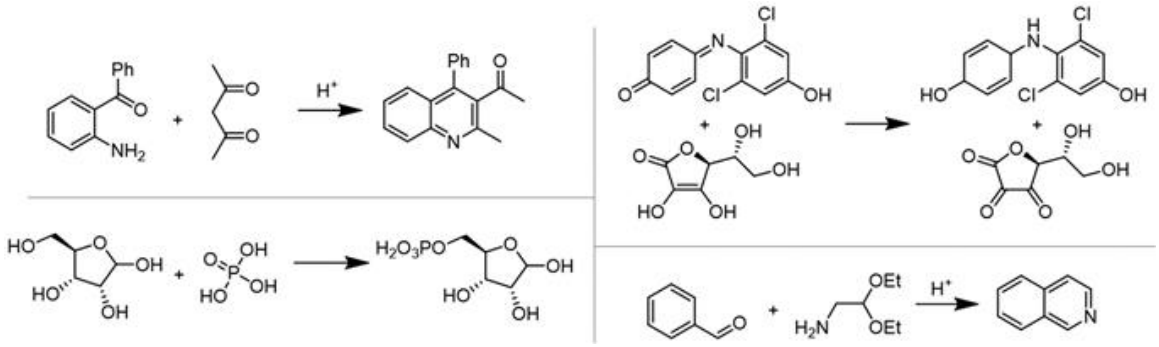
6.2 The Synthesis Search Engine: Automating Synthetic Planning

In this section, we discuss algorithms that leverage machine learning (ML) in the context of other discoveries and demonstrations in the field. First, ML-driven optimization of known reactions is presented (see Section 6.2.1). The remaining sections deal with the synthesis of novel compounds, starting from single reactions (see Section 6.2.2) and then complete retrosyntheses (see Section 6.2.3).

6.2.1 Optimizing Known Routes

Most experiments carried out in chemistry laboratories are not novel reactions. Researchers are typically repeating a reaction from the literature or a laboratory notebook for the purpose of synthesizing a compound for further use. The main concern, therefore, is not what product will form but how efficiently it can be generated. There are many conditions which can be altered: time, temperature, and solvent, to name a few. Chemists can use heuristics and past experience to guess the optimal values, but the number of parameters makes the optimization procedure better suited for statistical algorithms. Indeed, design of experiments (DoE) approaches have been demonstrated which can find the optimum conditions with a minimum number of experiments using algorithms such as stable noisy optimization by branch and fit (SNOBFIT).[8]

In 2017, Zhou *et al.* demonstrated an ML approach to DoE optimization.[9] A reinforcement learning system was designed using a recurrent neural network to optimize concentration and temperature for several reactions (see Scheme 6.1). The system outperformed non-ML algorithms by requiring 71% fewer steps to identify optimal conditions. By pairing the system with a droplet-based screening platform, optimal reaction conditions could be determined in about 30 minutes for the four reactions tested (see Scheme 6.1). We expect to see continued integration of ML algorithms into reaction optimization systems such as those in use in academic[10,11] and pharmaceutical labs.[12,13]

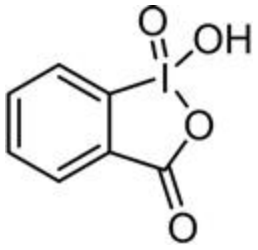


Scheme 6.1 Reactions investigated by Zhou *et al.*[9] Reproduced from ref. 9, https://doi.org/10.1021/acscentsci.7b00492, with permission from American Chemical Society, Copyright 2017.

Given the long history of ML for optimization in fields such as power management,[14] manufacturing,[15] logistics,[16] and telecommunications,[17] it is not surprising that these types of algorithms can help scientists optimize reactions. However, the ability to screen non-discrete variables such as solvent or catalyst requires more specialized chemical knowledge to be integrated. Ahneman *et al.* demonstrated the possibility of this approach by employing a random forest to predict yields using a variety of palladium catalysts for a cross-coupling reaction.[18] Using high throughput reaction screening data, they were able to predict which catalysts would best pair with particular substrates.

The Ahneman work highlights the value of pairing ML with reaction screening. The greatest impact of this combination, however, could lie

beyond studying a single system. For an individual laboratory, it may not be worthwhile preparing small amounts of compounds to improve a suboptimal (*e.g.*, low yielding, poor e-factor, hazardous) reaction. In this way, the existing route or a reference to the original procedure is propagated in literature, leading to suboptimal routes being used repeatedly. But with just a few of these tools, distributed across laboratories working in different areas of chemical synthesis, improved conditions could be determined and propagated across an entire field, resulting in significant aggregate savings of time, energy, and materials. As an example, consider 2-iodoxybenzoic acid (IBX), a commonly used oxidation reagent first prepared in 1893 by Hartmann and Meyer (Scheme 6.2).[19] Historically, the poor solubility of this reagent has limited its use to a few solvents. Countless chemists toiled away using high-boiling DMSO until 2002 when Moore reported that the reaction was effective in more facile solvents.[20] Though it may not have been worthwhile for a single chemist to screen different solvent systems, the cumulative benefit across the field was substantial. The problem is likely more pronounced for less common reagents, where the incentive to publish improved preparation or methodology is further diminished.



Scheme 6.2 Structure of 2-iodoxybenzoic acid (IBX).

Flexible flow chemistry systems are becoming more common for scalable synthesis, even at the laboratory level. When combined with ML algorithms, these systems can perform on-the-fly optimization of temperature, concentration, and flow rate in a continuous or discrete manner. Thus, ML and automated hardware are a way to lower the barrier of process optimization for small-scale syntheses. Combined with proper documentation and data sharing (see Section 6.3), this will lead to

continuous improvements in efficiency and encourage the adoption of greener solvents and reagents.

6.2.2 Predicting Reaction Outcomes

Compared with reaction optimization, prediction of chemical reactivity is an area with little precedent. This problem is fundamentally more challenging because of the many ways that two or more molecules can interact. Consider the human approach to predicting reactivity:

1. Identify the reactive sites (atoms) on the reactants
2. Determine possible types of reactions that could occur
3. Evaluate the possibility and extent of a certain reaction

AI systems that follow this approach generally utilize reaction templates described below. Other approaches may not separate these tasks explicitly and are categorized as non-template methods, described subsequently.

Template based: The choice of molecular representation (see Section 6.2.4) will both affect and depend on the approach to generating and evaluating candidate transformations. Graph-based representations are widely used in both rule-based and ML approaches to compare the molecule against reaction templates and subsequently identify applicable reactions and the reaction center (see Figure 6.1). One approach pioneered by Coley and co-workers[21,22] applied 1700 templates extracted from the US patent database, and then used a neural network to rank the likelihood of products generated. This demonstrated the feasibility of ML for forward prediction; however, the method suffered from the high computational cost of, for each template, determining if the molecular graph contains a subgraph isomorphic to the template. To mitigate this, both Segler[23] and Coley[24] used a neural network to suggest a template, rather than comparing directly to each one.

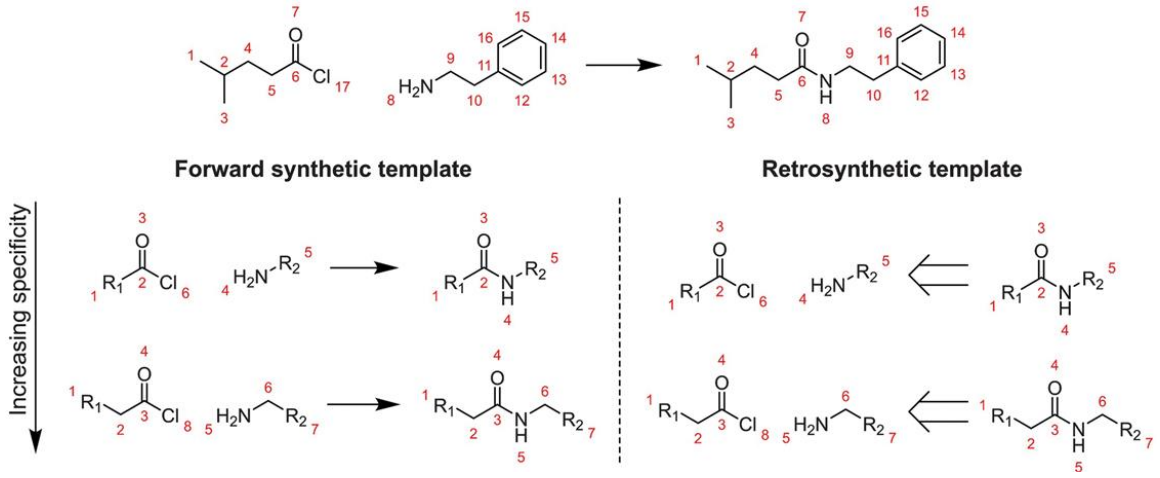


Figure 6.1 Application of reaction templates in the forward and reverse direction, with increasing numbers of adjacent bonds included. Reproduced from ref. 22, https://doi.org/10.1021/acs.accounts.8b00087, with permission from American Chemical Society, Copyright 2018.

Template-based approaches include fundamental tradeoffs between coverage, accuracy, and speed. This is evident when specifying the complexity of the reaction core. Templates which only contain the bond-forming/bond-breaking atoms lack the crucial context of electronic and steric effects elsewhere in the molecule. However, more complex templates that include adjacent fragments will take longer to compute and may over-specify a transformation. Another tradeoff is the number of templates used. It would not be possible to consider every transformation; therefore, models are usually limited to examining hundreds or thousands of templates. Often this parameter is expressed as the cumulative percentage of reactions from which templates were learned. Unfortunately, it seems clear that even good coverage of well-known reactions (*e.g.*, including templates for 99% of the reactions in Reaxys) is not sufficient for complex retrosynthetic analysis across broader organic chemistry. Analysis of the "network of organic chemistry" performed by Szymkuć found a power law relationship between the number of times a reaction is found in the literature and its ranking based on that frequency.[25] Such a relationship indicates that a few outliers (*i.e.*, rarely used reactions) can have a significant impact on the overall statistics; in this case, the network of known organic chemical reactions. Szymkuć contends that this requires the enumeration of many thousands of

chemical rules (∼80 000 for Chematica program),[26] something that proves challenging for both machine-extracted rules and human-encoded ones.

Template-free: In contrast to direct template matching, one could compute directly on the molecular SMILES string, treating the reaction as a translation task between products and reactants (see Figure 6.2). This approach, which uses an encoder-decoder model called sequence-to-sequence (seq2seq), was explored separately by Nam,[27] Liu,[28] and Schwaller.[29] A few surprising results stood out. First, the models were able to generate valid SMILES strings of products as outputs a majority of the time, without explicit specification or filtering. Second, the models showed a large performance gap between forward and retro prediction. Liu's model, which operated in the reverse (retro) direction, was not able to outperform Coley's template-based model. However, the best overall performance was reported for Schwaller's seq2seq model, which predicted reaction products in the forward direction and included an attention layer. Attention models, which allow the algorithm to focus on particular parts of the input, have the added benefit of providing some explainability, something that template free-models normally lack compared with template-based ones. The impact of including attention also suggests that model architecture is an important area for future investigation. Indeed, Yang[30] showed that in addition to the representation used, the neural network architecture is of critical importance to the property prediction task.

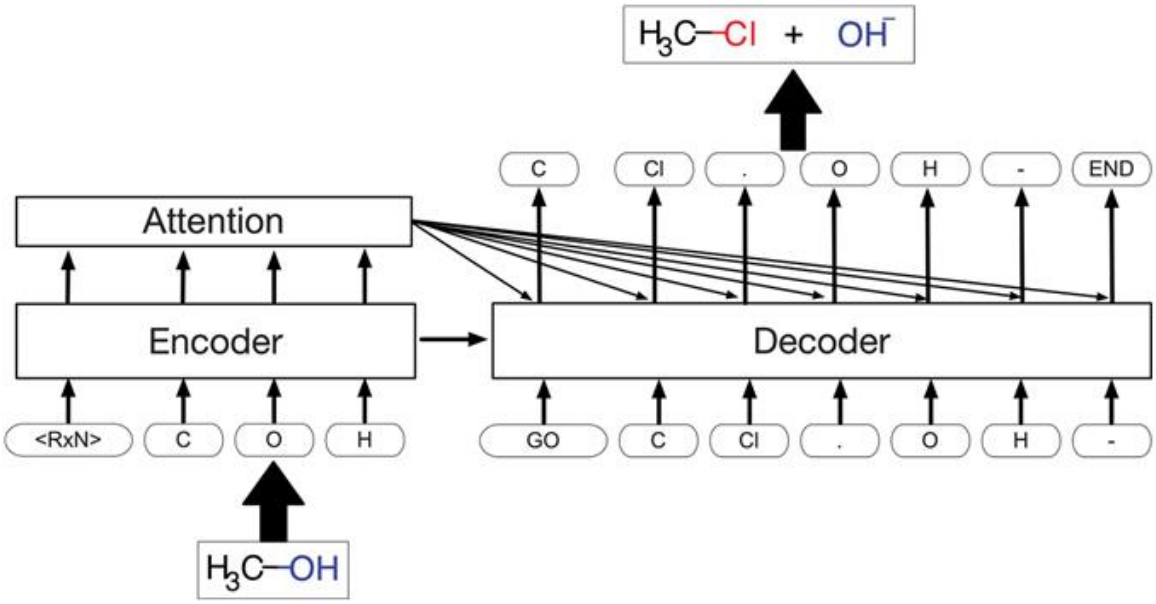


Figure 6.2 Attention-based seq2seq model as implemented by Pande.[28] The input is a SMILES string of a molecule, here methanol, which is passed to sequential recurrent neural networks, an encoder and decoder, which output a SMILES string of the predicted reagents. The attention mechanism allows the model to focus on particular characters (usually atoms) in the input. Reproduced from ref. 28, https://doi.org/10.1021/acscentsci.7b00303, with permission from American Chemical Society, Copyright 2017.

6.2.3 Performing Retrosynthesis

The ultimate goal for CASP is to reliably produce multi-reaction synthesis plans for any molecule, known or novel. For a human chemist, this process is ostensibly considered in the retrosynthetic direction, from a target toward buyable starting materials. But given that reactions are taught (and occur) in the forward direction, the chemist is necessarily performing a holistic evaluation of pathways that simultaneously considers both the value of a certain reaction for building the target of interest and the likelihood it will yield the desired product. It is challenging to create algorithms for such a nuanced task, and though seq2seq suggests that there could be important distinctions, the differences in the predictive power of algorithms performing the forward *versus* retro tasks have not been thoroughly explored.[31]

The large number of reactions available to make each molecule creates an exponential explosion of possible routes at each step. Consider the tree structure whose root is a target molecule and leaves are buyable substrates. The average number of different five-reaction syntheses from a target can be 10^{16} by some estimates.[25] Therefore, to predict a retrosynthetic route for a target, ML algorithms are needed to identify and navigate promising portions of these tree structures without needing to exhaustively explore them. Such an algorithm must be able to determine and prioritize transformations which lead to synthesizable molecules and ultimately buyable starting materials. The criteria by which a model makes such prioritizations are referred to as a cost or reward, depending on whether the quantity should be minimized or maximized. In general, the notion of cost or reward can include many factors that one wants to optimize, such as monetary costs of starting materials, reaction yields, time of reaction, environmental factors, or reaction complexity.

Describing complexity is deceptively challenging. Using an intuitive relationship—more complex molecules are more expensive to make—one could simply consider the differences in length of the SMILES strings for reactants and products as a proxy for the cost of a reaction.[25] Another approach is to consider structural features, assigning each a predetermined complexity cost. However, molecular complexity, as perceived by humans, is an unreliable metric for navigating chemical space. Greedy application of complexity heuristics one might use for evaluating a transformation may lead to synthetic shortcuts being overlooked. A perfect example of this is the use of protecting groups. Their installation increases the number of bonds and atoms but only transiently; their effective use should be seen as a decrease in cost by a well-performing model. Many of the most elegant syntheses in organic chemistry contain transformations that initially appear unintuitive, inefficient or counterproductive, but ultimately set up highly efficient reactions.

A measure of complexity is necessary to impart expert-level chemical intuition on retrosynthetic algorithms. Conveniently, Coley was able to leverage the existing machine-coded reaction dataset to develop SCScore, a prediction of the number of reactions required (some of which may temporarily increase bond or atom counts) to synthesize any molecule from

starting materials.[32] By learning a holistic notion of synthesizability from data, the SCScore metric is useful to not just guide retrosynthetic analysis, but also to prioritize drug candidates for synthesizability or identify complexifying reactions for chemical library construction.

While SCScore provides a useful heuristic, it does not specify a policy for choosing which transformations will make up the optimal synthetic route. This can be achieved through reinforcement learning. This approach learns an optimal decision-making policy that will accept short-term loss in order to maximize long-term reward. Following training, the policy may be executed to determine the optimal route along a retrosynthetic tree. Segler[23] demonstrated the effectiveness of reinforcement learning for retrosynthesis using the Monte Carlo Tree Search (MCTS) method, made famous by the success of DeepMind's AlphaGo algorithm in learning the complex game Go.[33] Briefly, their implementation learns a decision policy by using MCTS (see Figure 6.3) to perform simulated “playouts” of a small subset of possible routes within the full reaction trees of various targets, records the ultimate outcome (success or failure in reaching buyable starting materials), and notes it for each reaction in the branch which was played out. The learning process balances between playouts of the most successful routes and exploration of unseen ones, and results in a policy that may be used to successfully predict valid retrosyntheses of molecules not seen during training.

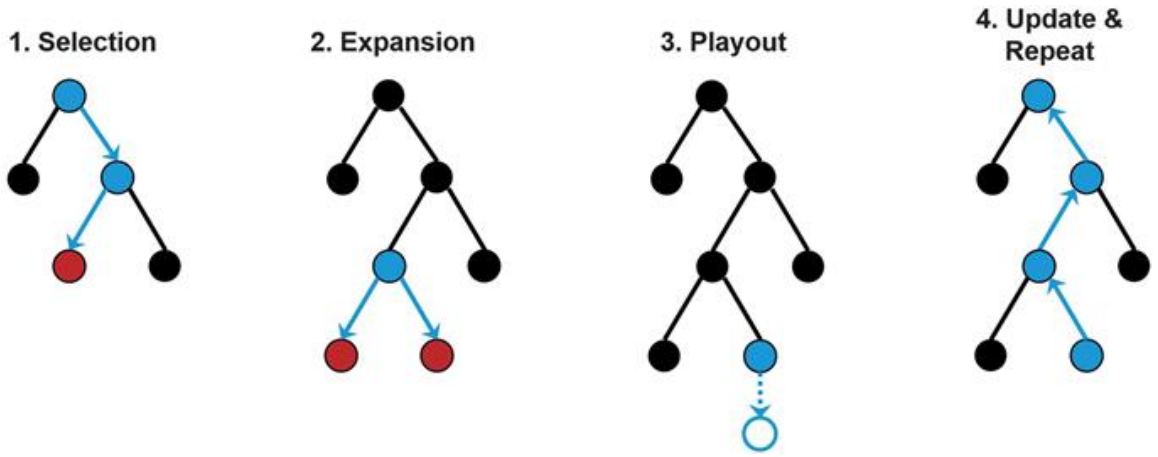


Figure 6.3 Monte Carlo Tree Search. For retrosynthesis, nodes represent compounds and lines represent reactions. Selection finds a promising node that has yet to be explored. Expansion is applying templates or some other strategy to generate candidate precursors. The playout uses retrosynthetic steps to try and reach starting materials. The success or failure to do so is determined and propagated up through the parent nodes and the process is repeated.[23]

Schreck[34] published a similar method for learning policies using deep reinforcement learning (see Figure 6.4). In comparison with the greedy application of a SMILES-based complexity heuristic, both Waller[23] and Schreck achieved significant improvement in the number of molecules for which a full retrosynthetic path could be found in reasonable time.

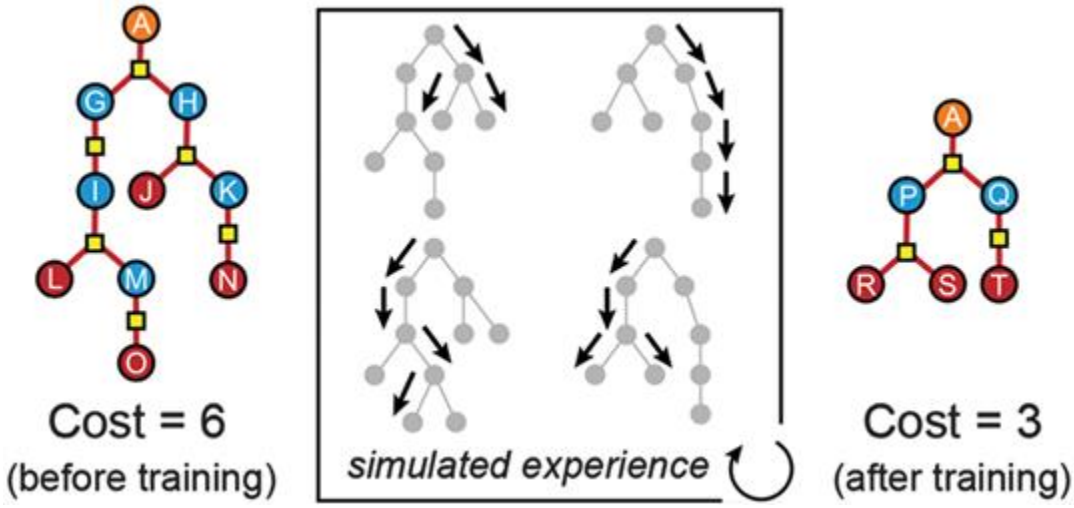


Figure 6.4 Reinforcement learning for determining cost functions, as implemented by Bishop[34] Reproduced from ref. 34, https://doi.org/10.1021/acscentsci.9b00055, with permission from American Chemical Society, Copyright 2019.

Ultimately, retrosynthetic algorithms must consider more information to provide a real-world estimate of synthetic cost including imperfect reaction yields and the cost of purification. It is even more challenging to estimate costs that depend on the sequence of the reactions, such as switching solvents between reactions.

6.2.4 Assessment

The elements of a CASP system—reaction optimization, forward prediction, retrosynthesis—can each be addressed as separate problems. But, the Make-It program revealed how interconnected each aspect is. To deliver a full end-to-end solution, the retrosynthetic algorithm must consider how many templates are used to define the branching number and determine cost, based on the solvents and reagents needed. It remains uncertain how much these tasks should even be dictated. Research on seq2seq models suggests that properly specified ML models can, in some cases, perform these tasks in a concerted manner, perhaps at the expense of explainability.

Assessing the performance and value of a model is also not a straightforward task. Ground truth within reaction databases is frequently used to determine a "top 1" or "top 5" prediction accuracy for forward reactions (*i.e.*, the fraction of time that the algorithm predicted the correct product within its top X results). However, there is no comparable metric for multistep retrosyntheses. Generated routes can be compared with published ones, but it is expected that they will differ based on small parametric differences in the model, or even offer improved routes to those known. Attempts have been made to baseline against human experts.[23] One could envision a "chemical Turing test" wherein passing means that chemists cannot distinguish between a route generated by a machine or by legendary synthetic chemist E.J. Corey. But this may be biased toward reactions which chemists are comfortable with, rather than those which are most efficient. The ultimate validation is carrying out the proposed synthesis, which is slow and expensive. New ways to test novel routes through rapid experimentation or *in silico* are needed to provide feedback for these systems.

6.2.5 Adoption

The Make-It program aims for a push-button system that will evaluate any molecule, tell the user if and how it can be synthetized, and, provided the proper starting materials, produce it autonomously. However, reaching that vision, or even making progress toward it, requires that these tools have demonstrable uses in the near term. Make-It technologies are already being adopted by partners in industry and government to address real challenges.

An early success from the program was Chematica software developed by Grzybowski and coworkers.[35] This rule-based expert retrosynthesis system was challenged by MilliporeSigma (formerly Sigma-Aldrich) to provide routes to targets which for which previous synthesis attempts had failed. Chematica provided routes to all six targets that MilliporeSigma chemists were able to successfully perform within the constraints of the challenge. MilliporeSigma subsequently purchased Chematica and is now marketing the software under the name Synthia™ and using it internally to expand their product catalog.

In all cases, user experience is critical to adoption; a system is only as useful as it is usable. Usability involves not just the user interface design (*e.g.*, legible reaction schemes, easy stereochemical inputs) but also the ability to prioritize the most useful information for display. For instance, early in the development of CASP systems in Make-It, the search output would often highly rank very similar results, essentially offering the same route over and over again with minor variance. Later iterations of the software were designed to provide diverse routes and group similar ones when presenting them to the user. Another practical consideration is the computational requirements of such systems. Even for CASP systems that employ expensive template matching and large libraries of transformations, a single retrosynthetic step can usually be performed in a fraction of the time it would take a human. However, the bottleneck for system performance is the explosion of possible routes, requiring large amounts of memory to store. In systems developed under Make-It, routes to reasonably complex drug-like molecules can be computed in 6–60 seconds on a multi-core desktop computer.[24]

One might characterize current CASP software as an early version of a "synthesis search engine." Just as Google and Bing have become necessities for navigating the vastness of the internet, CASP enables chemists to similarly navigate reactivity space. All the systems described above allow chemists to easily link to source materials, saving countless hours of gathering and reviewing papers, pouring over the literature for a key reference. A reaction, which might be common in one field, is rare in another, such that it takes a discipline-agnostic software to suggest to a chemist to use it. We predict that current CASP systems will continue to

increase in complexity, applicability, and performance. Today's systems are a stepping stone toward AI-powered chemical awareness.

Finally, it should be noted that these algorithms can be adapted to co-examine multiple targets, identifying a minimal set of starting materials for a library of compounds. They can even be parallelized to perform a meta-analysis of the entire reaction corpus to guide reaction development toward the most impactful new transformations. For a large organization like the DoD, which purchases a diverse range of chemicals, such tools could identify common building blocks and intermediate compounds in order to streamline production and reduce costs.

6.3 Data Challenges for Statistical Learning in Chemistry

Quantitative structure–activity relationships (QSAR) were first reported in 1961, launching the field of cheminformatics.[36] Early work relied mostly on correlating physical properties such as logP with a biological outcome. Eventually, fragment-based and quantum mechanical descriptors started to be incorporated and in the early 2000s, neural networks were used for property prediction.[37] In 2014, the U.S. National Institutes of Health (NIH), Environmental Protection Agency (EPA), and Food and Drug Administration (FDA) sponsored Tox21, a challenge to predict the performance of thousands of molecules in toxicological assays. The entries showcased a wide range of ML techniques for prediction of toxicity, including random forests, deep neural nets, support vector machines, and others.[38]

Despite the popularity of statistical methods for cheminformatic modeling, however, ML was not embraced by synthetic chemists. The Make-It program, launched in 2015, attempted to leverage many of the underlying QSAR techniques to retrosynthesis and forward prediction. Doing so illuminated several challenges that will need to be overcome for complex tasks such as property prediction and molecular design. While in some sense these challenges are universal across ML applications, they have aspects unique to experimental sciences and chemistry in particular.

This section will focus on the challenges of integrating chemical data to statistical learning systems.

ML algorithms thrive in domains where data are abundant, or at least easy to produce, and relevant to the task at hand. Advances in computer vision (CV) and natural language processing (NLP)—prototypical examples of ML's success—are aided tremendously by the ubiquity of digital data online and the efforts of a community that has been focused in these areas for decades. Such data are produced by the public at no cost and carefully curated and annotated by the research community into standard, high-quality collections to further model development and provide common references and benchmarks (*e.g.* ImageNet,[39] CoNLL Shared Tasks,[40]*etc.*).

This is in stark contrast to the chemical sciences, wherein data are comparatively expensive to generate and often treated as valuable intellectual property that should be safeguarded rather than shared. Data published in academic journals are publically available but must be aggregated to provide sufficient training data typical for ML. These composite datasets are (1) biased, (2) inaccessible, and (3) narrow, challenges which will be discussed in that order.

6.3.1 Data are Biased Toward Successful Experiments

Journals prioritize confirmatory studies in which a hypothesis was supported by the experimental data. A project for which the experimental data are not explained by the hypothesis is deemed a failure, forever relegated to a lab notebook. This model has drawn criticism for incentivizing data and statistical manipulation.[41] But a widespread and arguably more serious effect is the bias it generates in scientific databases toward positive results (see Figure 6.5). With few exceptions, chemical reaction databases contain only successful reactions, *i.e.*, reactions which have significant yield of the reported product. Negative results, such as no reaction, or an unwanted one, are seldom reported. This makes using supervised learning classifiers difficult as these require explicit positive and negative labels for each class.

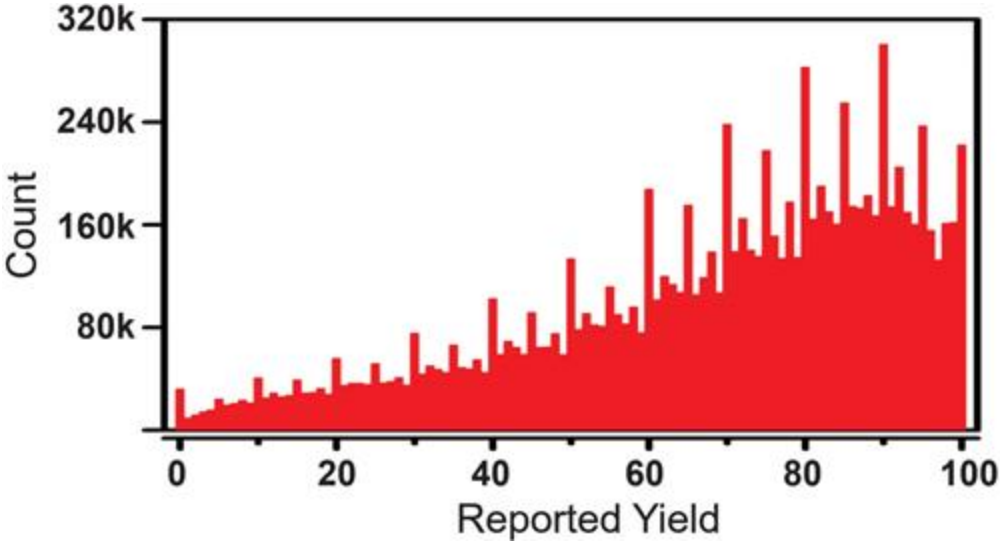


Figure 6.5 Aggregate yields reported in Reaxys. Figure courtesy of K. Jensen, presented in part at the Make-It Industry Forum, October 2019.

This challenge was encountered in the Make-It program. Coley trained supervised models on the Reaxys database to predict the reactants yielding a target product (see Section 6.1). Without examples of failed reactions or side products in the training data, however, the models were not able to classify reactions as plausible. Both Segler[23] and Coley[21] overcame this by generating "synthetic" negative examples to provide additional class labels. Coley identified reactions that had over 50% yield and surmised that no other products (including expected ones) could be present in the product mixture. Waller applied the same strategy and further supplemented the negative data pool by randomly sampling amongst scrambled reactant-product pairs to give 70 million false pairings. This enabled reasonably good performance to be achieved on the Reaxys data; however, this should be considered something of a stopgap measure. Most valuable to the model are the reactions which would be expected (by some human heuristic) to give one product, but instead gave another. Fortunately, these occur all the time in the course of chemical research; unfortunately, they are rarely published or otherwise disseminated.

6.3.2 Existing Data are Unreliable and Non-standardized

As cheminformatics gained popularity, databases emerged containing some combination of formulae, structures, physical properties, and biological activities. These data are routinely used to train and validate QSAR models for a variety of drug discovery and materials applications. In addition to the biases described above, the data are likely to suffer from a more fundamental problem. A recent study audited just the structures in six large databases and found that up to 3.4% of entries contained an inaccurate structure, such as a misplaced chlorine atom or swapped hydroxyl and methoxy.[42] The authors found that this level of error had significant impact on the accuracy of models trained on the data. Yet publications often report model performance without any experimental validation—just train:test splits that would fail to uncover systematic issues with the data.

If databases are unreliable, one might look to scientific journals to obtain primary source data. Chemistry literature is a particularly appealing source of knowledge for ML because, unlike reaction databases, it occasionally describes reactions and conditions that were ineffective in addition to those that were successful. To the extent that negative results are disclosed in the chemical literature, it is usually *via* discussion in the text, which in turn may reference figures and tables which are necessary to fully understand the specifics. Humans are able to understand this information readily, but the nuances of language and unconstrained diversity of figures and tables inhibit automated interpretation by machines. Though significant progress has been made in NLP and CV techniques in general, relatively little effort has been expended applying these techniques to the chemistry domain. As a result, most publicly available NLP and CV models, having been trained on more general domains, struggle when applied to chemistry data's unique vocabulary and syntax.

Finally, one must fundamentally trust the published data. Though publications may be peer reviewed, most experiments are not reproduced by other laboratories because tenure-seeking faculty are incentivized to demonstrate and publish novel discoveries. Even when that is not the case, an experiment may be too expensive or complex for another lab to reproduce, or the detailed procedure is not made available. As evidenced by Figure 6.5, which shows disproportionate numbers of experiments reporting yields rounded to the nearest 10, it is clear that published results do not

always reflect accurate reporting of real experimental data. How and where this impacts the viability of models and subsequent predictions should be considered very carefully.

6.3.3 Data are Relevant to a Narrow Collection of Tasks

A fundamental assumption of supervised learning, though seldom met in practice, is that the data on which a model is trained are from the same distribution as the data on which the model will be tested. Effectively extrapolating to data that are dissimilar to the training data, though aided by such methods as encoding first principles and continual learning, is still an open research topic in the machine learning community.[43] In chemistry, however, the most valuable predictions are those involving molecules, reactions, or properties for which little experimental data exist from the distribution of interest (*i.e.*, predictions outside the training set). Therein lies the challenge.

The highly competitive nature of pharmaceutical research, while important for innovation, generates large caches of proprietary data, much of which is never used but also not released for fear of enabling a lucrative discovery by a competitor. But even if this could be overcome (*vida infra*), the data would reflect the large imbalance between aggregate pharmaceutical R&D and other chemical industries—a disparity that extends to publically available data as well. In 2018, the R&D budget for the top 15 pharmaceutical firms was $100 billion,[44] compared with just $11 billion[45] for the top 15 chemical firms in other sectors. This naturally leads to databases oriented toward the pharmaceutical industry and its applications. Virshup[46] illustrated this by performing an analysis of the popular PubChem and ZINC databases. First, the entirety of the drug-like small molecule universe (estimated $\sim 10^{60}$ molecules, originally described by Lipinsky[47]) was mapped uniformly to 90 thousand "bins," or neurons, using a self-organizing map according to Tanimoto similarity of their fingerprints. It was then shown that 98% of PubChem compounds were assigned to just 2% of the neurons, mostly characterized by compounds with low molecular weights and few rings. Various conclusions can be drawn from this result, but the facts are the following: (1) much of the small

molecule universe is unexplored, and (2) that which has been explored is not representative of the whole.

This data disparity extends to the property space as well. Wu *et al.*[48] recently aggregated a collection of property datasets to establish benchmarks for various property prediction tasks. While some datasets contain property information for many compounds (for example, PubChem BioAssay (PCBA) has biological activity of over 400 000 compounds), data on other properties are less abundant (lipophilicity values on 4200 curated compounds from ChEMBL). Data quantity is not a sufficient condition for model quality; however, in many cases, it is a necessary one, particularly for complex deep learning models with many parameters and when the generation of new data cannot be controlled by the model's developer. Figure 6.6 illustrates the performance of several different models based on training set size.

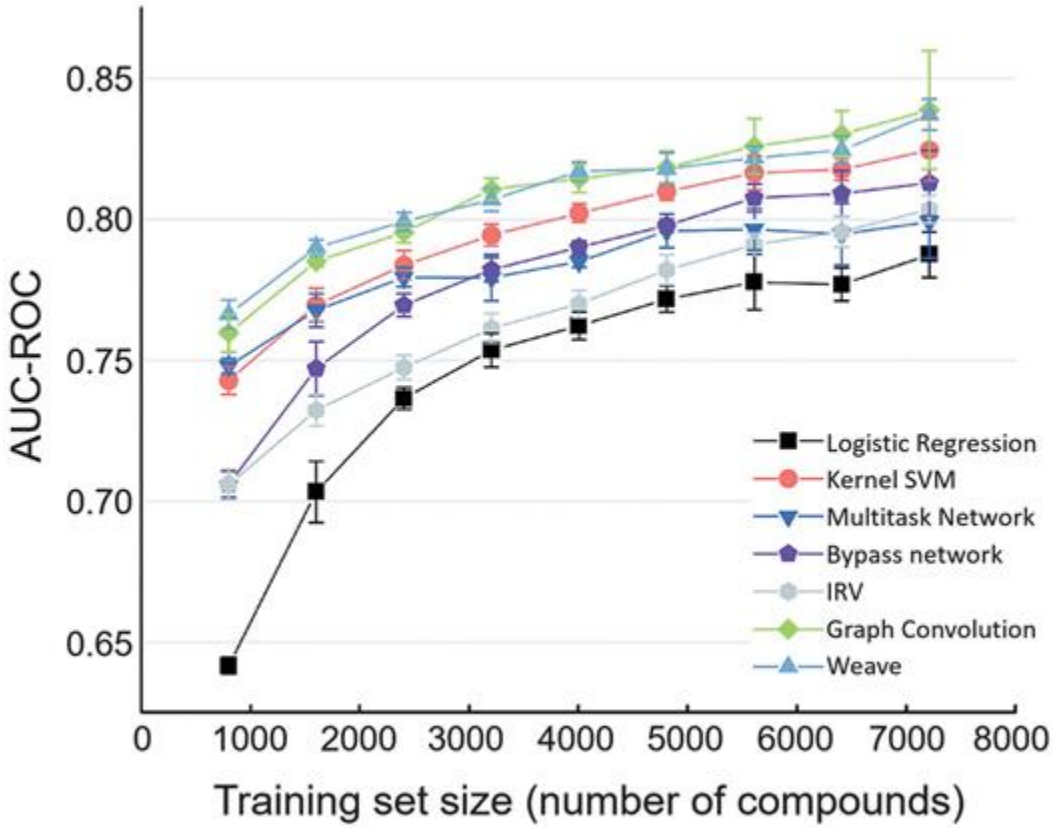


Figure 6.6 Importance of large datasets and the performance variation between several different models for property prediction. Adapted from ref. 48 with permission from the Royal

Society of Chemistry.

This incongruence between the data available and the tasks attempted suggests that, without generating task-specific data, ML will be heavily reliant on exploiting existing knowledge rather than exploring new chemical space, and most successes for ML in chemistry will be restricted to these areas in which the community is already heavily invested. This is particularly challenging for properties of specific DoD interest, as the community of researchers that produces such data is so small.

6.3.4 There Is No Standard Representation of Chemistry Data for Machine Learning Models

Even supposing that the ideal data for an ML task in chemistry can be acquired, the format in which it should be presented to the model is not obvious. Until recently, NLP and CV relied heavily on manual feature engineering as a preprocessing step to transform the raw data into a form that would be ingested by the model. The standard practice was computing multiple features such as the context-free grammar parse of a sentence, or the scale-invariant feature transform of an image and then using feature selection techniques to determine which features to include.

Though still a component of ML learning pipelines, the importance of feature engineering was diminished with the advent of deep neural networks. Many current deep neural network models instead take some more fundamental representation of data as an input, such as a one-hot vector for each word in a corpus or a convolution of neighboring pixels in an image, from which successively more salient features are automatically derived as part of the model's training process. The advantage of this technique is that, loosely speaking, these representations are more information complete than engineered features, in that they contain essentially the same information from the raw data that an oracle (in this case, a person) would use to make a decision.

Chemistry's oracle is nature, having sufficient information to resolve the time-dependent Schrödinger equation for multi-electron systems to arbitrary precision. It is impossible to provide all such information to an ML model in a classical computer,[49] so the question becomes: what is the appropriate

abstraction of the information nature has available, and how should this abstraction be represented? There is almost certainly no single answer to this question; the appropriate representation will vary depending on the chemistries involved and the ML task attempted.

Molecular representations typically fall into two broad categories: fixed representations, analogous to the engineered features described previously which do not vary depending on the task (*e.g.* molecular descriptors and fingerprints), and learned representations which are developed in the context of the model's task.[50] Examples of each are shown in Figure 6.7.

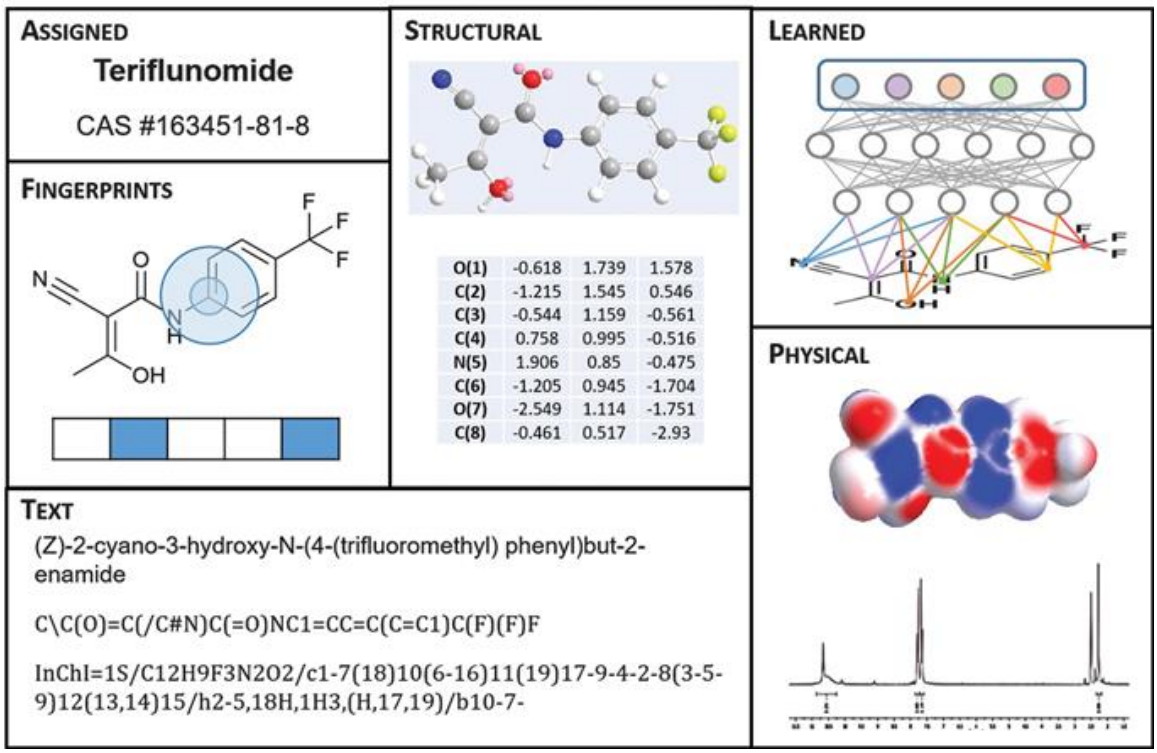


Figure 6.7 Examples of different molecular representations.

An example of a fixed representation is the two-dimensional, atom-bond skeletal formula familiar to many people. It corresponds directly to a molecule's atom and bond arrangement and can be interpreted as an undirected graph. A text-based representation, which is familiar to human chemists but perhaps less interpretable, is SMILES, which are alpha-numeric strings that represent the graph-connectivity of a molecule.

Molecules may be represented by expert-defined descriptors, hundreds of which can be calculated by the popular Dragon software.[51] Extended connectivity fingerprints[52] are determined by a hashing function that incorporates local topological information of the molecule. One attractive feature of these is that they are easy and quick to generate using off-the-shelf tools such as RDKit and easy to customize for particular applications. Waller's retrosynthetic algorithm (see Section 6.2) used fingerprints to suggest transformations that would be applicable to a given target. Such fixed representations have long been used, in various forms, in many QSAR approaches.[50]

Learned representations in chemistry are a relatively recent advance and are typically produced by operating on the graph structure of the molecule. Duvenaud[53] developed the first such implementation using convolutions over arbitrary molecular graph structures for various property prediction tasks. A similar approach was adopted by Coley for the forward reaction prediction task in the Make-It program.[54]

The literature is not consistent on whether fixed or learned representations are preferred for prediction tasks in general. In predicting binary assay outcomes that indicate whether a certain compound, for example, binds to a specific receptor, inhibits some pathway or induces toxic effects, Wu finds that learned representations tend to be more successful,[48] while Mayr suggests that fixed representations perform better.[55] Yang suggests that the discrepancy likely arises from differences in evaluation protocols and further demonstrates the benefits of hybrid representations that combine convolutions and descriptors.[30] The advantages offered by a hybrid representation illustrate a broader point: different application spaces lend themselves to fundamentally different representations. Much like the datasets discussed above, work in this area has been steered toward the prediction of therapeutic properties, and the particulars of what fixed features to include or how to produce learned features will likely vary when applied to predicting different properties such as vapor pressure or heat of formation. One focus of the DARPA AMD program is developing and evaluating molecular representation across varied application spaces to provide better understanding of these questions.

6.4 Realizable Solutions to Data Challenges

Data are the lifeblood of an AI-powered laboratory. In Section 6.3, we discussed how ensuring the objectivity, accessibility, and relevance of data are the most significant constraints on model performance when applying ML to chemistry. The particulars of how to address these issues depend on the source of the data, be it from one's own lab, external sources (*e.g.*, databases or other labs), simulations, or extracted from the scientific literature. Here, we highlight several solutions various researchers and organizations have implemented to address these data shortfalls.

6.4.1 Automation and Standardization of Experimental Data Collection

A chemist has the most control over the experiments performed in their own laboratory, with their own equipment, and under their own protocols. It is therefore the most natural place to effect change in the way data are generated. We present here three objectives for laboratory workflow which are fundamental to improving data utility, objectivity, and accessibility.

Objective 1: Recording Data in a Machine-Readable Format The push for digitization of data and experimental protocols is not new. Electronic lab notebooks (ELNs) initially appeared in the 1990s.[56] However, the adoption of such tools has been slow for a myriad of reasons,[57] including concerns about the cost of software, data security, and legal issues. Usability, however, remains the largest challenge. Chemists may struggle to enter the nuances of a chemical reaction scheme[58] or be forced to adopt an unfamiliar organizational scheme. ELNs must strike a balance between being generalized enough to be used by different types of chemists but powerful enough to account for the nuances of each field.

Attempts to develop a universal ELN continue. But, lamentably, even if a highly usable, widely applicable system were available, there is no guarantee it would be adopted by researchers who may have entrenched preferences and habits. It may ultimately be more successful to recast the ELN as an auxiliary observer, automatically

documenting an experimental procedure and recording the data. The Ley[59] lab has explored the use of computer vision for monitoring experiments through video and natural language processing for ingesting written text and converting it into standardized, machine-readable formats.[60] Similar approaches rely on integrating analytical instrumentation into digital repositories. However, the APIs provided by manufacturers of such instrumentation often produce data in proprietary formats different for each instrument. Exporting the data to open source formats for preservation or integration into statistical models is time-consuming and may not maintain the original data resolution. Pressure on manufacturers from the research community to be more open with their APIs would likely require a community-wide effort combined with stiff industry competition.

A program called ASPIRE at the NIH National Center for Translational Science (NCATS), which seeks to combat opioid abuse and find safer options for pain management, includes a specific challenge to develop "next-generation, open source electronic laboratory notebook[s] that would serve as an electronic synthetic chemistry knowledge portal and allow for … unbiased data collection during the synthesis planning, execution and analysis."[7] Designs will be selected which integrate with ML models and analytical tools to provide a complete chemical discovery workflow.

Objective 2: Standardized, Algorithmically Generated Experimental Protocols. Capturing all possible details about an experiment is one way to ensure faithful, reproducible data. Another is to create protocols and experimental plans that enforce standardized execution. The University of Washington's (UW) Biofabrication Center[61] launched the software Aquarium which monitors sample inventory and specifies preparations but leaves manual operations and complex observations to be performed by a laboratory technician.[62] This enforces not just standardized data collection but also standardized experimental design, which will help increase the accessibility to ML models. The UW effort is part of the DARPA Synergistic Discovery and Design (SD2) program, which

takes a data-driven approach toward model building across a wide variety of domains.[63]

Objective 3: Hands-free Execution of Experiments: Robotic experimentation offers cost savings and safety improvements, but an additional driver of laboratory automation is the ability to generate unbiased, fully documented datasets. Eliminating the manual execution of more tedious tasks such as pipetting, heating, mixing and instead relying on automated tools allows more experiments to be conducted, more data to be acquired, and also improves the reproducibility and specificity of that data. Large-scale realization of laboratory automation is exemplified by the development of "cloud labs," which are quickly becoming viable options for biological experimentation, where robotic automation has had a large head start *versus* chemistry. One example is Transcriptic, which allows for biological assays to be run without a human in the loop.[64] Another is Emerald Cloud Lab, which offers all the capabilities of an organic synthesis lab, including solid handling, rotovapping, purification and many different analytical capabilities.[65]

The first two objectives align with a larger goal for scientific data generation and stewardship, which is adoption of the FAIR (Findable, Accessible, Interoperable, Reusable) Data Principles by data producers and publishers.[66] While not an objective in and of itself, the FAIR principles are recognized as a necessary condition for innovation and discovery using AI in chemistry.[67]

6.4.2 Sharing Data Across Laboratories

An exciting aspect of lab automation is the ability to decentralize research. This approach requires rapid, clear communication between data scientists and experimental labs. These collaborations are commonplace today but are usually *ad hoc*, 1:1 partnerships that operate *via* "data dumps" from one group to another. This occurs even within research groups, where a graduating student leaves poorly annotated and disorganized data. In either case, the data scientist or model builder must spend a great deal of time cleaning the data, a process that may introduce errors in interpretation. In

the future, we expect data sharing to move from small specific collaborations to massive, public datasets that are continuously added to and utilized by many groups. Publicly available datasets such as PubChem or BindingDB, as well as commercial ones such as CAS and Reaxys, continue to grow,[68] but problems with data format and accuracy create new bottlenecks.

An alternative to these resources is data produced directly from research groups in conjunction with a publication, something that some journals now require using consistent data standards.[66] The best path toward FAIR data in chemistry will be a combination of these approaches: decentralized release of data from many data producers, in consistent and useful formats, combined with the data management tools of centralized databases. Though the adoption of FAIR principles has many proponents, adherence is not yet common practice in the community.

Another early step toward making existing data more FAIR is Google's Dataset Search.[69–71] This search engine allows users to search for datasets that might otherwise be difficult to find, as most search engines will not index the internal records in a hosted database. For now, the tool requires that data providers annotate their datasets with an open metadata format, and it does not provide a way to access the data other than linking to the host site. In the future, this or other tools may integrate an API that allows models to directly utilize multiple datasets and even automatically search for them.

Ultimately, however, even combined datasets from academic and government laboratories may not match the scale of those held by companies and organizations that are unable to share their data for competitive or privacy reasons. A number of workarounds have emerged to harness this propriety data, ranging from privately funded “open innovation” consortia, to tightly controlled partnerships.[72] One potentially scalable model is that of Accelerating Therapeutics for Opportunities in Medicine (ATOM) consortium, which relies on Lawrence Livermore National Laboratory (LLNL) as a trusted intermediary to aggregate, manage, and securely analyze data provided by members from the pharmaceutical industry.[73] This set-up allows members to benefit from both shared data as well as the uniquely powerful computing capabilities and

expertise provided by LLNL. At MIT, an alternative framework is employed by the Machine Learning for Pharmaceutical Discovery and Synthesis Consortium. Here, the ML models are developed jointly but are deployed in-house to be trained separately on companies' proprietary data.[74,75] This allows for collaborative development of tools and models but does not benefit from pooled datasets.

A different approach to utilizing non-public data is altering it to conceal the specifics but still inform ML models. Homomorphic encryption[76,77] allows computation without ever decrypting the data or having access to the decryption keys. Another example is data obfuscation.[78] Here, data are augmented with new samples or random noise in such a way that effective ML models can be trained on the data, but sensitive information related to individual samples or groups of samples obscured. Further research is needed in this area to make these techniques more robust, easy to implement, and better-performing.

6.4.3 Leveraging Experiments Described in Scientific Literature

Compared with large, structured databases, chemical literature contains disorganized and unstructured data. Even for data explicitly stated in tables and figures, the formatting and structural representations are not easy for machines to interpret. Olivetti and McCallum have demonstrated two promising approaches for accessing knowledge within materials literature sources. In 2017 the group published an unsupervised technique to extract synthesis parameters for 30 different oxide systems from a corpus of 640 000 journal articles.[79] More recently, they constructed a dataset of 230 synthetic procedures annotated by domain experts with labeled graphs that express the semantics of the synthesis sentences, which may be used for shallow annotation of unseen text.[80] These techniques circumvent the major hurdle of manually annotating large datasets that are commonly encountered when applying NLP methods to novel domains. The Optical Structure Recognition Application (OSRA)[81] is one of the better-known systems for classifying images of molecular structures, though it assumes that the images have already been extracted from the articles. Staker[82] recently developed end-to-end deep learning solutions for both segmenting

molecular structures from documents and predicting chemical structures from the segmented images.

The community is also working to make the machine-readability of journals an intrinsic design goal by developing easily accessible metadata at the time of publication.[83] It is currently focused on enhancing discovery by search engines, but if it was adopted broadly and extended to describing text and figures, this could obviate the need for sophisticated NLP and CV extraction techniques.

6.4.4 Generating Data Through Simulation

Some data are impractical to gather experimentally, either because of cost or the limits of analytical equipment, and are therefore more efficient to acquire through molecular dynamics (*e.g.*, a model predicting binding affinity[84]) or quantum mechanical simulations such as density functional theory (DFT). However, the time and computational resources required for such simulations present another opportunity to leverage ML. Gilmer[85] developed an ML model, trained on the QM9 database, which was able to approximate quantum mechanical quantities such as HOMO and LUMO values with comparable accuracy to DFT in hundredths of a second – over 300 000 times faster than simulation-based DFT methods. A directed-bonds variant of the message-passing neural network architecture developed by Gilmer was shown by Yang[30] to be applicable to a wide range of property prediction tasks. Not only are these tools valuable themselves, but the rate at which these estimates are computed makes it feasible to provide accurate approximations to downstream models.

ML predictions were used to prioritize further simulation and experimentation by Gómez-Bombarelli for the discovery of efficient molecular organic light-emitting diodes.[86] Simple fragments were combined in a donor–acceptor configuration to create 1.6 million structures —far too large a pool to screen with quantum simulation. Therefore, molecular fingerprints were generated and analyzed by a neural network to prioritize candidates for quantum calculations. Initially, the model was trained on calculated values for 40 000 randomly selected structures. To improve the model for prediction of the most efficient structures, it was periodically retrained during the screening. The virtual screen eventually

generated five candidates that were synthesized and demonstrated favorable properties.

6.4.5 Closed-loop Integration of Data Generation and Predictive Models

To the extent that the technologies above are in use today, the generation of data and the development of models which leverage them are largely independent. We believe the most promising approach to ensuring that there exists objective, accessible, and highly relevant data for a particular ML task is to integrate data generation and predictive model components into a closed loop workflow. Recently, methods have been demonstrated[43] which enable the models themselves to use Bayesian optimization or other active learning techniques to identify the next experiment, simulation or data query to execute based on the model's hypothesis. Most importantly, this is done with the explicit objective of acquiring the optimal data to inform that specific model and improve its performance. The model's hypothesis is then updated with the new data, and the process repeats.

There are several impacts of this paradigm. First, data from experiments designed in this way are more informative to the model than experiments conducted outside of the loop, and therefore less data are needed to produce accurate models. Indeed, they are maximally relevant in terms of the optimization function used to select the next experiment. This enables rapid development of models for domains where little data exist, and a means to intelligently navigate the astronomically large unexplored chemical space to extend into new domains. Second, the data are complete and objective, in the sense that all experimental conditions can be specified and controlled, and results can be preserved regardless of whether they confirm the hypothesis. The format can be specified to ensure accessibility of the data to models. Last, such a laboratory workflow may evolve from automation to autonomy—not only relieving the chemist of the drudgery of manually executing many aspects of an experiment, but also of deciding what experiment to execute and what its parameters are. Consider that robotic automation serves as the hands of a chemist, and AI tools as the mind of one. In the most sweeping conception, this laboratory would be able to discover and synthesize compounds supported by a minimal number of

technicians and guided by scientists who would need only provide the chemical search space and the desired properties. The automated system would provide fully documented data collection and traceable conclusions.

6.5 Initial Demonstrations of the Self-driving Laboratory

The term self-driving laboratory (SDL), coined by Aspuru-Guzik, describes an AI-powered lab, which can generate hypotheses and test them without human intervention.[87] This closed-loop aspect elevates an automated lab to an autonomous one. The conceptualization of the SDL aligns with the human chemist workflow—planning, experiment, analysis, and optimization—but the implementation of it is likely to streamline research in unforeseen ways. Where a human may have biases toward a certain type of reaction or analysis, the machine is agnostic toward these things, as its choices are entirely data-driven. Given the complexities of implementation, it is challenging to deliver quantitative measures of the advantage of the SDL. Nevertheless, several case studies within material science have yielded results that imply the benefits of this approach. Two are discussed here.

Nikolaev *et al.* utilized an AI planner built using a random forest/genetic algorithm to optimize conditions for carbon nanotube growth.[88] The authors were successful in increasing the growth rate throughout 600 sequential, autonomous experiments, and suggested that this type of autonomous experimentation (termed ARES) could be applied to other materials fields including metal oxidation, additive manufacturing, and chemical vapor deposition without much difficulty.

An investigation by Xue and coworkers discovered nickel-titanium shape memory alloys.[89] These flexible materials, which recover their shape upon heating, can be characterized by the thermal hysteresis (ΔT) required to cause the effect. To find alloy compositions, which had minimal ΔT, researchers tested several regression models and applied an efficient global optimization (EGO)[90] selector algorithm to suggest new candidate compositions. After nine experimental cycles, the system found an alloy

with ΔT=1.8 K, the lowest ever demonstrated. The key to success was the ability to balance exploration and optimization in the selector.

These examples and many others have demonstrated the promise of self-driving laboratories for material synthesis and discovery[91–94]—yet examples within chemistry are difficult to find. Comparison to the materials science experiments suggests that the main challenge is adopting design algorithms to non-continuous chemical space. Innovative strategies to transfer the self-driving laboratory tools to chemical space are needed.

The Cronin lab has worked on this problem using an experimental workflow called the Chemputer, which integrates closed-loop reaction screening and AI models together with the ability to ingest data from databases or NLP sources.[95] The system autonomously explored the reactivity of a defined set of molecules by performing a series of experiments, examining and classifying the product as reactive or unreactive, and using a support vector machine to predict the reactivity of untested pairs. The goal of the system was to find new reactivity, rather than to optimize a particular property. In a demonstration, the system discovered several novel multicomponent reactions.

To truly reproduce the traditional molecular design process within an autonomous workflow requires models that can emulate human creativity. The models discussed so far have primarily been discriminative models, which estimate a conditional probability $p(y|x)$. Given an observation x (*e.g.*, some molecular representation), the model predicts the probability of a property y (toxicity, solubility, *etc.*). Performance, and to a lesser degree interpretability, are of primary importance in determining representations for such discriminative models. Generative models, in contrast, attempt the more complex task of estimating the joint distribution $p(x,y)$ of observations and properties, from which new observations with desired properties may be described by computing $p(x|y)$. Generative models such as variational autoencoders and generative adversarial networks have gained popularity for generating novel speech, text, and imagery, and have also recently been applied to inverse molecular design.[96] In this application, the viability of the inverse design candidate is also of utmost importance, as ultimately the candidate must be synthesized. This has been approached by imposing constraints on the topology of the generated molecular graphs,[97] and

enforcing a Chomsky type-2 grammar that encodes chemical knowledge upon a string-based representation of a molecule.[98]

The ability to generate completely novel structures is one of the most exciting possibilities enabled by improvements in ML algorithms for chemistry. Recently, a new kinase inhibitor drug was discovered by a generative model.[99] Using an autoencoder approach, Zhavoronkov and coworkers used reinforcement learning to find an inhibitor of kinase implicated in fibrosis. The ability of the AI system to generate and prioritize candidates led to the discovery and validation of a nanomolar binder in just 46 days, compared to a typical timeline of 10 years or more and billions of dollars.[100,101] This demonstration study represents arguably the boldest step yet toward replicating human chemical intuition, while critical commentary suggests its limitations.[102] The DARPA Accelerated Molecular Discovery program will push these boundaries to explore wider swaths of chemical space and integrate them into closed loop workflows for a variety of DoD and other applications.

6.5.1 Implications of Autonomous Research

The ability for laboratories to design and synthesize novel molecules offers a future where vaccines are created in hours, vehicle coatings can be easily customized to specific climate, and affordable, customized catalysts neutralize greenhouse gases. But as AI enters the laboratory, it is wise to consider the ancillary impacts as well. It is impossible to design autonomous systems without recognizing the possible impact on employment. In most cases, scientific automation has served to expand experimental throughput rather than replace bench chemists. And the power of ML has led to huge increases in data science jobs. Yet, both have led to a shift in the types of skills required for researchers, and combining both these trends is expected to accelerate that shift. In the future, researchers will need to be capable coders, able to interpret ML results and comfortable manipulating structures on a computer.

It is not unusual for chemistry PhD students to learn these skills, but usually it is by necessity. In the future, they must be integrated into the core educational curriculum. Performing ibuprofen synthesis in microfluidics or building an ML model to predict binding could all be implemented into

students’ laboratory curricula without sacrificing the core chemistry ideas or requiring large investments in new equipment. It will, however, require that educators become familiarized with these new fields. The National Science Foundation, the American Chemical Society, and other organizations can help with this by providing educational seminars and materials; these can also be leveraged to benefit chemists working in industry who will also need to gain proficiency in these skills. Despite these changes, we believe that AI will support and empower scientists to be more efficient, creative, and prolific.

Outside of core research, other related areas will be disrupted. Chemical safety and environmental regulation must adapt to future discovery paradigms. Traditionally, regulatory approval for a new drug application required a standard battery of tests. Today's FDA is increasing in flexibility, something the new research paradigm outlined above could help support. Research within the FDA Center for Drug Evaluation and Research (CDER) has embraced the use of QSAR models for predicting toxicity and other outcomes based on a candidate drug's structure.[103] As such models become more common for discovery as well as the regulatory approval, it will enable toxicity and other assessments to be performed at earlier stages of development, avoiding costly drug regulatory failures and shortening approval time for patients. In a similar way, the ability for AI prediction of reaction byproducts and compound stability could lead to more streamlined assessment of a particular manufacturing route.

A compelling question is, if computers are inventing the molecules, who owns the intellectual property (IP)?[104] Currently, many computational tools are used for drug discovery, but they require enough human guidance such that they are not the source of IP. They might be considered the writer's pen, which has no claim to the author's copyright on their text. But if AI algorithms generate molecules autonomously, should the person who coded the algorithm maintain some part of the intellectual property? License agreements will need to address this. Imagine an open-source tool, which would rapidly generate high-quality drug candidates using publicly available data, analogous to Google's open-source AI tools for generating paintings or music. Would it be a race to submit them to the trademark office? Or, if no IP were granted, would we begin to see novel drugs

without patent?[105] Legal policy and business structure stand to be greatly impacted by the development and adoption of these tools. These factors must be considered alongside, not after, the development of automation and autonomy in the chemical sciences.

6.6 Conclusion

In 2016, Malaria killed 445 000 people[106] and drug-resistant bacteria killed 700 000.[107] For targets such as these, the moral obligation to leverage science against human suffering aligns with the national security interest to prevent global unrest sparked by pandemics. Yet, neither area attracts a fraction of the pharmaceutical investment that cancer or central nervous system illnesses do.[108] We believe that DoD is well positioned to lead the way. In August 2018, the FDA approved the first prophylactic drug for malaria in more than 18 years, which was developed at the Walter Reed Army Institute of Research.[106] Other DoD laboratories are developing cutting edge bulletproof plastics,[109] improved thermophotovoltaics,[110] and safer lasers.[111]

In the private sector, successes such as these would be followed by dramatically increased investment, drawing from large profits. This is not the case for government labs, which have relatively static budgets and do not reap the benefits of massive licensing deals. Yet, we have the opportunity to enhance the important work being done at these laboratories by providing AI tools to the researchers in them. The DARPA programs discussed here, as well as many other programs supported by the agency's $2 billion AINext campaign,[112] will create tools to make laboratories working on areas such as soldier health, munition performance, or jet fuel safety, operate more efficiently, providing benefits to the warfighter, the nation, and our allies. DoD investment to integrate ML and automation will both serve as a force multiplier for those researchers and de-risk the adoption of these tools for chemical and biological researchers in academia and industry.

Additional Reading

I. W. Davies, Digitization of Organic Synthesis, *Nature*, 2019, **570**, 175–181.

F. Feng, L. Lai and J. Pei, Computational Chemical Synthesis Analysis and Pathway Design, *Front. Chem.*, 2018, **6**, 199.

K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev and A. Walsh, Machine learning for molecular and materials science, *Nature*, 2018, **559**, 547–555.

A. F. de Almeida, R. Moreira and T. Rodrigues, Synthetic organic chemistry driven by artificial intelligence' *Nat. Rev. Chem.*, 2019, **3**, 589–604.

Acknowledgements

We would like to thank Dr Anne Fischer for her helpful suggestions on this chapter.

References

1. About DARPA, https://www.darpa.mil/about-us/about-darpa.
2. T. McQuade, *presented in part at the Make-It Proposers Day*, June, 2015.
3. Battlefield Medicine, https://www.darpa.mil/program/battlefield-medicine.
4. The Right Chemistry, Fast: Employing AI and Automation to Map Out and Make Molecules, https://www.darpa.mil/news-events/2018-06-19.
5. Accelerated Molecular Discovery, https://www.darpa.mil/program/accelerated-molecular-discovery.
6. A. G. Kusne, Machine Learning for High Throughput Materials Discovery and Optimization Applications, https://www.nist.gov/programs-projects/machine-learning-high-throughput-materials-discovery-and-optimization-applications.

7. 2018 NCATS ASPIRE Design Challenges, https://ncats.nih.gov/aspire/challenges.
8. W. Huyer and A. Neumaier, *ACM T. Math. Softw.*, 2008, **35**, 9:1–9:25.
9. Z. Zhou, X. Li and R. N. Zare, *ACS Cent. Sci.*, 2017, **3**, 1337–1344.
10. M. Wleklinski, B. P. Loren, C. R. Ferreira, Z. Jaman, L. Avramova, T. J. P. Sobreira, D. H. Thompson and R. G. Cooks, *Chem. Sci.*, 2018, **9**, 1647–1653.
11. A.-C. Bédard, A. Adamo, K. C. Aroh, M. G. Russell, A. A. Bedermann, J. Torosian, B. Yue, K. F. Jensen and T. F. Jamison, *Science*, 2018, **361**, 1220.
12. N. J. Gesmundo, B. Sauvagnat, P. J. Curran, M. P. Richards, C. L. Andrews, P. J. Dandliker and T. Cernak, *Nature*, 2018, **557**, 228–232.
13. D. Perera, J. W. Tucker, S. Brahmbhatt, C. J. Helal, A. Chong, W. Farrell, P. Richardson and N. W. Sach, *Science*, 2018, **359**, 429.
14. D. Niu, Y. Wang and D. D. Wu, *Expert Syst. Appl.*, 2010, **37**, 2531–2539.
15. T. Wuest, D. Weimer, C. Irgens and K.-D. Thoben, *Prod. Manuf. Res*, 2016, **4**, 23–45.
16. M. Nazari, A. Oroojlooy, L. Snyder and M. Takac, in *Advances in Neural Information Processing Systems 31*, ed. S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi and R. Garnett, Curran Associates, Inc., 2018, pp. 9839–9849.
17. A. Forster, in *2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, 2007, pp. 365–370.
18. D. T. Ahneman, J. G. Estrada, S. Lin, S. D. Dreher and A. G. Doyle, *Science*, 2018, **360**, 186.
19. C. Hartmann and V. Meyer, *Ber. Dtsch. Chem. Ges.*, 1893, **26**, 1727–1732.
20. J. D. More and N. S. Finney, *Org. Lett.*, 2002, **4**, 3001–3003.
21. C. W. Coley, R. Barzilay, T. S. Jaakkola, W. H. Green and K. F. Jensen, *ACS Cent. Sci.*, 2017, **3**, 434–443.
22. C. W. Coley, W. H. Green and K. F. Jensen, *Acc. Chem. Res.*, 2018, **51**, 1281–1289.
23. M. H. S. Segler, M. Preuss and M. P. Waller, *Nature*, 2018, **555**, 604.
24. C. W. Coley, D. A. Thomas, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen, V. Schultz, T. Hart, J. S. Fishman, L. Rogers, H. Gao, R. W.

Hicklin, P. P. Plehiers, J. Byington, J. S. Piotti, W. H. Green, A. J. Hart, T. F. Jamison and K. F. Jensen, *Science*, 2019, **365**, eaax1566.
25. S. Szymkuć, E. P. Gajewska, T. Klucznik, K. Molga, P. Dittwald, M. Startek, M. Bajczyk and B. A. Grzybowski, *Angew. Chem., Int. Ed.*, 2016, **55**, 5904–5937.
26. B. A. Grzybowski, S. Szymkuć, E. P. Gajewska, K. Molga, P. Dittwald, A. Wołos and T. Klucznik, *Chem*, 2018, **4**, 390–398.
27. J. Nam and J. Kim, *Eprint ArXiv161209529*, 2016, arXiv:1612.09529.
28. B. Liu, B. Ramsundar, P. Kawthekar, J. Shi, J. Gomes, Q. Luu Nguyen, S. Ho, J. Sloane, P. Wender and V. Pande, *ACS Cent. Sci.*, 2017, **3**, 1103–1113.
29. P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas and A. A. Lee, *ACS Cent. Sci.*, 2019, **5**, 1572–1583.
30. K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen and R. Barzilay, *J. Chem. Inf. Model.*, 2019, **59**, 3370–3388.
31. Recent work in natural language processing has shown that bidirectional models which explicitly consider information from preceding and proceeding tokens during training achieve state of the art results. The analogous task of applying such a model architecture to jointly considering forward and retrosynthesis has yet to be investigated but may prove promising, J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, Association for Computational Linguistics, Minneapolis, MN, 2018, pp. 4171–4186.
32. C. W. Coley, L. Rogers, W. H. Green and K. F. Jensen, *J. Chem. Inf. Model.*, 2018, **58**, 252–261.
33. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel and D. Hassabis, *Nature*, 2016, **529**, 484–489.
34. J. S. Schreck, C. W. Coley and K. J. M. Bishop, *ACS Cent. Sci.*, 2019, **5**, 970–981.
35. T. Klucznik, B. Mikulak-Klucznik, M. P. McCormack, H. Lima, S. Szymkuć, M. Bhowmick, K. Molga, Y. Zhou, L. Rickershauser, E. P.

Gajewska, A. Toutchkine, P. Dittwald, M. P. Startek, G. J. Kirkovits, R. Roszak, A. Adamski, B. Sieredzińska, M. Mrksich, S. L. J. Trice and B. A. Grzybowski, *Chem*, 2018, **4**, 522–532.

36. A. Cherkasov, E. N. Muratov, D. Fourches, A. Varnek, I. I. Baskin, M. Cronin, J. Dearden, P. Gramatica, Y. C. Martin, R. Todeschini, V. Consonni, V. E. Kuz'min, R. Cramer, R. Benigni, C. Yang, J. Rathman, L. Terfloth, J. Gasteiger, A. Richard and A. Tropsha, *J. Med. Chem.*, 2014, **57**, 4977–5010.
37. A. Cherkasov and B. Jankovic, *Molecules*, 2004, **9**, 1034–1052.
38. R. Huang, M. Xia, D.-T. Nguyen, T. Zhao, S. Sakamuru, J. Zhao, S. A. Shahane, A. Rossoshek and A. Simeonov, *Front. Environ. Sci.*, 2016, **3**, 85.
39. ImageNet, http://www.image-net.org/.
40. CoNLL, 2019, CoNLL, https://www.conll.org/.
41. M. L. Head, L. Holman, R. Lanfear, A. T. Kahn and M. D. Jennions, *PLoS Biol.*, 2015, **13**, e1002106.
42. D. Young, T. Martin, R. Venkatapathy and P. Harten, *QSAR Comb. Sci.*, 2008, **27**, 1337–1345.
43. C. O. Benjamin Sanchez-Lengeling, L. G. Gabriel and A. Aspuru-Guzik, 2017.
44. The top 10 pharma R&D budgets in, 2018, https://www.fiercebiotech.com/special-report/top-10-pharma-r-d-budgets-2018.
45. M. Reisch, *CEN News*, 2019, 97.
46. A. M. Virshup, J. Contreras-García, P. Wipf, W. Yang and D. N. Beratan, *J. Am. Chem. Soc.*, 2013, **135**, 7296–7303.
47. C. A. Lipinski, F. Lombardo, B. W. Dominy and P. J. Feeney, *Adv. Drug Delivery Rev.*, 1997, **23**, 3–25.
48. Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, *Chem. Sci.*, 2018, **9**, 513–530.
49. To store all possible configurations of 125 orbitals would require more bits of memory than atoms in the universe. However, a future quantum computer could potentially model such a system with just 250 qubits, K. Bourzac, *CEN News*, 2017, **95**, 27–31.
50. Y.-C. Lo, S. E. Rensi, W. Torng and R. B. Altman, *Drug Discovery Today*, 2018, **23**, 1538–1546.

51. Molecular descriptors calculation – Dragon – Talete srl, http://www.talete.mi.it/products/dragon_description.htm.
52. D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.
53. D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik and R. P. Adams, MIT Press, Montreal, Canada, 2015, pp. 2224–2232.
54. C. W. Coley, W. Jin, L. Rogers, T. F. Jamison, T. S. Jaakkola, W. H. Green, R. Barzilay and K. F. Jensen, *Chem. Sci.*, 2019, **10**, 370–377.
55. A. Mayr, G. Klambauer, T. Unterthiner, M. Steijaert, J. K. Wegner, H. Ceulemans, D.-A. Clevert and S. Hochreiter, *Chem. Sci.*, 2018, **9**, 5441–5451.
56. M. Matthews, 1993.
57. R. Kwok, *Nature*, 2018, **560**, 269–270.
58. It is certainly possible that the chemical representations being developed for ML purposes could find dual purpose as innovative ways to input chemistry for ELNs. Furthermore, the ability to integrate automated retrosynthesis or forward prediction would provide a powerful capability that could induce adoption of ELNs for synthetic chemistry research.
59. S. V. Ley, D. E. Fitzpatrick, R. J. Ingham and R. M. Myers, *Angew. Chem., Int. Ed.*, 2015, **54**, 3449–3464.
60. Chemify, http://www.chem.gla.ac.uk/cronin/chemify/.
61. UW BIOFAB, http://www.uwbiofab.org/.
62. Aquarium, https://www.aquarium.bio/.
63. J. Roberts, *presented in part at the SD2 Proposer's Day*, November, 2016.
64. *Custom Crafted Organisms, the Robotic Cloud Laboratory and DARPA*, https://www.digital-science.com/blog/news/custom-crafted-organisms-the-robotic-cloud-laboratory-and-darpa/.
65. Emerald Cloud Lab, https://www.emeraldcloudlab.com.
66. M. D. Wilkinson, *et al.Sci. Data*, 2016, **3**, 160018.
67. L. Howes, *CEN News*, 2019, 97.
68. M. Samwald, A. Jentzsch, C. Bouton, C. S. Kallesøe, E. Willighagen, J. Hajagos, M. S. Marshall, E. Prud'hommeaux, O. Hassanzadeh, E. Pichler and S. Stephens, *J. Cheminformatics*, 2011, **3**, 19.

69. Google AI Blog: Facilitating the discovery of public datasets, https://ai.googleblog.com/2017/01/facilitating-discovery-of-public.html.
70. Google AI Blog: Building Google Dataset Search and Fostering an Open Data Ecosystem, https://ai.googleblog.com/2018/09/building-google-dataset-search-and.html.
71. Making it easier to discover datasets, https://www.blog.google/products/search/making-it-easier-discover-datasets/.
72. I. V. Tetko, O. Engkvist, U. Koch, J.-L. Reymond and H. Chen, *Mol. Inform.*, 2016, **35**, 615–621.
73. Atom Consortium, https://atomscience.org.
74. MLPDS – Machine Learning for Pharmaceutical Discovery and Synthesis Consortium, https://mlpds.mit.edu/.
75. Applying machine learning to challenges in the pharmaceutical industry MIT News, http://news.mit.edu/2018/applying-machine-learning-to-challenges-in-pharmaceutical-industry-0517.
76. N. Dowlin, R. Gilad, K. Laine, K. Lauter, M. Naehrig and J. Wernsing, CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy, Microsoft Research, 2016.
77. N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig and J. Wernsing, *Manual for Using Homomorphic Encryption for Bioinformatics*, Microsoft Research, 2015.
78. T. Zhang, Z. He and R. B. Lee, ArXiv E-Prints, 2018, arXiv:1807.01860.
79. E. Kim, K. Huang, A. Tomala, S. Matthews, E. Strubell, A. Saunders, A. McCallum and E. Olivetti, *Sci. Data*, 2017, **4**, 170127.
80. S. Mysore, Z. Jensen, E. Kim, K. Huang, H.-S. Chang, E. Strubell, J. Flanigan, A. McCallum and E. Olivetti, ArXiv E-Prints, 2019, arXiv:1905.06939.
81. I. V. Filippov and M. C. Nicklaus, *J. Chem. Inf. Model.*, 2009, **49**, 740–743.
82. J. Staker, K. Marshall, R. Abel and C. M. McQuaw, *J. Chem. Inf. Model.*, 2019, **59**, 1017–1029.
83. J. Starr, E. Castro, M. Crosas, M. Dumontier, R. R. Downs, R. Duerr, L. L. Haak, M. Haendel, I. Herman, S. Hodson, J. Hourclé, J. E. Kratz,

J. Lin, L. H. Nielsen, A. Nurnberger, S. Proell, A. Rauber, S. Sacchi, A. Smith, M. Taylor and T. Clark, *PeerJ Comput. Sci.*, 2015, **1**, e1.
84. J. Ash and D. Fourches, *J. Chem. Inf. Model.*, 2017, **57**, 1286–1299.
85. J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, in *Proceedings of the 34th International Conference on Machine Learning*, ed. D. Precup and Y. W. Teh, PMLR, International Convention Centre, Sydney, Australia, vol. 70, 2017, pp. 1263–1272.
86. R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, T. D. Hirzel, D. Duvenaud, D. Maclaurin, M. A. Blood-Forsythe, H. S. Chae, M. Einzinger, D.-G. Ha, T. Wu, G. Markopoulos, S. Jeon, H. Kang, H. Miyazaki, M. Numata, S. Kim, W. Huang, S. I. Hong, M. Baldo, R. P. Adams and A. Aspuru-Guzik, *Nat. Mater.*, 2016, **15**, 1120.
87. F. Häse, L. M. Roch and A. Aspuru-Guzik, *Trends Chem.*, 2019, **1**, 282–291.
88. P. Nikolaev, D. Hooper, F. Webber, R. Rao, K. Decker, M. Krein, J. Poleski, R. Barto and B. Maruyama, *NPJ Comput. Mater.*, 2016, **2**, 16031.
89. D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue and T. Lookman, *Nat. Commun.*, 2016, **7**, 11241.
90. D. R. Jones, M. Schonlau and W. J. Welch, *J. Global Optim.*, 1998, **13**, 455–492.
91. Y. Liu, T. Zhao, W. Ju and S. Shi, *J. Materiomics*, 2017, **3**, 159–177.
92. S. V. Kalinin, B. G. Sumpter and R. K. Archibald, *Nat. Mater.*, 2015, **14**, 973.
93. D. P. Tabor, L. M. Roch, S. K. Saikin, C. Kreisbeck, D. Sheberla, J. H. Montoya, S. Dwaraknath, M. Aykol, C. Ortiz, H. Tribukait, C. Amador-Bedolla, C. J. Brabec, B. Maruyama, K. A. Persson and A. Aspuru-Guzik, *Nat. Rev. Mater.*, 2018, **3**, 5–20.
94. J.-P. Correa-Baena, K. Hippalgaonkar, J. van Duren, S. Jaffer, V. R. Chandrasekhar, V. Stevanovic, C. Wadia, S. Guha and T. Buonassisi, *Joule*, 2018, **2**, 1410–1420.
95. J. M. Granda, L. Donina, V. Dragone, D.-L. Long and L. Cronin, *Nature*, 2018, **559**, 377–381.
96. R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre,

T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2018, **4**, 268–276.
97. W. Jin, R. Barzilay and T. Jaakkola, in *Proceedings of the 35th International Conference on Machine Learning*, ed. J. Dy and A. Krause, PMLR, Stockholmsmässan, Stockholm Sweden, vol. 80, 2018, pp. 2323–2332.
98. M. Krenn, F. Häse, A. Nigam, P. Friederich and A. Aspuru-Guzik, ArXiv E-Prints, 2019, arXiv:1905.13741.
99. A. Zhavoronkov, Y. A. Ivanenkov, A. Aliper, M. S. Veselov, V. A. Aladinskiy, A. V. Aladinskaya, V. A. Terentiev, D. A. Polykovskiy, M. D. Kuznetsov, A. Asadulaev, Y. Volkov, A. Zholus, R. R. Shayakhmetov, A. Zhebrak, L. I. Minaeva, B. A. Zagribelnyy, L. H. Lee, R. Soll, D. Madge, L. Xing, T. Guo and A. Aspuru-Guzik, *Nat. Biotechnol.*, 2019, **37**, 1038–1040.
100. S. M. Paul, D. S. Mytelka, C. T. Dunwiddie, C. C. Persinger, B. H. Munos, S. R. Lindborg and A. L. Schacht, *Nat. Rev. Drug Discovery*, 2010, **9**, 203–214.
101. J. Avorn, *N. Engl. J. Med.*, 2015, **372**, 1877–1879.
102. S. Lemonick, *Chem. Eng. News*, 2019, 97.
103. H. Hong, M. Chen, H. W. Ng and W. Tong, in *In Silico Methods for Predicting Drug Toxicity*, ed. E. Benfenati, Springer, New York, NY, 2016, pp. 431–459.
104. A. Guadamuz, *WIPO Magazine*, 2017, **5**, 14–19.
105. A. Tombling, *The Chemical Engineer*, 2019, issue 939, available at: https://www.thechemicalengineer.com/features/where-next-for-ai-in-drug-discovery/.
106. New antimalarial drug, tafenoquine, approved for malaria prevention WRAIR, https://www.wrair.army.mil/node/67.
107. Tackling Drug-Resistant Infections Globally: Final Report and Recommendations, 2016.
108. Big Pharma has spent $91.1 billion developing cancer treatment drugs in 2019 - Axios, https://www.axios.com/cancer-drug-development-spending-big-pharma-2b658b9f-9738-4c6e-b637-c16798a10142.html.
109. Army Engineers New Body Armor 14-Times Stronger Against Enemy Fire, https://defensemaven.io/warriormaven/land/army-engineers-new-

body-armor-14-times-stronger-against-enemy-fire-9pr7UMMgX0K0YLiw-Hxe9w/.

110. Army converts heat to electricity at record high rates, breakthrough research shows U.S. Army Research Laboratory, https://www.arl.army.mil/www/default.cfm?article=3265.
111. Researchers Use Nano-Particles to Increase Power, Improve Eye Safety of Fiber Lasers | News, https://www.nrl.navy.mil/news/releases/researchers-use-nano-particles-increase-power-improve-eye-safety-fiber-lasers.
112. DARPA Announces $2 Billion Campaign to Develop Next Wave of AI Technologies, https://www.darpa.mil/news-events/2018-09-07.

CHAPTER 7

Machine Learning for Chemical Synthesis

ALEXE L. HAYWOOD,[a] JOSEPH REDSHAW,[b] THOMAS GAERTNER,[c] ADAM TAYLOR,[d] ANDY M. MASON[d] AND JONATHAN D. HIRST(0000-0002-2726-0983)*[a]

[a] School of Chemistry, University of NottinghamUniversity ParkNG7 2RDUK
[b] School of Computer Science, University of NottinghamJubilee CampusNG8 1BBUK
[c] TU Wien (Technical University of Vienna), Faculty of Informatics, Information Systems Engineering InstituteFavoritenstraße 9-111040 ViennaAustria
[d] GlaxoSmithKlineGunnels Wood RdStevenage SG1 2NYUK
*Email: jonathan.hirst@nottingham.ac.uk

7.1 Introduction

Synthetic chemistry is at the heart of medicinal and materials chemistry. Anticipating how molecules will react (forward synthetic planning) and how a molecule can be synthesised (retrosynthesis) rely on the expert knowledge and experience of a synthetic chemist. Repositories such as Reaxys and SciFinder contain millions of compounds, reactions, references, and bioactivities. These repositories allow chemists to search through vast

regions of chemical space to fill in any gaps in their knowledge. Reaction precedents on analogous structures and/or reaction types to molecules with no reported synthesis routes can be retrieved. Although integrated tools enable the search results to be filtered, analysing the information to make informed decisions is a manual process that requires the intellect and time of an expert synthetic chemist.

Automating retrosynthetic analysis, the recursive breakdown of a target molecule until reactant precursors are reached that are commercially available, would significantly reduce the time taken to plan chemical syntheses. The concept of using a computer to plan synthetic routes was first proposed around 50 years ago.[1] The history of CASD tools has been reviewed in detail,[2–5] from the pioneering work of E. J. Corey in the development of Logic and Heuristics Applied to Synthetic Analysis (LHASA)[6] to contemporary commercially available programs. Thus far, chemists are yet to implement these tools when planning synthetic routes to new molecules. A survey was completed in 2017 (13 chemists in two companies)[3] to determine what chemists expect from CASD tools. The following were identified as the most important aspects to include: (i) user-friendly, (ii) provide supportive literature examples, (iii) define possible bonds to be broken, (iv) lead to commercially available reactant precursors, (v) recognise conflicting reactivity and suggest protecting groups, and (vi) prioritise results. While aspects (i) to (v) can be implemented regardless of the chemist's area of expertise, prioritising results is a challenge. Chemists working in different parts of the chemical industry have different priorities when designing reactions. Their criteria for a suitable reaction could be based on cost, greenness, reaction conditions (such as temperature or catalysts), and so on. Therefore, CASD tools need to be flexible and rank pathways based on the criteria provided by the chemist.

Forward synthetic planning is the prediction of products given the reactants, reagents, and a set of reaction conditions. Experiments, which are predominantly used to identify reaction outcomes, are expensive, time-consuming, and require experienced chemists. It would therefore be beneficial for computational tools to identify the major product, any side-products, and validate retrosynthetic predictions. Optimising reaction conditions, such as catalysts and solvents, is also an important part of

synthetic planning. Changing a set of reaction conditions, even slightly, could result in the formation of a different major product or a failed reaction. Integrating CASD with high-throughput screening and robotic equipment holds much promise for the future of reaction optimisation.

Advances in artificial intelligence, machine learning methods and availability of big data have provoked renewed interest in CASD. In this chapter, we will outline sources and representations of reaction data, give a brief description of machine learning methods focused on synthetic chemistry, and provide detailed approaches to CASD with contemporary examples and comparisons.

7.2 The Nature of the Chemical Data

7.2.1 Data Sources

Large amounts of data are required for training machine learning algorithms. Typically, supervised learning methods (see Section 7.3) are used in this area; these require labelled data. This means the ground truth associated with each input is required, whether that be the reactants, products, or yield data. Reactions are often extracted from reaction databases, cleaned, and re-written in a machine-readable format.[7,8] The lack of a conventional approach for data extraction has resulted in different types of information, managed in different ways, used as the input to the machine learning methods. Databases are noisy and can contain duplicate and even erroneous reactions. These should be removed in the cleaning process. Recorded reactions that generated multiple products can be managed in two ways. The highest yielding (*i.e.* major) product can be kept and the others discarded,[8] or the reaction can be split into multiple single-product reactions.[9–12] Atom-to-atom mapping is often required in retrosynthesis and forward synthetic planning tools, usually to identify reaction centres. Atom map indices are used to label all atoms in the reactant species that are mapped to the atoms in the product species as shown in Figure 7.1. Reactions with incorrect or incomplete indices should be removed. Other reaction information, for example stereochemistry, has historically been

difficult to incorporate due to sparse availability, difficulty in encoding into the molecular descriptors, or difficulty in incorporating into the model.

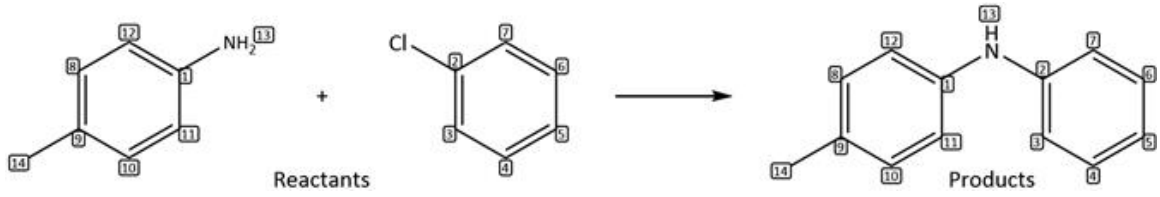


Figure 7.1 An example of an atom mapped reaction.

Reaction data reported in scientific journal publications and patents can be accessed through commercial repositories, such as Reaxys and SciFinder. Synthetic chemists use these tools to retrieve relevant experimental procedures, chemical literature, and patents. However, extracting large quantities of reaction data for data analysis or to train machine learning models requires a specific licence. Although rare, academic research groups have extracted ∼12.5M single-step reactions from Reaxys and built models for forward reaction prediction,[13,14] retrosynthetic route design[14,15] and the prediction of reaction conditions.[16] Nonetheless, to compare models an open-source dataset is required where access is not an issue.

Open-source patent data from the United States Patent and Trademark Office (USPTO) is the alternative dataset of choice when commercial licences are not available. US patents from 1976 to September 2016 were parsed by Lowe.[17] Reactants, products, reagents, solvents, quantities of molecules used, yield, temperature, time, preparation and workup steps were extracted using text-mining.[18] This common benchmark dataset contains ∼1M single-step reactions. Simplified Molecular-Input Line-Entry System (SMILES)[19] strings (see Section 7.2.2) represent the molecules in each atom mapped reaction. The USPTO dataset is useful for building synthetic design models, as it consists of many popular chemical reactions, particularly ones used in medicinal chemistry. Subsets of the dataset have been pre-processed and split into training, validation and testing sets. The USPTO_MIT subset of *ca.* 480K reactions has been used in reaction prediction.[20] For the retrosynthesis task, the filtered dataset of 50K

reactions (USPTO_50K) have been classified into ten reaction types (see Table 7.1).[21] This enables models to be compared based on individual reaction types. The USPTO dataset contains prophetic examples which are predicted experimental methods and results that have not been carried out.[22] Prophetic examples are allowed in US patents as long as they are not written in the past tense. Models based on patents that include prophetic examples should be considered with caution, as synthetic routes may not have been predicted based on experimentally valid examples.

Table 7.1 The ten reaction types in the USPTO_50K dataset.[21]

Reaction class	Reaction name	Size (%)
1	Heteroatom alkylation and arylation	30.3
2	Acylation and related processes	23.8
3	C–C bond formation	11.3
4	Heterocycle formation	1.8
5	Protections	1.3
6	Deprotections	16.5
7	Reductions	9.2
8	Oxidations	1.6
9	Functional group interconversions	3.7
10	Functional group addition	0.5

Reaction data published in journals and patents are biased toward high yielding reactions; low yielding and failed reactions are rarely reported. The latter may include reactions where all of the starting material remains or reactions leading to the formation of unexpected products. Supervised learning models built on published data are therefore limited, as counter-examples are required in training for optimum performance. The absence of failed data has been overcome by generating negative examples. Here, reaction templates (see Section 7.4) are used to generate chemically plausible “wrong/false” products.[13,15,23]

Electronic Laboratory Notebooks (ELNs) are used in the pharmaceutical industry by chemists to document their experimental procedures, conditions, and notes. They also store other useful data including yields and spectra. ELNs contain failed reaction data and hence can be utilised to

enhance supervised learning models, without generating artificial negative data. The restricted reactant space covered in ELNs contains common reactions used by chemists. The reactions are considered to be of high quality, reproducible, and reliable. Unfortunately, ELNs are not accessible to the public, nor in commercial databases due to confidentiality reasons. However, with access, the reaction data within ELNs has been used to build reaction prediction[24] and retrosynthesis models[8,24] as well as to predict reaction conditions.[25]

7.2.2 Molecular Descriptors

For a machine learning algorithm to learn from chemical structures, those structures must be represented in a way that an algorithm can process. Molecular descriptors are based on the structural, physiochemical, electronic, or topological nature of molecules. The type of descriptor used affects the performance of the algorithm as it has to capture the relevant information required.

One-dimensional descriptors represent molecules as a list of structural fragments. They are fast, easy to calculate, and simple to compare. An example is molecular fingerprints. These are vectors composed of binary digits that correspond to the presence or absence of specific substructures. This encoded information only defines one-dimensional features of the molecules. Molecular fingerprints can also contain two-dimensional information relating to atom connectivity by encoding the neighbourhood of each atom. Three-dimensional conformer information can also be encoded. A type of fingerprint called Morgan circular fingerprints has been used to predict reactant precursors by comparing the similarity of target molecules with products in a knowledge-base.[9] A reaction fingerprint, calculated by the difference between the sum of the products and the sum of the reactants fingerprints, has been developed to classify reaction types.[26]

Two-dimensional descriptors encode the topology (*i.e.* connectivity) of a molecule. For example, molecular graphs contain nodes that correspond to the atoms, and edges that correspond to the bonds that connect the atoms. The nodes are labelled by atom type and the edges are weighted based on bond type. Molecular graphs are fast to compute and easy to interpret, as molecules with identical graphs are identical molecules. Another type of

two-dimensional descriptors are line notations that describe a molecule as a sequence of characters. The most widely used line notation is SMILES and its extensions, SMARTS (SMILES arbitrary target specification) and SMIRKS (a reaction transform language). SMILES strings are defined as follows.

- Atoms are written as their atomic symbols.
- Single, double, triple and quadruple bonds are written as “-”, “=”, “#” and “$” respectively, although single bonds are usually omitted.
- Atoms in aromatic rings are encoded using lower case letters.
- Reactions can be written by splitting the reactants, reagents, and products with a “>” symbol.

For a single molecule, multiple SMILES strings may be valid, as the string can start from any atom within the molecule. This should be accounted for when building models, by either canonicalizing the SMILES strings or building a model that understands which different SMILES strings are equivalent. SMARTS strings are an extension of SMILES which include additional logical operators to allow searching for generic substructures. SMIRKS are another extension that describes generic reaction transforms. These line notations can be used as a direct input to machine learning algorithms.[27–29]

Another type of descriptor employs properties found using quantum calculations. Examples include NMR shifts, electrostatic charges, HOMO/LUMO energies, vibrational frequencies, and intensities. Quantum chemical calculations have been used as descriptors in the prediction of yield.[30] However, density functional theory (DFT) calculations are computationally expensive.

7.3 Machine Learning Methods

Machine learning algorithms can be divided into many categories depending on the given task. This section will concern itself with two of the most popular, namely supervised and unsupervised learning. In supervised learning, the relationship between input-output pairs is modelled, to build a

predictive model. The input data can be represented with various valid descriptors, examples range from a vector of numeric values to molecular graphs (see Section 7.2.2). Supervised learning models process the input data and return output data as a numerical or binary value. The key aspect of supervised learning is that the values of outputs are already known and can be used as examples to guide the learning process.

If the output data is a continuous or quantitative value, modelling its relationship with an arbitrary set of inputs is known as regression. Defining $X=\{x_i\}^N_{i=1}$ as the set of N input data points, $Y=\{y_i\}^N_{i=1}$ as the set of corresponding output data points and F as the unknown relationship between the two, regression estimates F based on the evidence that $y_i=F(x_i)$ for $i=1,\ldots, N$. The unknown function, F, does not have to represent a direct relationship between the input and output; it could represent a probability distribution over input-output pairs. However, the methods to infer F are the same and it is these that we are concerned with. Standard regression techniques assume that F belongs to a set of functions of similar structure and can therefore be parametrised based upon this structure. For example, one could assume that F is linear and parametrised by a vector β, thus $y_i=\beta^T x_i$ for $i=1,\ldots, N$. However, this can be restrictive if the relationship being modelled is not sufficiently described by the assumed form. To overcome this, many machine learning approaches make no such assumptions and, instead, infer F solely based upon the supplied data.

One popular approach to regression would be to use an artificial neural network (ANN). This is a model that can approximate an arbitrary relationship between the input-output pairs and is defined by a matrix of parameters W called the weights. During the learning process, the ANN uses the "Backpropagation" algorithm[31] in conjunction with a gradient-based optimisation scheme (for example, gradient descent) to produce an estimate $\hat{F}$ of F that minimises the error in its predictions. Each prediction $\hat{y}_i$ is a function of the input x_i and the weights W, so $\hat{y}_i=\hat{F}(x_i, W)$. Backpropagation computes the gradient of the loss function (which quantifies how accurate a prediction is) with respect to each weight. The gradient-based optimisation scheme then utilises this information to adjust W so that the error in prediction between $\hat{y}_i$ and y_i is reduced. This process

is repeated numerous times to allow the ANN to learn a set of weights that best describe the relationship. A more mathematical, yet still approachable, introduction to ANNs can be found in the review by Higham and Higham.[32]

We have described a method to model the relationship between input and output data, where the output is a real number. Another supervised learning problem considers the output as a binary value *i.e.* $y_i=1$ or $y_i=0$. This is known as classification. Whilst ANNs can also be used for classification, another effective model is the Support Vector Machine (SVM).[33] SVMs classify through the use of a hyperplane (a generalisation of a straight line to any number of dimensions) that best separates the dataset. If a data point resides on one side of the hyperplane, it is classified one way. If it lies on the other side of this hyperplane, it is classified the other way. This can be formalised by writing any hyperplane as the set of points x satisfying $w^t x-b=0$. Thus, any hyperplane is defined by w and b, which are identified by training an SVM classifier. Assuming the input data is linearly separable (it can be divided into two non-intersecting sets with a hyperplane), training an SVM selects two parallel hyperplanes of the form $w^t x-b=1$ and $w^t x-b=-1$ by minimising the cost function shown in (7.1).

$$\min_{w} \frac{1}{2}\|w\|^2 \qquad (7.1)$$

$$\text{subject to } y_i(w^t x_i - b) \geq 1$$

The constraint $y_i(w^t x_i-b)\geq 1$ ensures that both hyperplanes separate the data whilst also maximising the distance between them, which is usually referred to as the margin. To maximise the margin, it is necessary that each of the two hyperplanes must pass through at least one of the data points. Such data points are known as “support vectors”, hence the model's name. The SVM determines the hyperplane that “best separates” the data by maximising the margin. The resulting classifying hyperplane takes the form $w^t x-b=0$ and lies exactly halfway between the two. Figure 7.2 illustrates this. A new data point $\hat{x}$ is classified as 1 if $w^t\hat{x}-b\geq 0$ and -1 if $w^t\hat{x}-b\leq 0$. The previous description was based on the assumption that the input data are linearly separable, but this often is not the case. To account for this case,

one needs to incorporate slack variables into the SVM formulation. These variables allow certain data points to reside on the wrong side of the best separating hyperplane by adding a penalty to the cost function, as shown in eqn (7.2).

$$\min_{w} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \tag{7.2}$$

$$\text{subject to } y_i(w^t x_i - b) \geq 1 - \xi_i,\ \xi_i \geq 0$$

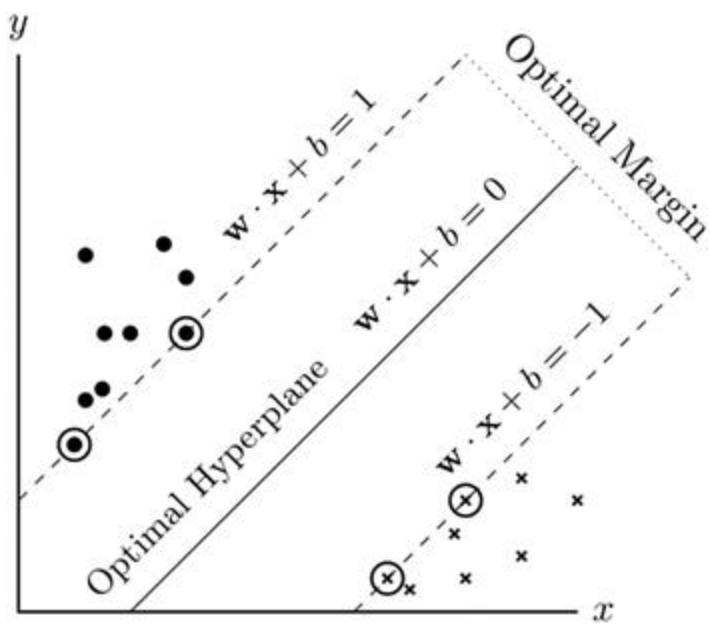


Figure 7.2 An example of a set of linearly separable data points. Also shown is the optimal hyperplane that separates them. The highlighted points are the support vectors.[33]

Here, the slack variables ξ_i are greater than zero if a data point x_i resides on the incorrect side of the hyperplane. A penalty of $C\sum_i \xi_i$ is added to the cost function to account for this. C is a user-defined parameter that balances the number of incorrectly classified points against the best fitting hyperplane. A more thorough treatment of this can be found in the article by Bennett and Campbell, which also includes the extension to non-linear decision surfaces.[34] Furthermore, a number of other algorithms can be used

to perform classification and these have been the subject of many reviews.[35,36]

Unsupervised learning is an approach in which we are only provided with input data and the aim is to find global (describing a property of the whole dataset or distribution) patterns within this. We distinguish this from data mining, which is a similar area of research that is more concerned with local (describing a subset of the whole dataset) patterns in the data. With no prior knowledge of the pattern, the method identifies them by searching for and exploiting commonalities within the data. A standard approach to this task would be to partition the data into sets of "similar" data points, or clusters. A clustering algorithm therefore creates a number of clusters and each data point is uniquely assigned to the cluster that it is most similar to. A simple, yet effective, algorithm to perform such a task would be *k*-means.[37] Initially, *k* data points are randomly selected to be the cluster centres. Each cluster is formed by the subset of data points closest to the respective cluster centre. The cluster centres are recalculated as the mean of the points assigned to the cluster. All data points that are not cluster centres are reassigned to their closest cluster and this process repeats until the clusters no longer change. Whilst it can be used for classification and regression (see Erman, Arlitt and Mahanti[38]), the process of identifying clusters is mainly used to better understand the dataset in question. Although *k*-means is guaranteed to find a clustering of the data points into *k* clusters, it is up to the practitioner to determine if anything meaningful can be inferred from the clustering. Another popular unsupervised learning algorithm is Principal Component Analysis (PCA). This is a dimensionality reduction algorithm that reduces a set of possibly correlated variables into a set of linearly uncorrelated variables called the principal components. Each principal component is calculated in such a way that it retains as much variance as possible. The first principal component retains the largest possible variance and all subsequent components retain the largest possible variance given that they are orthogonal to the previous principal components. This results in a set of variables that best represent the data; PCA is commonly incorporated into data visualisation. By plotting the first two principal components, one can visualise a low-dimensional representation of the original dataset. This is one of the more traditional

algorithms to perform dimensionality reduction. We point the reader to the review by Maaten, Postma, Herik for an overview of more recent approaches that aim to overcome the limitations of traditional PCA.[39] *k*-means is one of the simpler clustering methods and we encourage the reader to inspect the review by Fung for a more thorough overview of clustering methods.[40]

Having chosen an algorithm and built a model, the next point of concern is to evaluate how well it performs. Whilst performance can be measured in a number of meaningful ways, a key issue is to understand how well it generalises. Analogous to how a school teaches its pupils a topic and then tests them on related, but unseen questions, we "teach" the model using a set of data and test it on similar, but previously unseen data. This allows us to assess whether the model has learnt anything useful and evaluate how well it has learnt. In such analyses, one first splits the dataset into a training and test set (a 70:30 split is common). A model is built using only the training set and is used to make predictions on the test set. The key idea here is that the model has never before seen the test set, so it is using the knowledge it has learnt from the training data to make its predictions. This approach allows us to state the generalisability of the model to unseen data points. However, this may not be an accurate assessment of generalisability, as it is dependent on which data points belong to the train and test set. *K*-Fold Cross Validation (*K*-Fold CV) is a procedure to overcome this. Instead of splitting a dataset into a train and test set once, *K*-Fold CV performs the split *K* times. Initially, it divides the dataset into *K* sets (or folds) of approximately equal size. It takes the first fold to be the test set and the remaining $K-1$ folds to be the training set. The model is trained on the training set and makes predictions on the test set. This process is repeated using each fold once as the test set (ensuring that each data point belongs to the test set once). Figure 7.3 illustrates this. A score that evaluates how close the model predictions are to the actual values can be calculated for each train/test split, from which an average can be taken to get an overall score. Further details about *K*-Fold CV, as well as a number of other methods to estimate model accuracy and generalisability, can be found in the study by Kohavi.[41] Although *K*-Fold CV is a quick and easy way to reduce over-estimation of model accuracy, the data is split into *K* sets

randomly. Therefore, it could lead to an unreliable over-estimation of model performance if there is a distributional shift in the data. If the training and test sets are very similar, machine learning models can exhibit high accuracy due to memorising the training set (overfitting the data).[42,43] In the case of synthetic route design (see Section 7.4), care needs to be taken in the construction of training and test sets. In particular, one should be mindful that if highly similar reactions are present in both the training and test set, one may make an overly optimistic estimate of the generalisation achieved by the algorithm. Conversely, if the test set reactions are too dissimilar to those in the training set, one will be trying to make predictions beyond the scope of what one might reasonably expect the algorithm to learn.

Figure 7.3 An example of how K-Fold Cross Validation works. Here, *K*=5.

Now we can quantify how good our model is, it is worth considering how to pick the best model from a set of models. Generally, we should only compare models performing the same task *e.g.* two regression models. Let's assume we have the results of *K*-Fold CV for two different regression models. The next step in picking the best model would be to use a statistical test to determine whether the difference in performance is solely due to chance. One such test is the *t*-test, with a null hypothesis of “the difference in the means is equal to zero”. This test can be used to evaluate whether the difference between the means of two datasets is statistically significant, and

we could easily apply it to the results of our two *K*-Fold CV procedures. Performing such a test would allow us to accept or reject the null hypothesis. As α is defined as the probability of rejecting the null hypothesis when it is in fact true, the test indicating a rejection of the null hypothesis implies that we are $(1-\alpha)\times 100\%$ certain that the difference in the means is not zero. Taking alpha to be 0.05, another way to state this is that the probability of observing these two *K*-Fold CV results and their difference being due to chance is 0.05. This is an important point because things would change if we were to compare three models, which would involve three separate hypothesis tests. Assuming our confidence for each test is α, the probability of one of the differences due to chance is now $1-(1-0.05)^3=0.1426$ and is approximately three times as likely. To account for this, we use the Bonferroni Correction which states that, when testing *N* hypotheses, our confidence for each test must be equal to $\frac{\alpha}{N}$. This is one of many corrections that are applicable to multiple hypothesis testing, see the paper by Goeman and Solari[44] for a wider overview and the article by Demšar[45] for a thorough treatment of many statistical tests related to machine learning.

7.4 Synthetic Route Design

Designing synthetic pathways can be divided into two categories that are inverse processes. The first category, forward reaction planning, requires prior knowledge of reactant precursors to predict possible products of reactions. Retrosynthesis is the second, working backwards from a target molecule, predicting possible bond disconnections to obtain simpler reactant precursors. This step happens recursively until commercially available reactant precursors are reached. The overall synthetic pathway of a target molecule is a series of reaction steps. Single-step retrosynthetic design is far more complex than forward reaction prediction as there is no single correct answer. This is due to functional groups present in the output (reactants) being absent from the input (products). In the example Buchwald-Hartwig reaction shown in Figure 7.4a, the halide leaving group is not present in the product. Multiple precursors and different reactant

types may be plausible for a single bond disconnection. For example, the C–N bond in the target shown in Figure 7.4 can be synthesised by a Buchwald-Hartwig reaction with various halides or a Chan-Lam reaction. Without defining the reaction type, multiple bonds could be disconnected, particularly in larger molecules. In reaction prediction, all relevant functional groups are present in the input (reactants). This significantly reduces the number of possible reaction types that may occur and hence reduces the number of possible outputs in comparison to retrosynthesis.

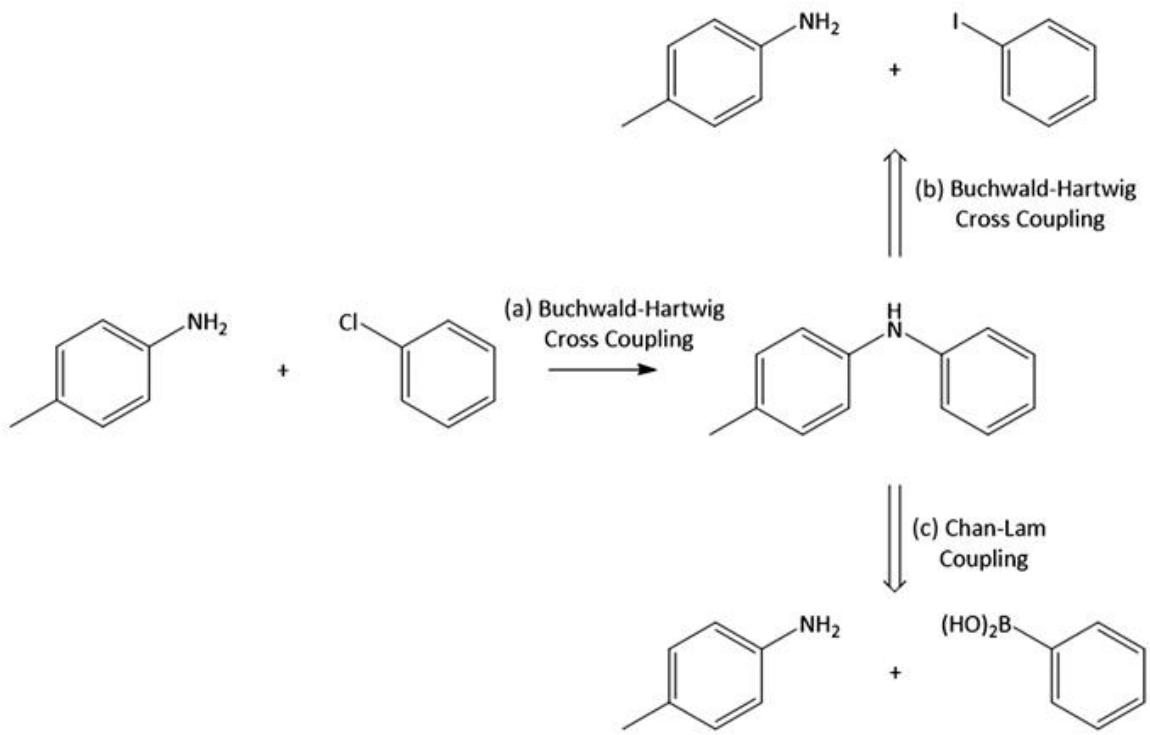


Figure 7.4 An example of forward synthetic planning (a) and retrosynthesis route design (b) and (c).

The most prominent approach to synthetic route design is the use of pre-defined heuristics to encode reactions. Heuristics, also known in this context as reaction templates, define a single chemical transformation from reactant(s) to product(s), or *vice versa*. Basic reaction templates contain the atoms and bonds involved in bond-making/breaking (the reaction centre) and any neighbouring atoms that may influence the reaction. Additional information may not be included due to data availability and

inconsistency.[8,23] More complex templates can include reaction conditions, reactivity conflicts, protection requirements as well as stereo-[46] and regioselectivity.[47,48] In multi-step retrosynthesis, the aim is to find the minimal number of reaction templates that need to be applied to a target molecule to reach commercially available reactant precursors. There is an inevitable trade-off between generalisation and specificity when defining reaction templates. If the templates are too generalised, the model will not be able to perceive any potential reactivity conflicts and may result in improper use. On the other hand, overly specific templates result in poor generalisability to unseen molecules.

Reaction templates are written in a user-friendly syntax, usually specific to the developers. Early programs manually hand-coded reaction templates.[49] This method suffered from three major issues: (1) Expert synthetic chemists with years of experience were required to code each reaction. (2) This labour-intensive task was not scalable to a large number of chemistries and incomplete template libraries covering a limited scope of reaction classes were often a result. (3) The complexity of reactions is not straight-forward due to reactivity conflicts and dependencies on reaction conditions and hence was difficult to codify. Despite these limitations, the commercially available program Synthia (formally Chematica) has proved that, with sufficient time investment, hand-coded reactions can be used successfully for the retrosynthesis prediction task.[47,48,50] A more efficient and less time-consuming process for defining templates is to extract reaction data automatically from the literature. Providing the reactions in the dataset are atom mapped, the reaction centre can be identified.[8,15,46,51–54] Tools, such as the Reaction Decoder Tool (RDT), are available to calculate atom map indices if they are not known.[52,53] A toolkit has also been developed for handling stereochemical information when defining reaction templates.[54] It is far easier to maintain automatically extracted template libraries and they have been employed in reaction prediction[14,55] and retrosynthesis.[14,15,56]

Contemporary template-based methods consist of two components: a pre-defined template library and a scoring function. These components can be combined in two different ways. The first is to exhaustively apply all possible reaction templates to obtain potential reactants or products and

then rank the generated candidates.[23] The second is to rank the reaction templates and only apply the most relevant templates to the input molecule to generate candidate species.[14,15,46] ANNs have been applied to both candidate and template ranking.[14,15,23] The similarity-based approach is slightly different to contemporary methods, as there is no pre-defined template library. This method is used exclusively in retrosynthesis. The target molecule is compared to the products in a knowledge base, reaction templates are extracted from the most similar reactions and are applied to the target to generate reactant candidates. The similarity of the candidates is compared to the precedent's reactants and used in combination with the product similarity score to rank the reactant precursors. This similarity-based approach differs from purely template-based methods as it is fully data driven.

Although template and similarity-based methods are easily interpretable, as they align with the way a synthetic chemist thinks, they have several limitations. The constant maintenance of template libraries is time-consuming and can be challenging when including new chemistries. Automatic extraction of templates from datasets is heavily dependent on atom-mapped reactions, which are not always completed or correct. Also, common atom-mapping tools are based on predefined expert rules. Reaction templates only focus on the local environment of reaction centres and, therefore, important groups in the global environment may be missed. Template-based methods are not able to make accurate predictions outside of their template library and hence cannot discover novel chemistries.[14] However, these methods are useful in aiding synthetic chemists with everyday retrosynthesis tasks.

The alternative to template-based methods is template-free methods. These methods are fully data driven and trained end-to-end without the construction of predefined reaction heuristics. Therefore, template-free methods are scalable to large datasets at a fraction of the computational cost and are generalisable to novel chemistries. To date, template-free models have either been graph-based or sequence-based. Graph-based models follow the same two-step procedure as those that are template-based in that reaction centres are identified to generate reactant precursors or products, depending on the task. Predictions generated from this approach are easily

interpretable as the approach is analogous to that of a synthetic chemist. Reaction centres have been identified to generate product candidates using a convolutional neural network on molecular graph descriptors.[20,55] Graph-based models have only focused on reaction prediction and not retrosynthesis.

Sequence-based models have a different approach to the above-mentioned methods. Here, the problem can be thought of as a machine translation task, whereby a language (*e.g.* English) is entered into a model that translates the text into another language (*e.g.* Spanish). When applied to reaction prediction and retrosynthesis the molecules can be represented by text notation, such as SMILES strings (see Section 7.2.2). In reaction prediction, the reactant SMILES strings are translated into a product SMILES string and *vice versa* for retrosynthesis. Sequence-to-sequence models have an encoder-decoder architecture consisting of two recurrent neural networks and an additional attention mechanism.[10,27,29] In this type of model each token in the SMILES string is assumed to be related to its neighbour and hence influences the nature of the neighbouring token. For each SMILES token, these models will only learn the local surrounding tokens and not tokens that are far away in the sequence. A transformer-based architecture has been adapted for the machine translation task.[57] These models differ from sequence-to-sequence models, as they do not contain recurrent neural networks. They have a fully attention-based encoder-decoder architecture. As the recurrent component is missing, the model is able to correlate individual tokens in the SMILES string regardless of their location. Unlike the sequence-to-sequence models, transformer models are able to capture long range interactions between tokens. Transformer models have shown success in both forward reaction prediction[28] and single-step retrosynthesis.[11,12,24,58] Sequence-based models have some advantages over two-step graph-based models. They do not require atom-mapping, atom features nor any chemical knowledge. The global environment of the molecules is also implicitly incorporated into the model. The major drawback to sequence-based methods is SMILES invalidity. The models do not directly learn the terminology of the SMILES notation, so these cannot be guaranteed to predict valid SMILES. The attention mechanisms present in all the template-free methods discussed

enable the models to be interpreted. Attention weights of the atoms (*i.e.* the SMILES tokens) can be used to identify which aspects of the input species influence the output species.

The performance of synthetic route design models is a balance between accuracy, reliability, and adaptability. Model accuracy can be calculated as the percentage of test set predictions where the ground truth was ranked within the top-*n* per cent of the predicted molecules for the given target. For example, the top-1 accuracy is the percentage of predictions where the correct molecule was ranked in the top-1 per cent. A model should be able to estimate the reliability of its predictions. The calculation of confidence scores should be implemented into model evaluation to identify when the model is likely to fail.[24,28] It is important to know how adaptable a model is. A model may have high accuracy scores, but can it generalise across chemical space? For example, if trained on patent data can it accurately predict synthetic routes to molecules in ELNs that cover a different range of chemical space?[24] These are important points to consider when evaluating synthetic route design models.

7.4.1 Retrosynthesis

7.4.1.1 Single-step Retrosynthesis

A combination of template-based and template-free models have been developed for the single-step retrosynthesis task. A target molecule is entered into the model which then disconnects a bond and generates possible reactant precursors. A benchmark dataset of 50 000 reactions extracted from US patents, herein named the USPTO_50K (see Section 7.2.1),[21] has been used to train and test single-step retrosynthesis prediction tools (see Table 7.2). A template-based model was built by Liu *et al.* as a baseline model.[10] The reaction templates consisting of the reaction centre, neighbouring atoms, and the reaction class were automatically extracted from the training data. For each target molecule, all templates in the same reaction class were applied to generate reactant candidates that were subsequently ranked. The similarity-based approach described previously, another type of template-based method, outperformed the baseline and the template-free sequence-to-sequence model developed by Liu *et al.* in every

reaction class.[9,10] The similarity-based model does not focus on finding only the major product but ranking all possibilities. As a result, it has the highest top-5 per cent accuracy of 81.2% when the reaction class is known prior. This approach is limited beyond the scope of the knowledge base and will always disfavour creative disconnections.

Table 7.2 The top-n accuracy for the Baseline[10] and Similarity[9] template-based models as well as the Seq2seq,[10] Transformer – Lin *et al.*,[12] SCROP,[11] MT[24] and Transformer – Karpov *et al.*[58] template-free models.

		Top-*n* accuracy (%), *n*=		
Data	Model	1	3	5
With reaction class	Baseline	35.4	52.3	59.1
	Seq2seq	37.4	52.4	57.0
	Similarity	52.9	73.8	**81.2**
	Transformer – Lin *et al.*	54.6	**74.8**	80.2
	SCROP	**59.0**	**74.8**	78.1
Without reaction class	Similarity	37.3	54.7	63.3
	Transformer – Lin *et al.*	43.1	**64.6**	**71.8**
	SCROP	43.7	60.0	65.2
	MT	**43.8**	60.5	—
	Transformer – Karpov *et al.*	42.7	63.9	69.8

Sequence-based, template-free models (Transformer – Lin *et al.*,[12] SCROP,[11] MT[24] and Transformer – Karpov *et al.*[58]) with a transformer architecture can overcome these limitations and have shown to be state-of-the-art even when the reaction class is unknown (see Table 7.2). As previously mentioned, a drawback to these methods is the prediction of invalid SMILES. The standard transformer architecture does not learn the grammar of the SMILES notation. Therefore, a predicted SMILES string may be grammatically invalid, but the grammatically correct version may be the ground truth. The self-corrected retrosynthesis predictor (SCROP) has an additional component to the other transformer models.[11] Once a

reactant SMILES is predicted, a transformer-based syntax predictor is used to automatically correct the SMILES string if it is not syntactically plausible. The addition of the syntax predictor improves the SMILES invalidity rate (see Table 7.3), remaining under 2.0% for the top-5 per cent of SMILES strings. The SMILES invalidity rate of the SCROP[11] model is also lower than the Seq2seq[10] and Transformer – Lin *et al.*[12] models. The Molecular Transformer (MT[24]) model does not report a SMILES invalidity rate for comparison. The SCROP[11] model achieves the highest top-1 accuracy of 59.0% when the reaction class was known.

Table 7.3 The top-n SMILES invalidity rate for the Seq2seq,[10] Transformer – Lin *et al.*,[12] and SCROP[11] template-free models.

	Top-*n* SMILES invalidity rate (%), *n*=		
Model	1	3	5
Seq2seq	12.2	15.3	18.4
Transformer – Lin *et al.*	2.1	3.5	4.7
SCROP-noSC	1.1	9.6	15.1
SCROP	0.7	1.4	1.8

All models were trained and tested with prior knowledge of the reaction class (apart from MT[24] and Transformer – Karpov *et al.*[58]) which is useful if a particular reaction is of interest. However, it is more realistic that the reaction class is unknown. Model training and testing of the Similarity,[9] Transformer – Lin *et al.*,[12] SCROP,[11] MT[24] and Transformer – Karpov *et al.*[58] models were performed without prior knowledge of the reaction class. Removal of the reaction class resulted in a drop of 11-16% in the top-1 per cent accuracy (see Table 7.2). All the transformer-based models have comparable top-1 accuracies.

7.4.1.2 Multi-step Retrosynthesis

The aim of CASD software is not only to predict the reactant precursors of a target molecule but also to break down the precursors recursively until they are commercially available. This multi-step retrosynthesis task can be thought of as a synthesis tree. A synthesis tree will have a branching factor -

the number of possible steps that can be taken from a particular molecule - and a depth - how many steps until the path is terminated. Pathway termination will occur if the precursors found are commercially available or if a predefined depth constraint is reached. For retrosynthesis, the branching factor is usually large and the depth low. The number of possible synthetic routes drastically increases with the number of steps in a pathway (depth). CASD tools require intelligent algorithms to identify the most promising branches and avoid exhaustive calculation of all possibilities. Unfortunately, choosing the optimum predicted reaction at every step may not lead to the most efficient pathway. The effect of each decision will not be known until the cost of the entire pathway is calculated. This cost could be related to the price of starting materials, the number of steps, waste generated, greenness, ease of product purification, sustainability, safety hazards, or environmental hazards. CASD tools generate synthesis trees to minimise a cost function that is often user-defined.

Commercially available CASD programs include InfoChem's ICSYNTH,[46] Wiley's ChemPlanner™ (formerly ARChem Route Designer)[51] that has integrated into CAS' SciFinder^{n} and Synthia™ (formerly Chematica), a trademark of Merck KGaA.[47,48,50] These programs are based on reaction templates. Synthia™ contains a template library of *ca.* 60 000 reaction templates, hand-coded by expert synthetic chemists. Along with reaction transforms, the templates also encode possible incompatibility groups, protecting groups and reaction conditions. ICSYNTH and ChemPlanner™ automatically extracted templates, consisting of the reaction centre and neighbouring atom/groups, from reaction databases. ICSYNTH also provides software to generate template libraries from in-house data that can be used independently or alongside the supplied libraries. Scoring functions, based on user-defined criteria, evaluate each synthetic step and guide the search to commercially available substrates. Synthia™ and ICSYNTH filter the predictions to remove any unwanted structures. Additional heuristics are defined in Synthia™ to penalise non-selective reactions, strained intermediates, and unlikely structural motifs. In ICSYNTH, the predicted structure is compared to a predefined list of unacceptable structures. ChemPlanner™ uses a similar approach to filter out functional group incompatibilities. Synthia™ and ChemPlanner™

account for regiochemistry and electronic effects. Synthia™ can also handle stereochemistry and steric effects. Each program generates synthesis trees with references to supporting literature and the cost of starting materials.

Segler and Waller have published their neural-symbolic model, composed of three neural networks (expansion policy, in-scope filter and rollout phase) combined with Monte Carlo tree search (MCTS) to guide the multi-step retrosynthesis search.[15] Reaction templates were extracted from 12.4 million reactions published before 2015 in Reaxys. The expansion policy and in-scope filter together are known as the expansion procedure, which is based on a knowledge base of 301 671 reaction templates containing reaction centres. The expansion procedure was applied to each target molecule to predict the reactant precursors for a single step. Here, the target molecule was represented by fingerprints and inputted to the expansion policy to prioritise the reaction templates. The top *k* templates were applied to the target to generate reactant candidates. The in-scope filter classified the reactions as feasible or not, to filter out any unfeasible reactions. The rollout policy, based on 17 134 reaction templates (containing reaction centres and neighbouring atoms), calculated tree position values. The expansion procedure and the rollout policy are incorporated into the MCTS algorithm to forms the 3N-MCTS model. Predicted pathways generated by the model for drug-like compounds could not be distinguished from literature routes according to a double-blind AB test. While the 3N-MCTS model can account for stereochemistry, it does not consider the prediction of reaction conditions in pathway generation.

7.4.2 Reaction Prediction

Retrosynthetic routes, either predicted by chemists or machine, are not guaranteed to be experimentally feasible. To validate the routes, one option is to test all the possibilities experimentally, which requires experienced synthetic chemists, a substantial amount of time, and high costs. An alternative validation technique is forward reaction prediction. Here, the reactants, reagents and possibly reaction conditions are known, and the aim is to predict possible products. It is important to include reaction conditions in the model as slight changes in solvent or temperature can result in a large change in yield or the formation of a different major product. Quantum

chemical calculations can be used to identify the major product, but the calculations are computationally expensive and limited to smaller molecules. Machine learning and artificial intelligence can be applied to large systems at a low computational cost.

The benchmark dataset for reaction prediction is the USPTO_MIT dataset (Section 7.1) composed of *ca.* 480K reactions.[20] State-of-the-art in reaction prediction are template-free models (see Table 7.4). The graph-based Weisfeiler-Lehman Difference Network (WLDN[55]) model outperformed the sequence-to-sequence (S2S[27]) model with an increase of 5.3% top-1 per cent accuracy. The WLDN model identifies the reaction centre from a reactant pool consisting of reactants, reagents and solvent species using a Weisfeiler-Lehman Network, a type of graph convolutional neural network. Another graph convolutional neural network (WLDN) then ranks the candidate products generated by focused enumeration. A sequence-based transformer model, the Molecular Transformer (MT[28]), outperformed the graph-based model (WLDN) by 4.8% in the top-1 per cent accuracy. The reported accuracies are for the USPTO_MIT dataset that was pre-processed to distinguish between reactants and reagents, prior to model training and testing. Removing the distinction results in lower model accuracy, 5.6% lower for the WLDN model and 1.8% for the MT model.

Table 7.4 The top-n accuracy for the S2S,[27] WLDN,[55] and MT[24,28] template-free models.

			Top-*n* accuracy (%), *n*=			
Training set	Test set	Model	1	2	3	5
USPTO	USPTO	S2S	80.3		86.2	87.5
		WLDN	85.6		92.8	93.4
		MT	**90.4**		94.6	95.3
USPTO	PF-ELN	WLDN	50.4	51.7	52.2	
		MT	**69.0**	80.3	82.9	
PF-ELN	PF-ELN	MT	**97.0**	98.5	98.8	

To assess the adaptability of the MT model, the MT and WLDN models were trained on the USPTO_MIT data and tested on reaction data within ELNs from Pfizer.[24] The two datasets cover different regions of chemical

space, which ensures the elimination of control bias from analogous reactions in the reported model accuracies. The MT model predicts the correct molecule within the top-1 per cent 69.0% of the time when predicting outside of its trained chemical space (see Table 7.4). The transformer-based architecture is currently the highest performing model type for both forward reaction prediction and single-step retrosynthesis.

7.4.3 Optimising Reaction Conditions

Identifying reaction conditions requires the expert knowledge and experience of a synthetic chemist. A reaction may be carried out under a variety of reaction conditions, but the most common conditions selected are those used for similar reactions that proved to be successful. Selection based on similarity can be insufficient, as reaction outcomes are influenced by a number of reaction parameters. Reaction conditions are therefore challenging to anticipate, especially because small changes in reaction parameters can result in quite different reaction outcomes or failed reactions. Chemists are required to find a combination of catalyst(s), reagent(s), solvent(s), temperature and pressure that work. Predictions from CASD programs and reaction prediction methods can only be validated experimentally if corresponding reaction conditions are also predicted. Therefore, incorporating condition prediction into such tools is crucial.

The prediction of reaction conditions is an under-explored area that has mainly focused on the optimisation of a single reaction condition or single reaction class. A binary classification model has been implemented for the prediction of the type of catalyst and solvent used in Michael additions.[59] A similarity-based approach was employed for the identification of catalysts for catalytic hydrogenation reactions, a type of deprotection reaction. The greatest interest has been in catalyst prediction and optimisation. Catalysts are important as they can increase rates of reactions and, in asymmetric catalysis, allow for the selectivity of one stereoisomer. Efforts to predict the selectivity of asymmetric catalysts have the potential to change how catalysts are designed, when one stereoisomer is favoured.[60–63]

It is desired that reaction condition prediction tools allow for the prediction of all reaction conditions. A neural network has been built to predict catalyst(s), solvent(s), reagent(s) and temperature for a range of

chemical reactions, which can be applied on a large scale.[16] The combination of machine learning with an automated synthesis and testing module is the future of reaction condition optimisation.[64] Computer-assisted synthetic planning and a robotic flow chemistry platform have been built for the execution of predicted synthetic routes to small molecules. This is a step closer to fully automatic synthetic chemistry.

7.5 Real-life Applications

CASD programs (see Section 7.4.1.2) that predict multi-step synthetic pathways to molecules may have high validation scores, but chemists will remain cautious until there is evidence supporting their true value. Applications thus far are primarily focused on drug-like targets and bioactive molecules.

InfoChem, in collaboration with AstraZeneca, has evaluated the performance of ICSYNTH (see Section 7.4.1.2) to act as an idea generator.[46] The ability of the program to predict routes to medicinal targets, present in AstraZeneca's commercial drug projects or the literature, was assessed. Predicted pathways were compared to the literature, brainstorm proposals and in-house experience of chemists working on the projects. Without prior knowledge of the brainstorm proposals, ICSYNTH rediscovered known chemistries, returned brainstorm suggestions and provided new unreported synthetic routes. Due to its unbiased nature, it also identified an unconventional transformation that leads to a non-intuitive solution to a problem.[46] ICSYNTH is able to generate ideas to complement synthetic chemists.

In a case study, synthetic routes proposed by ChemPlanner™ (see Section 7.4.1.2) were verified experimentally.[65] Synthetic routes to medicinally relevant targets were compared to routes developed by chemists. ChemPlanner™ identified almost all routes designed by chemists, although not all were identified as the best routes due to the scoring functions within the program. Alternative routes with fewer steps and lower costs were also suggested. Therefore, ChemPlanner™ can assist synthetic chemists in designing synthetic routes, saving them time and money.

Synthia™ (see Section 7.4.1.2) uses network theory, high power computing, AI, and expert chemical knowledge to design retrosynthetic pathways.[47] It can design novel synthesis routes to medicinally and industrially relevant targets. The program found synthetic routes to the selection of eight commercial bioactive substances and natural products.[47] The target products were chosen where previous attempts at synthesis were low yielding, not scalable, or failed. The predicted pathways were evaluated experimentally. Synthia™ designed routes that improved yields and saved money and time. It also designed a successful pathway to one of the targets that had no known synthetic routes. An extension of Synthia™ enables one to avoid patented routes.[48] It is important not to breach existing patented routes when designing pathways to novel molecules. A bond preservation method identifies bonds, which are essential to patents and prevents their disconnection in retrosynthesis. To validate this method, three commercial drugs linezolid, sitagliptin, and panobinostat were evaluated by Synthia™. Predicted pathways were generated with and without the patent bond constraints. Without constraints, Synthia™ suggested similar routes to the patents. However, by selecting the bonds that are not to be broken, the program could navigate around them and identify alternative routes.

7.6 Conclusions

In the pharmaceutical industry, progression from early-stage drug candidates to preclinical trials is performed in the quickest and the most cost-efficient manner. Synthetic chemistry has facilitated both drug discovery and development. Medicinal chemistry is a significant part of drug discovery, the aim being to design and synthesise novel small drug candidates for testing medicinal chemical hypotheses. For a single biological target, many analogues of lead compounds are synthesised on a small scale for screening, to identify structure–activity relationships. Medicinal chemists require reliable synthetic pathways that will deliver material for test quickly. Consequently, medicinal chemists are biased towards robust reactions when designing routes to novel molecules.[66] The use of artificial intelligence and machine learning in CADS programs

removes this personal bias in its predictions and therefore can broaden the medicinal chemists' toolbox.

Process chemistry is the key in drug development, whereby active pharmaceutical ingredients are scaled up and reactions are optimised. Process chemists usually work on a single target molecule, as opposed to many analogues. Synthetic routes to target molecules are optimised to meet SELECT (safety, environmental, legal, economics, control, and throughput) criteria.[67] Synthetic accessibility of molecules is one of the several criteria that influence which compounds are selected to be synthesised in medicinal chemistry and is an important concept in process chemistry. It is desirable to know whether a compound is synthesisable prior to performing the reactions experimentally. The closest to knowing is to predict synthetic pathways with high confidence. CASD programs could assist medicinal and process chemists in accelerating synthetic planning timelines, reducing the number of steps in a pathway, reducing costs, and suggesting alternative, unconventional pathways.

There are clear advantages for synthetic chemists to incorporate CASD into their everyday working lives. However, poor model interpretability has created suspicion among synthetic chemists. As a result, there is limited implementation of the tools. The so-called black-box tools will not be accepted unless they can be justified by literature or preferably, experimentally. Early and strong engagement with synthetic chemists will facilitate more rapid development of CASD with wider acceptance and uptake from the community.

There are other challenges that also need to be addressed to achieve advancement in CASD. Reaction data and code sharing are at the top of the list. Open-source large datasets containing counter-examples are required for training algorithms to perform optimally. Reaction data need to be of high quality and reproducible, containing no prophetic examples. There is a call for open sharing of machine-readable data and to define publication standards in order to incorporate this. Open sharing would allow for fair benchmarking of different methods. Rigorous validation methods for the assessment of model generalisability need to be established to ensure that reported results are not misleading.

CASD programs are aimed to assist synthetic chemists in the decision-making process for designing pathways. They are not intended to replace synthetic chemists; chemical intuition is still needed. The combination of robotics with CASD is the future of automated reaction optimisation. Such units would improve productivity in synthetic laboratories but will not replace the bench chemists.

Acknowledgements

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/S035990/1]. ALH is supported by EPSRC/NPIF [EP/S515516/1] through a CASE PhD studentship in partnership with GSK. JR is supported by an NPIF PhD studentship through the MRC IMPACT Doctoral Training Programme [MR/S502431/1]. We thank Dr Simon Macdonald and Dr Richard Hatley for useful discussions in the early stages of the drafting of this chapter.

References

1. E. J. Corey and W. T. Wipke, *Science*, 1969, **166**, 178–192.
2. C. W. Coley, W. H. Green and K. F. Jensen, *Acc. Chem. Res.*, 2018, **51**, 1281–1289.
3. O. Engkvist, P. Norrby, N. Selmi, Y. Lam, Z. Peng, E. C. Sherer, W. Amberg, T. Erhard and L. A. Smyth, *Drug Discov. Today*, 2018, **23**, 1203–1218.
4. A. F. de Almeida, R. Moreira and T. Rodrigues, *Nat. Rev. Chem.*, 2019, **3**, 589–604.
5. A. Cook, A. P. Johnson, J. Law, M. Mirzazadeh, O. Ravitz and A. Simon, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2012, **2**, 79–107.
6. E. Corey, A. Long and S. Rubenstein, *Science*, 1985, **228**, 408–418.
7. M. C. Swain and J. M. Cole, *J. Chem. Inf. Model.*, 2016, **56**, 1894–1904.
8. C. D. Christ, M. Zentgraf and J. M. Kriegl, *J. Chem. Inf. Model.*, 2012, **52**, 1745–1756.

9. C. W. Coley, L. Rogers, W. H. Green and K. F. Jensen, *ACS Cent. Sci.*, 2017, **3**, 1237–1245.
10. B. Liu, B. Ramsundar, P. Kawthekar, J. Shi, J. Gomes, Q. Luu Nguyen, S. Ho, J. Sloane, P. Wender and V. Pande, *ACS Cent. Sci.*, 2017, **3**, 1103–1113.
11. S. Zheng, J. Rao, Z. Zhang, J. Xu and Y. Yang, *arXiv:1907.01356*, 2019.
12. K. Lin, Y. Xu, J. Pei and L. Lai, *arXiv:1906.02308*, 2019.
13. M. H. S. Segler and M. P. Waller, *Chem. – Eur. J.*, 2017, **23**, 6118–6128.
14. M. H. S. Segler and M. P. Waller, *Chem. – Eur. J.*, 2017, **23**, 5966–5971.
15. M. H. S. Segler, M. Preuss and M. P. Waller, *Nature*, 2018, **555**, 604–610.
16. H. Gao, T. J. Struble, C. W. Coley, Y. Wang, W. H. Green and K. F. Jensen, *ACS Cent. Sci.*, 2018, **4**, 1465–1476.
17. D. M. Lowe, *Chemical reactions from US patents (1976-Sep2016)*, https://figshare.com/articles/Chemical_reactions_from_US_patents_1976-Sep2016_/5104873, (accessed 20 May 2019).
18. N. Schneider, D. M. Lowe, R. A. Sayle, M. A. Tarselli and G. A. Landrum, *J. Med. Chem.*, 2016, **59**, 4385–4402.
19. D. Weininger, *J. Chem. Inf. Model.*, 1988, **28**, 31–36.
20. W. Jin, C. W. Coley, R. Barzilay and T. Jaakkola, *arXiv:1709.04555*, 2017.
21. N. Schneider, N. Stiefl and G. A. Landrum, *J. Chem. Inf. Model.*, 2016, **56**, 2336–2346.
22. J. Freilich and L. L. Ouellette, *Science*, 2019, **364**, 1036–1037.
23. C. W. Coley, R. Barzilay, T. S. Jaakkola, W. H. Green and K. F. Jensen, *ACS Cent. Sci.*, 2017, **3**, 434–443.
24. A. A. Lee, Q. Yang, V. Sresht, P. Bolgar, X. Hou, J. L. Klug-McLeod and C. R. Butler, *Chem. Commun.*, 2019, **55**, 12152–12155.
25. P. Raccuglia, K. C. Elbert, P. D. F. Adler, C. Falk, M. B. Wenny, A. Mollo, M. Zeller, S. A. Friedler, J. Schrier and A. J. Norquist, *Nature*, 2016, **533**, 73–76.
26. N. Schneider, D. M. Lowe, R. A. Sayle and G. A. Landrum, *J. Chem. Inf. Model.*, 2015, **55**, 39–53.

27. P. Schwaller, T. Gaudin, D. Lányi, C. Bekas and T. Laino, *Chem. Sci.*, 2018, **9**, 6091–6098.
28. P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas and A. A. Lee, *ACS Cent. Sci.*, 2019, **5**, 1572–1583.
29. J. Nam and J. Kim, *arXiv:1612.09529*, 2016.
30. D. T. Ahneman, J. G. Estrada, S. Lin, S. D. Dreher and A. G. Doyle, *Science*, 2018, **360**, 186–190.
31. D. E. Rumelhart, G. E. Hinton and R. J. Williams, *Nature*, 1986, **323**, 533–536.
32. C. F. Higham and D. J. Higham, *SIAM Rev.*, 2019, **61**, 860–891.
33. C. Cortes and V. Vapnik, *Mach. Learn.*, 1995, **20**, 273–297.
34. K. P. Bennett and C. Campbell, *Acm Sigkdd Explor. Newsl.*, 2000, **2**, 1–13.
35. P. Kotsiantis, S. B. Zaharakis and I. Pintelas, *Emerg. Artif. Intell. Appl. Comput. Eng*, 2007, **160**, 3–24.
36. T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, NY, 2nd edn, **vol. 1**, 2009.
37. J. B. MacQueen, *Proc. fifth Berkeley Symp. Math. Stat. Probab.*, 1967, **1**, 281–297.
38. J. Erman, M. Arlitt and A. Mahanti, in *Proceedings of the 2006 SIGCOMM workshop on Mining network data*, ACM Press, New York, New York, USA, 2006, pp. 281–286.
39. J. Van Der Maaten, L. Postma and E. Van den Herik, *J. Mach. Learn. Res.*, 2009, **10**, 13.
40. G. Fung, *A Comprehensive Overview of Basic Clustering Algorithms*, 2001.
41. R. Kohavi, *Proc. 14th Int. Jt. Conf. Artif. Intell.*, 1995, **2**, 1137–1143.
42. I. Wallach and A. Heifets, *J. Chem. Inf. Model.*, 2018, **58**, 916–932.
43. J. Sieg, F. Flachsenberg and M. Rarey, *J. Chem. Inf. Model.*, 2019, **59**, 947–961.
44. J. J. Goeman and A. Solari, *Stat. Med.*, 2014, **33**, 1946–1978.
45. J. Demšar, *J. Mach. Learn. Res.*, 2006, **7**, 1–30.
46. A. Bøgevig, H.-J. Federsel, F. Huerta, M. G. Hutchings, H. Kraut, T. Langer, P. Löw, C. Oppawsky, T. Rein and H. Saller, *Org. Process Res. Dev.*, 2015, **19**, 357–368.

47. T. Klucznik, B. Mikulak-Klucznik, M. P. McCormack, H. Lima, S. Szymkuć, M. Bhowmick, K. Molga, Y. Zhou, L. Rickershauser, E. P. Gajewska, A. Toutchkine, P. Dittwald, M. P. Startek, G. J. Kirkovits, R. Roszak, A. Adamski, B. Sieredzińska, M. Mrksich, S. L. J. Trice and B. A. Grzybowski, *Chem*, 2018, **4**, 522–532.
48. K. Molga, P. Dittwald and B. A. Grzybowski, *Chem*, 2019, **5**, 460–473.
49. D. A. Pensak and E. J. Corey, *Computer-Assisted Organic Synthesis*, American Chemical Society, Washington, DC, **vol. 61**, 1977.
50. S. Szymkuć, E. P. Gajewska, T. Klucznik, K. Molga, P. Dittwald, M. Startek, M. Bajczyk and B. A. Grzybowski, *Angew. Chem., Int. Ed.*, 2016, **55**, 5904–5937.
51. J. Law, Z. Zsoldos, A. Simon, D. Reid, Y. Liu, S. Y. Khew, A. P. Johnson, S. Major, R. A. Wade and H. Y. Ando, *J. Chem. Inf. Model.*, 2009, **49**, 593–602.
52. S. A. Rahman, G. Torrance, L. Baldacci, S. Martínez Cuesta, F. Fenninger, N. Gopal, S. Choudhary, J. W. May, G. L. Holliday, C. Steinbeck and J. M. Thornton, *Bioinformatics*, 2016, **32**, 2065–2066.
53. P. P. Plehiers, G. B. Marin, C. V. Stevens and K. M. Van Geem, *J. Cheminf.*, 2018, **10**, 11.
54. C. W. Coley, W. H. Green and K. F. Jensen, *J. Chem. Inf. Model.*, 2019, **59**, 2529–2537.
55. C. W. Coley, W. Jin, L. Rogers, T. F. Jamison, T. S. Jaakkola, W. H. Green, R. Barzilay and K. F. Jensen, *Chem. Sci.*, 2019, **10**, 370–377.
56. J. S. Schreck, C. W. Coley and K. J. M. Bishop, *ACS Cent. Sci.*, 2019, **5**, 970–981.
57. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, *arXiv:1706.03762*, 2017.
58. P. Karpov, G. Godin and I. V. Tetko, in *Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions*, ed. I. V. Tetko, V. Krurková, P. Karpov and F. Theis, Springer International Publishing, Cham, 2019, pp. 817–830.
59. G. Marcou, J. Aires de Sousa, D. A. R. S. Latino, A. de Luca, D. Horvath, V. Rietsch and A. Varnek, *J. Chem. Inf. Model.*, 2015, **55**, 239–250.

60. A. F. Zahrt, J. J. Henle, B. T. Rose, Y. Wang, W. T. Darrow and S. E. Denmark, *Science*, 2019, **363**, eaau5631.
61. J. P. Reid and M. S. Sigman, *Nature*, 2019, **571**, 343–348.
62. J. L. Melville, B. I. Andrews, B. Lygo and J. D. Hirst, *Chem. Commun.*, 2004, 1410.
63. J. L. Melville, K. R. J. Lovelock, C. Wilson, B. Allbutt, E. K. Burke, B. Lygo and J. D. Hirst, *J. Chem. Inf. Model.*, 2005, **45**, 971–981.
64. L. M. Baumgartner, C. W. Coley, B. J. Reizman, K. W. Gao and K. F. Jensen, *React. Chem. Eng.*, 2018, **3**, 301–311.
65. S. Stark, R. Neudert, R. Threlfall, R. Neudert and R. Threlfall, *Wiley ChemPlanner Predicts Experimentally Verified Synthesis Routes in Medicinal Chemistry*, https://www.chemanager-online.com/en/whitepaper/wiley-chemplanner-predicts-experimentally-verified-synthesis-routes-medicinal-chemistry, (accessed 20 September 2019), 2016.
66. J. Boström, D. G. Brown, R. J. Young and G. M. Keserü, *Nat. Rev. Drug Discovery*, 2018, **17**, 709–727.
67. M. Butters, D. Catterick, A. Craig, A. Curzons, D. Dale, A. Gillmore, S. P. Green, I. Marziano, J.-P. Sherlock and W. White, *Chem. Rev.*, 2006, **106**, 3002–3027.

CHAPTER 8

Constraining Chemical Networks in Astrochemistry

SERENA VITI*[a,b] AND JONATHAN HOLDSHIP[a,b]

[a] Department of Physics and Astronomy, University College London, Gower Street, London, WC1E 6BT
[b] Leiden Observatory, Leiden University, PO Box 9513, NL-2300 RA Leiden, The Netherlands
*Email: serena.viti@ucl.ac.uk

8.1 Introduction

The space between the stars, the interstellar medium (ISM), is far from empty. The ISM is populated with interstellar clouds and filaments, made of gas (∼99%) and dust (∼1%). These clouds and filaments are the birth sites of stars.[1,2] Modelling the physical and chemical processes leading to the formation of stars and, hence, planets is a highly non-linear, very time-dependent, problem, involving a multi-layered interconnection between the physics and the chemistry of the gas and the dust. Despite the many advances driven by state-of-the-art telescopes, we are still far from solving the 'star formation problem': stars form in molecular filaments or clumps, cold (∼10 K) and relatively dense (>10^4 particles cm^{-3}) regions of the interstellar medium, where most of the gas is molecular. These regions also contain higher-density cores. The first two stages of the formation of solar-

like systems involve highly non-linear physical and chemical processes that are hard to corroborate by astronomical observations alone. In the first stage, some cores can become gravitationally unstable and initiate star formation. This collapse is controlled by the presence of molecules, which cool the gas and allow gravity to overcome hydrostatic equilibrium. During this pre-stellar core phase, due to the high density and low temperatures in the cores, species from the gas phase 'freeze' onto the dust grains present and form an icy mantle. The gas and surface compositions during these two stages exhibit a complicated time dependent, non-linear chemistry that strongly depends on the physical environment.[3] Little experimental information is available for the interstellar ices. What is the unprocessed ice composition? What are the efficiencies of the viable surface reactions? And how do the energetics of the ISM (cosmic rays, UV radiation, shocks) influence the processed ices?

Examples of key issues involve even the simplest molecules: the CO molecule sticks efficiently to surfaces at temperatures below ∼25 K and is abundant in the ices. Some of this CO can be converted to other species, and the observation of CO_2 and CH_3OH in ices[4] suggests that this *processing* does occur. H_2CO, CH_3OH, and glycolaldehyde are involved in the surface hydrogenation of CO, and determining their chemistry in star-forming regions is vital for studies of prebiotic chemistry.[5] Some ices return to the gas phase when the gas temperature rises above 20 K. At lower temperatures, non-thermal desorption processes can still return molecules from solid to gas-phase.[6] However, these non-thermal mechanisms 'compete' with the freeze-out. Gas-phase chemistry is better constrained than surface chemistry; nevertheless, many uncertainties still remain on the formation and efficiencies of routes for some of the most complex species (routinely called Complex Organic Molecules or COMs by astronomers) observed in space.[7,8] In summary, the composition of the gas and icy mantles varies according to a time-dependent process, which is highly dependent on the conditions of the gas and dust in any particular cloud.

In the second stage, *i.e.* when the protostar is born, the gravitational energy is converted into radiation and the envelope around the central object, the future star, warms up. The molecules frozen on the grain mantles during the previous phase acquire mobility and likely form new, more

complex species. When the temperature reaches the mantle sublimation temperature of ∼100 K, the molecules in the mantles are injected into the gas, where they react and form new, more complex molecules. Simultaneously, a fraction of matter is violently ejected outward in the form of highly supersonic collimated jets and molecular outflows. When the outflowing material encounters the quiescent gas of molecular cloud, it creates shocks, where the grain mantles are (partially) sputtered and the refractory grains are shattered. Once in the gas phase, molecules can be and are observed *via* their rotational lines. Again, as during the first stage, the interaction of gas and dust, and hence the gas composition, varies within very short timescales (less than one hundred years), and the effects of chemistry and dynamics are interlocked in a complex non-linear fashion.[9]

In both stages, molecules provide an essential tool for the analysis of the chemical and physical conditions of star- and planet-forming regions. Each stellar or planetary evolutionary stage is characterized by a chemical composition, which, *if properly interpreted*, leads to the determination of the physical processes of its phase.

This brief description of the multi-million years of star formation cycle highlights our challenge: the interconnection and non-linear correlation of the many parameters with each other or with extra, unknown ones, makes determining and specifying the parameter network within chemical models a highly challenging task.

Of particular importance is the reliability of the chemical reactions datasets. Databases of chemical reactions and rate coefficients for both gas and solid-phase reactions are the *key* input to all chemical models. While there are some gas-phase databases online, in reality each research group creates a personal version of such databases, especially as there is no accepted standard for databases; in addition, there is no database containing a complete set of solid-phase reactions. We explore the completeness and reliability of such databases in the next section.

8.2 Completeness and Reliability of Chemical Reactions Databases

Typically, chemical models of the ISM in use today employ databases of mainly 2-body (sometimes 3-body) chemical reactions; these contain long lists of gas and solid-phase chemical reactions. In the gas-phase, for each reaction, three constants are provided that are then used to calculate the rate coefficient. The first constant (usually α) represents the rate at 300 K and the other two constants (β and γ) give the temperature scaling of that rate. For photo-reactions, an equivalent set of constants gives the rate in an unshielded interstellar UV field and the extinction dependence of that rate. These three constants will have their own associated uncertainty. Depending on the physical conditions, between 20 and 50% of the reactions may not have been experimentally studied. In some cases, chemical reaction rates are therefore highly speculative.

For reactions occurring on the dust grains, the pathways, efficiencies, and branching ratios are even more uncertain than for the gas-phase reactions, due to the lack of experimental data. Surface reactions are, therefore, highly speculative and consequently are often not included in astrochemical models.

Reducing the degree of uncertainty for all the reactions would require an unfeasibly large number of laboratory experiments. Yet, chemical models fully rely on these databases, making their accuracy a prime problem for Astrochemistry. In the next two sections, we will describe elaborately the two categories of databases: gas-phase and surface reaction networks.

8.2.1 Gas-phase Networks

All the astrochemical models developed through the years calculate the abundances of hundreds of species involved in thousands of chemical reactions. Gas-phase reactions are the backbone of chemical networks as they are the primary routes for the formation and destruction of most molecules. Hence, a poor understanding of the rate coefficients can lead to large errors in the abundances of the main molecules observed in the ISM.

The range of kinetic temperatures and gas densities over which we need accurate rate coefficients will depend on the region of space that we need to model, but it can be from 10 K to >1000 K and from 100 to 10^{14} particles cm^{-3}, respectively. Chemical reactions, under these conditions, can be studied in the laboratory or theoretically. However, each experiment, or

study, takes at least a few months to complete. Often then, rate coefficients are estimated from other already known rate coefficients. More importantly, even if every reaction were to be experimentally or theoretically investigated, it would be impossible to do so under the whole range applicable to the ISM and many extrapolations are therefore performed. Measures have been taken by the community to compile databases where some quality assurance is performed. For example, the Kinetic Database for Astrochemistry (KIDA)[10] is a public gas-phase network, which provides all the rate coefficients from various sources and evaluates their accuracy where possible. We refer the reader to[10] for a detailed explanation of this network. However, even the best efforts at transparency and completeness such as KIDA fall short of providing the user with a complete, accurate, and reliable gas-phase network that includes all the possible species involved in the complex chemistry routinely observed in the ISM. This is especially true for complex organic molecules, which are species that contain 6 or more atoms, are present in the ISM mostly with low abundances and yet are key for our understanding of prebiotic chemistry in space.[7] Gas-phase networks including the formation and destruction of COMs are far from complete[11,12] and astrochemical models have been struggling to explain the observed abundances of COMs. While most current models tend to favour grain surface over gas-phase chemistry in COMs formation, several studies have now shown that COMs may indeed form in the gas phase but that, due to the incompleteness of gas-phase networks involving COMs, the relative contribution of gas-phase *versus* surface reactions cannot be quantified.

8.2.2 Grain-surface Networks

There is no doubt that dust grain chemistry plays a pivotal role in the formation of key abundant species observed routinely in the ISM *e.g.* H_2O, CH_3OH, and NH_3 which are primarily formed through solid state chemistry.[13,14] Although debate exists on whether gas-phase reactions contribute to the formation of COMs, it is clear that the latter are at least partly formed on the surface of the grains.[7,8,15]

We know that dust grains act as ‘catalysts’ meaning that – on their surface – surface reactions and energetic radiation can synthesize molecules

as complex as prebiotic species[16,17] starting from very simple molecules (*e.g.* CO and N_2) and atoms (H, C, O, N, S *etc.*) deposited from the gas when the temperature of the medium is close to 10 K. In the last couple of decades, many experiments have been performed to evaluate surface chemical reactions (see the review in ref. 18).

Molecular hydrogen was the first molecule to be studied on dust surfaces.[19] Many experiments since then have been performed to study the formation of more complex molecules *e.g.* ref. 20–22 as well as the ice morphology and ice mantle mechanisms *e.g.* ref. 23 and 24. However, all these experiments are performed within a constrained range of laboratory conditions, which differ from those found in the ISM: for example, atomic fluxes, ice temperatures, ice morphologies, and mixture ratios, energetic processes in the laboratory will differ from those found in the ISM. Hence formation, desorption and destruction routes and rates for surface molecules as derived from experiments may not always be exhaustive to the needs of the chemical modellers. In other words, experimental data for interstellar ices are limited, as the experimentation process is neither simple nor fast. As a consequence, most chemical models either include very simple surface reaction networks or ones where most of the reactions are essentially guesswork.

8.3 A Bayesian Approach

In astrochemistry, it is not common to use Bayesian methods as a means of deriving posterior probability distributions (PPDs) for model parameters from observations. The first study to do so[9] used Bayesian inference to derive parameters such as the gas density and cosmic-ray ionization rates within a dark molecular cloud from observations of species in the gas and ices using chemical models. This study particularly highlighted the nature of the problem at hand, namely that astrochemical problems are typically non-linear, ill-posed inverse problems where solutions may not be unique or may not depend continuously on the observational data. In one study,[9] a Bayesian approach based on the use of the Metropolis–Hastings (MH) algorithm (an example of a Markov chain Monte Carlo algorithm) was

adopted and they used a simplified version of UCLCHEM (a time-dependent gas–grain chemical model, now open source: https://uclchem.github.io/) to explore a nine-dimensional parameter space for molecular clouds. They ran two identical sets of eight MCMC chains differing only in the way the prior distribution information was set, with the first set having a non-informative prior information in the form of an acceptable range of possible values for observed ices, while in the second the prior include all the observational constraints from observations, including gas-phase abundances. Of particular relevance to this review, it is interesting to see how the different prior distributions have affected the resulting Posterior Probability Distributions (PPD) for one of their dimensions: the branching ratios for some key chemical reactions pathways *e.g.* those that control how much of the oxygen that freezes on the grains turns into H_2O or OH. With noninformative uniform prior, the high-density regions of the PPD cover excessively large sections of the distribution, meaning that the branching ratio could not be constrained enough. With informative priors, the high-density region in the PPD is reduced and gives a clear indication that the production of water over OH is favoured. This example showed for the first time in Astrochemistry that branching ratio parameters can be successfully estimated through Bayesian MCMC methods.

A more recent work[25] used the above methodologies as a proof of concept in an attempt to infer rates of reactions for a limited surface network, and in doing so provided the means to derive reduced networks in the context of observational constraints. In this work, a simple chemical model was developed that considers only the solid state chemistry in the ice mantles of dust grains in a dark molecular cloud. The simplified model is a time-dependent single-point model that generates a time series of solid-phase molecular abundances as a function of the physical conditions of the molecular cloud and the chemical parameters of the defined chemical network. This network only included surface reactions and mainly hydrogenation of common gas-phase species. We report these reactions in Table 8.1.

Table 8.1 Reactions included in the study by ref. 25. All the reactions are occurring on the grain surfaces. Reproduced from ref. 25, https://doi.org/10.3847/1538-4357/aae1fa, under the terms of the CC BY 3.0 licence, https://creativecommons.org/licenses/by/3.0/.

Number	Reactant 1	Reactant 2	Product 1	Product 2
1	O	H	OH	
2	OH	H	H_2O	
3	CO	OH	CO_2	
4	S	H	HS	
5	HS	H	H_2S	
6	H_2S	S	H_2S_2	
7	CS	H	HCS	
8	HCS	H	H_2CS	
9	CO	S	OCS	
10	OCS	H	HOCS	
11	H_2S	CO	OCS	
12	H_2S	H_2S	H_2S^2	
13	H_2S_2	CO	CS_2	O
14	H_2S	O	SO_2	
15	CS2	O	OCS	S
16	CO	HS	OCS	
17	S	O	SO	
18	SO	O	SO_2	
19	SO	H	HSO	
20	HSO	H	SO	
21	CO	H	HCO	
22	HCO	H	H_2CO	
23	H_2CO	H	H_3CO	
24	H_3CO	H	CH_3OH	

As one can see from this Table, only reactions among four types of atoms are considered: O, C, S, and H. Obviously, the reactions included are not

exhaustive of all the possible combinations: the criteria used to choose which reactions to include were based on (i) simple hydrogenation until saturation *e.g.* reactions 4 and 5 and (ii) reactions that have been found to be efficient, or even dominant, routes to forming a species *e.g.* reactions 21–24 to form methanol. For this toy model, gas-phase reactions were ignored, but the depletion of the gas phase on the surface was parameterized.

The result of this inference was the probability distribution of the reaction rates, shown as marginalized posteriors in Figure 8.1.

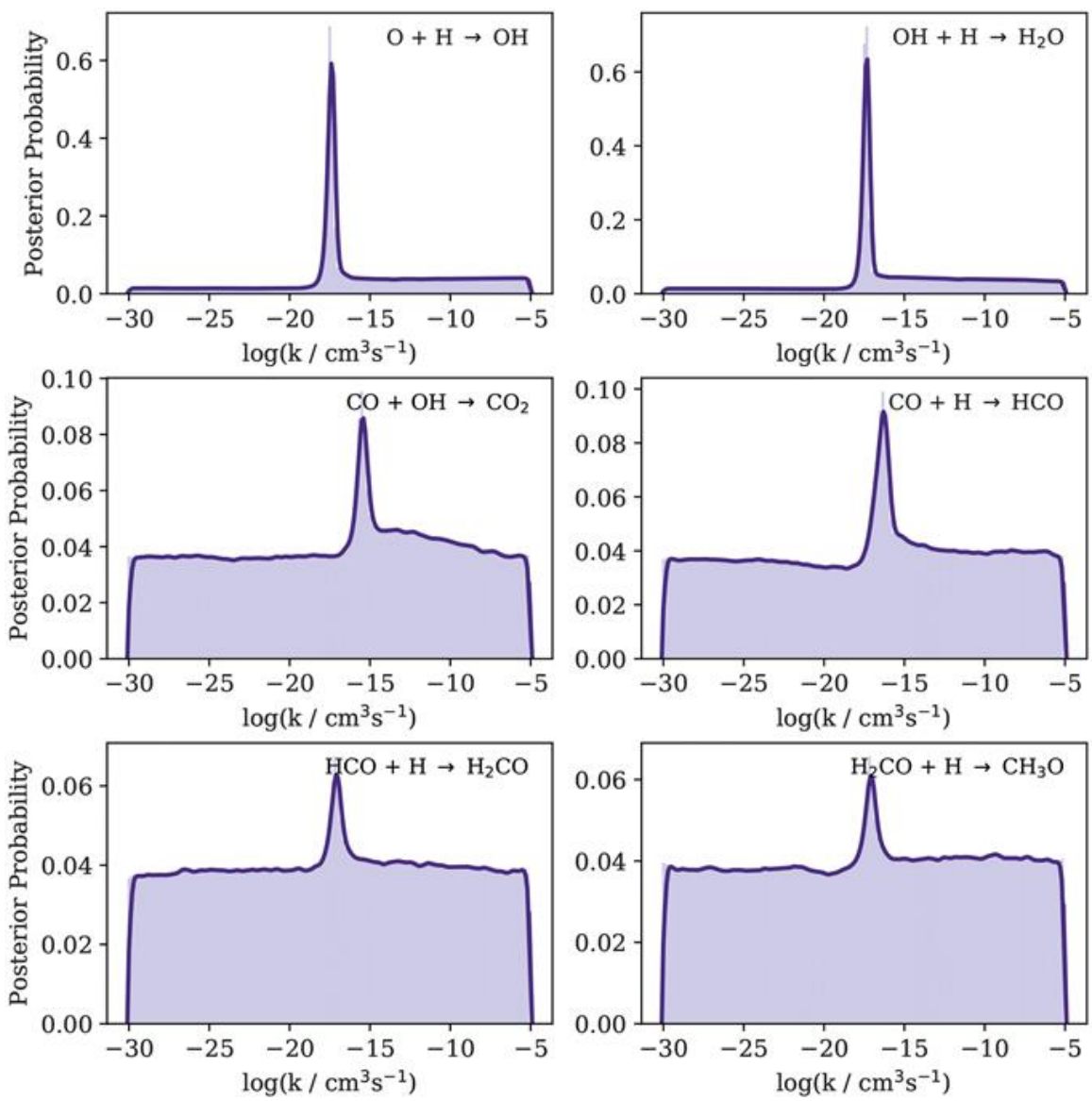


Figure 8.1 The probability distribution of selected reaction rates, shown as marginalized posteriors.

They find that when observational constraints directly constrain the abundance of a species involved in a reaction, the range of likely values for the rate of that reaction can be sufficiently narrowed for use in chemical models. Therefore, this method represents a method by which reliable reaction rates can be obtained for use in modelling until laboratory measurements or theoretical work supersede them.

Although this approach has many strengths, it requires a large amount of computational power. In both of the above examples, the chemical model used for the inference was greatly simplified in comparison with a typical model used to interpret observations. Where the model itself is not the subject of the inference (as it is in the work of ref. 25), we might use machine learning to map from model inputs to outputs in order to do more efficient parameter inference on the input parameters.

8.4 Machine Learning Techniques for Complex Networks

Ultimately, chemical modellers need to be able to rely on *complete* and accurate chemical reaction networks. Reducing the degree of uncertainty for such networks requires cutting edge Artificial Intelligence techniques combining Artificial Neural Networks, Bayesian Inference methods, and advanced Monte Carlo sampling algorithms. While we are still quite a way from achieving these, ML techniques have now started being used in Astrochemistry.

A neural network was used to emulate a full chemical model.[26] By creating a training set from the outputs of a full model (UCLCHEM[†]) for a wide range of input physical parameters, they were able to train a neural network to predict these outputs. This allows for the computation of chemical abundances required to interpret observations in milliseconds rather than the minutes of CPU time required to run a full chemical model. Consequently, computationally intensive parameter inference can be

performed using a close approximation of a full chemical model without taking an infeasible amount of CPU time.

Such neural networks would have uses beyond the direct interpretation of observations. Many radiative hydrodynamical simulations use highly approximate radiative-chemical treatments to provide the cooling rates for the simulated gas. This is a necessity when a full chemical model would add to the computational burden of an already intensive model. However, evaluating a neural network is generally a small task and as such, more accurate chemistry could be inserted into physical models in the form of pre-trained neural network emulators.

The use of machine learning techniques on chemical networks could be of particular importance for the reduction and accuracy of networks of complex organic molecules, in particular those that are as yet undetected in the ISM. For example, the 'holy grail' for astrochemists is to find amino acids in space. Amino acids are important organic compounds that play a key role in the formation of proteins. In the protein production process, amino acids join together to form polymer chains, which represent the structural units of proteins. It is believed that the formation of amino acids may indeed have occurred in the ISM, as they have been firmly detected in meteorites.[27] This is supported by laboratory experiments where amino acids are found to form on the surface of model dust grains by UV-photon and ion irradiation of suitable precursor molecules under astrophysically relevant conditions.[28,29] Despite great efforts towards their detection, a firm detection of amino acids in the ISM is still – unfortunately – eluding us.

One possible solution to this is to use ML tools to identify weak signals from the amino acids in spectroscopic telescope data. When detecting molecules, strong peaks in the spectra from an observed object are identified by matching their frequency to known emitting frequencies of molecules. If the emission is weak it may not be detected by a human, or the statistical significance of many weak lines may be missed. A classification model trained to categorize spectra as containing a particular molecule or not, based on the spectral noise and the emission profile at various frequencies, could give confident detections in cases that a human would miss. In the case of previously undetected molecules, such as amino acids, a training set can be produced by creating many synthetic spectra.

Further, such an analysis could be performed on thousands of targets with little effort which has made it an interesting prospect for the exo-planetary community aiming to identify molecules in planetary atmospheres.[30]

Another solution is to focus our search by determining the optimal conditions for observing amino acids. It is likely that, if present, such amino acids must form on the mantle of dust grains. It is therefore essential we determine the optimal conditions under which simple amino acids, such as glycine and alanine, can form on the dust icy mantles and can subsequently be released into the gas phase.

Key prerequisites for understanding how and where glycine and alanine form are the determination of the surface reactions that lead to their formation as a function of (i) kinetic temperature; (ii) gas density, (iii) structure of the gas, (iv) presence and abundance of other molecular species on the ices, and (v) UV and cosmic ray ionization rates, among other factors. To date, no large-scale predictions of the formation routes of glycine and alanine have been made, as a function of the parameters above.

A possible method to achieve this may be based on advanced and cutting edge techniques developed in the information sciences, ML and statistical disciplines. It should be possible to devise and apply a combination of statistical and ML techniques to perform large scale chemical models, involving large datasets of gas- and surface-phase chemical reactions to derive the physical and chemical conditions under which every surface chemical reaction is viable. A technique worth investigating is probabilistic graphical modelling, a branch of ML that studies how to use probability distributions to describe a particular problem whose ‘model’ has many uncertainties.

Ultimately, for any well-defined species set (COMs, amino acids, ice species *etc.*), we need to be able to simultaneously investigate the paths and efficiencies of their formation and destruction over a large physical (densities, temperatures, radiation fields, and cosmic ray ionization rates) and chemical (rate coefficients) parameter space, and discover if, where and under what conditions these species are abundant in space.

8.5 Conclusion

Machine Learning and probabilistic methods for solving typical astrochemical problems is a fast-growing field. As larger chemical reaction networks and more complex models are being employed in astrochemistry, the need for intelligent data mining algorithms will increase. It is clear that there are large uncertainties, as well as lack of information on formation and destruction routes, in gas and especially surface reactions and rate coefficients. To date, these uncertainties have only been tackled by laboratory experiments and quantum mechanical calculations. However, the complexity of the reaction networks, the lack of much prior information in the form of observed ices, and the length of each laboratory experiment, make a parameter exploration (in terms of gas densities, gas temperatures, UV, and cosmic ray fluxes for example) unfeasible. Initial 'experimentation' with ML algorithms has been proven to be an efficient avenue to tackle this challenge.

References

1. P. Andre', J. D. Francesco, D. Ward-Thompson, S. Inutsuka, R. Pudritz and J. Pineda, *Protostars and Planets VI*, ed. H. Beuther, R. S. Klessen, C. P. Dullemond and T. Henning, University of Arizona Press, Tucson, **914**, 2014, p. 27.
2. J. E. Dyson and D. A. Williams, *The Physics of the Interstellar Medium*, ed. J. E. Dyson and D. A. Williams, The Graduate Series in Astronomy, Institute of Physics Publishing, Bristol, 1997.
3. D. A. Williams, T. W. Hartquist, J. M. C. Rawlings, C. Cecchi-Pestellini and S. Viti, *Dynamical Astrochemistry*, Royal Society of Chemistry, 2017.
4. K. I. Obërg, A. C. A. Boogert and K. M. Pontoppidan, *et al.Astrophys. J.*, 2011, **740**, 109.
5. P. M. Woods, B. Slater, Z. Raza, S. Viti, W. A. Brwon and D. J. Burke, *Astrophys. J.*, 2013, **777**, 90.
6. J. F. Roberts, J. M. C. Rawlings, S. Viti and D. A. Williams, *Mon. Not. R. Astron. Soc.*, 2007, **382**, 733.
7. E. Herbst and E. F. Van Dishoeck, *Annu. Rev. Astron. Astrophys.*, 2009, **47**, 427.

8. P. Caselli and C. Ceccarelli, *Astron. Astrophys. Rev.*, 2012, **20**, 56.
9. A. Makrymallis and S. Viti, *Astrophys. J.*, 2014, **794**, 45.
10. V. Wakelam, J.-C. Loison and E. Herbst, *et al.Astrophys. J., Suppl. Ser.*, 2015, **217**, 20.
11. N. Balucani, C. Ceccarelli and V. Taquet, *Mon. Not. R. Astron. Soc.*, 2015, **449**, 16.
12. V. Barone, C. Latouche, D. Skouteris, F. Vazart, N. Balucani, C. Ceccarelli and B. Lefloch, *Mon. Not. R. Astron. Soc.*, 2015, **453**, 31.
13. B. Parise, C. Ceccarelli and S. Maret, *Astron. Astrophys.*, 2005, **441**, 171.
14. C. Ceccarelli, P. Caselli, E. Herbst, A. G. G. M. Tielens and E. Caux, in *Protostars and Planets V*, ed. B. Reipurth, D. Jewitt and K. Keil, University of Arizona Press, Tucson, AZ, 2007, p. 47.
15. H. Linnartz, S. Ioppolo and G. Fedoseev, *Int. Rev. Phys. Chem.*, 2015, **34**, 205.
16. W. Hagen, L. J. Allamandola and J. M. Greenberg, *Astrophys. Space Sci.*, 1979, **65**, 215.
17. V. Pirronello, W. L. Brown, L. J. Lanzerotti, K. J. Marcantonio and E. H. Simmons, *Astrophys. J.*, 1982, **262**, 636.
18. D. A. Williams and C. Cecchi-Pestellini, *The Chemistry of Cosmic Dust*, Royal Society of Chemistry, Cambridge, 2015.
19. V. Pirronello, C. Liu, L. Shen and G. Vidali, *Astrophys. J., Lett.*, 1997, **475**, L69.
20. N. Watanabe, A. Nagaoka, H. Hidaka and A. Kouchi, in *Protostars and Planets V*, ed. B. Reipurth, D. Jewitt and K. Keil, LPI, Houston, TX, 2005, p. 8244.
21. S. Ioppolo, M. E. Palumbo, G. A. Baratta and V. Mennella, *Astron. Astrophys.*, 2009, **493**, 1017.
22. M. Minissale, J.-C. Loison and S. Baouche, *et al.Astron. Astrophys.*, 2015, **577**, A2.
23. H. J. Fraser, M. P. Collings, J. W. Dever and M. R. S. McCoustra, *Mon. Not. R. Astron. Soc.*, 2004, **353**, 59.
24. M. P. Collings and M. R. S. McCoustra, IAU Symp. 231, in *Astrochemistry: Recent Successes and Current Challenges*, ed. D. C. Lis, G. A. Blake and E. Herbst, Cambridge Univerity Press, Cambridge, **vol. 405**, 2006.

25. J. Holdship, N. Jeffrey, A. Makrymallis, S. Viti and J. A. Yates, *Astrophys. J.*, 2018, **866**, 116.
26. D. de Mijolla, S. Viti, J. Holdship, I. Manolopoulou and J. Yates, *Astron. Astrophys.*, 2019, **630**, A117.
27. D. P. Glavin, J. P. Dworkin, A. Aubrey, O. Botta, J. H. Doty, Z. Martins and J. L. Bada, *Meteorit. Planet. Sci.*, 2006, **41**, 889.
28. G. M. Muñoz Caro, U. J. Meierhenrich, W. A. Schutte, B. Barbier, A. Arcones Segovia, H. Rosenbauer, W. H.-P. Thiemann, A. Brack and J. M. Greenberg, *Nature*, 2003, **416**, 403.
29. C. Meinert, I. Myrgorodka, P. de Marcellus, T. Buhse, L. Nahon, S. V. Hoffmann, L. Le Sergeant, d'Hendecourt and U. J. Meierhenrich, *Science*, 2016, **352**, 208.
30. I. Waldmann, American Astronomical Society, Extreme Solar Systems 4, id. 329.08, *Bull. Am. Astron. Soc.*, 2019, **51**(6).

---

† uclchem.github.io

CHAPTER 9

Machine Learning at the (Nano)materials-biology Interface

DAVID A. WINKLER[a,b,c,d]

[a] La Trobe Institute for Molecular Science, La Trobe University, Bundoora 3046, Australia
[b] Monash Institute of Pharmaceutical Sciences, Monash University, Parkville 3052, Australia
[c] School of Pharmacy, University of Nottingham, Nottingham NG7 2QL, UK
[d] CSIRO Data61, Pullenvale 4069, Australia
Email: d.winkler@latrobe.edu.au

9.1 Machine Learning Overview, Challenges, and Opportunities

Currently, machine learning (ML) is one of the most exciting computational techniques in science, technology, and medicine. The reason for the excitement is that, unlike traditional computer programs that have to be programmed precisely with every operation required to achieve a goal, ML methods can learn from examples in a similar way to humans. The same ML program can be trained to model and predict many different types of experiments and scenarios and can take into account heterogeneous data types like experimental conditions, physical and chemical properties, and processing history of samples.[1] ML models can generate models of continuous properties like water solubility, biological activity, and materials strength, but also categorical data (good/bad, active/inactive, nontoxic/moderately toxic/very toxic *etc.*). Newer

types of ML can also generate very good features, *i.e.* mathematical representations of intrinsic properties of molecules, materials, images *etc.* that generate improved models, as the quality of features is paramount when constructing ML models.[2]

Not surprisingly, artificial intelligence (AI) methods, largely ML, are generating a lot of new and potentially disruptive science across broad research and application domains.[3] There is an exponential increase in the number of publications that employ ML; many of these are in top-level journals, like Nature, Science, Cell and PNAS and their families of journals. However, ML is not new. The first neural network, the perceptron, was developed in 1958. A simple statistical modelling antecedent, quantitative structure–activity (or property) relationships (QSAR/QSPR), arose in chemistry at a similar time. Shortly after, neural networks were used to model relationships between chemical structures and other molecular and physicochemical properties of organic molecules and some useful property, especially biological effects. There was a steady progress in this field, but several important roadblocks hindered more rapid progress and wider application of these ML methods. The main issues were very limited computational technologies, lack of data to train ML models, a lack of clarity on how to generate mathematical entities that encode useful information, lack of understanding on how to reduce the dimensionality of ML modelling problems (feature selection), and even long-standing academic arguments on the best way to measure how predictive an ML model is.[4] Despite these limitations, a lot of useful research was conducted in this period; the method generated new drug candidates and predicted their possible toxicities and metabolic liabilities, and many of the limitations of low availability of computational resources were eased as computing power underwent spectacular development in subsequent decades.

At the moment, there is a happy coincidence of (or perhaps a deliberate push towards) advanced development in computational technologies and algorithms (*e.g.* deep learning (DL)), automation and robotics generating very large data sets suitable for training ML models, and development and/or application of a much broader range of powerful ML algorithms with unprecedented capabilities. What has also changed in chemical applications of ML is that we have found very good solutions to many of the vexing roadblocks in the QSAR/QSPR method (a model for ML in chemistry). Deep learning methods such as autoencoder/decoder, adversarial, and convolutional neural networks, for example, have simplified the problem of generating mathematical

descriptors to train ML models. Indeed, these methods can now take simple representations of molecular entities (*e.g.* SMILES strings, linear text representations of organic molecules) and generate relevant descriptors internally. We have also adopted powerful new descriptors like fragment and signature descriptors[5] that generate very good models. Unlike older and more arcane descriptors (*e.g.* those generated by descriptor generating packages like Dragon[6]), they are much easier to interpret in terms of molecular interactions and mechanisms. There is now a much wider use of efficient sparse feature selection methods (*e.g.* LASSO) that remove poor and uninformative descriptors, making models faster to train, easier to interpret, and more predictive.

These developments have driven a substantial expansion of ML into new areas such as forward and retrosynthetic reaction prediction,[7] prediction of thermodynamic and dynamic stability of materials, prediction of protein structures from sequence,[8] prediction or properties and performance of porous and 2D materials,[9] for energy applications, and many others.[10] Very recently, these techniques are being applied to autonomous evolutionary design of robots,[11] neatly completing the circle where robotics have allowed the generation of large amounts of data, which is then used to train models that can generate new robot architectures. This chapter will not discuss these applications as many are well covered in other chapters in the text.

Data availability has grown massively with rapid developments in robotics, automation, and omics technologies. Copious and diverse data representative of the domain of the problem being studied are essential for data-driven modelling methods like ML and DL. However, the size of possible accessible materials space ($\sim 10^{100}$, essentially infinite) clearly shows that experiments alone, no matter how fast or well planned (*e.g.* using design of experiments), cannot cover such spaces adequately. More efficient methods are required to identify 'islands of utility' in the vast sea of materials possibilities. This strongly suggests that ML methods need to couple synergistically with evolutionary algorithms that can explore very large search spaces very efficiently. This will be discussed in detail later in this chapter.

9.2 Scale and Importance of Problem/Special Issues with Complex Materials

Like ML and big data, the fields of materials science, regenerative medicine, and biomaterials are undergoing unprecedented expansion. They are taking advantage of new automated methods of synthesis, characterization, and testing of materials.[12] Much of the most interesting research is occurring at the boundary between materials and biology *e.g.* biomaterials, tissue engineering, stem cell control, implantable and indwelling devices, nanoscale smart delivery systems, 2D materials for wearable sensors, robotics, and prosthetics.

However, ML modelling of complex materials has special problems compared to small organic drugs or drug-like molecules. Nanomaterials exhibit distributions of shapes and sizes, their surfaces change dynamically depending on the environment in which they are embedded, and the materials are often not well characterized. Similarly, polymers are also not well characterized, they are polydisperse, crosslinked, their structures are not precisely known *etc.* There is a strong need for systematic studies on how properties of these materials affect the behaviour of systems constructed from them. Clearly, characterizing complex materials mathematically for use in training ML models is problematic. Materials-specific descriptors are largely not available or if they are, are quite arcane and uninterpretable. However, new DL methods such as encoder-decoder networks and general adversarial neural networks will significantly improve this deficiency, as they can generate internal descriptors ‘on the fly’ from chemically simple representations.

There are massive opportunities provided by contemporary ML technologies. They are providing, for the first time, the opportunity to tackle scientific, medical, social, and technological problems that have been out of reach due to their complexities. We now describe several areas in materials science where ML has provided novel solutions to very complex problems. The aim is to show how ML is now being used; the focus is on applications that are useful and interesting rather than those that are cutting edge but obscure.

9.3 Machine Learning Examples in Materials

We provide a few examples of where ML has been used to solve important problems in the energy and environmental arena. This section, which is brief because it is covered by others, shows recent research into ML modelling of porous and 2D materials properties.

Porous materials like metal organic frameworks (MOFs), zeolitic imidazolate frameworks (ZIFs), and covalent organic frameworks (COFs) are attracting much interest because of their diversity, tunability, and potential energy-related applications such as carbon capture, hydrogen and methane storage, electrocatalysts for carbon dioxide electro-reduction and water splitting reactions and related applications. There are millions of porous materials that could be synthesized that reside in large porous materials databases; however, it is not known which of these are stable and synthesizable. For every reported successful synthesis of an MOF, there are undoubtedly hundreds or thousands of failed reactions that have been attempted. ML could use knowledge of successful and failed reactions to predict and optimize the syntheses of new MOFs. MOF synthesis involves the self-assembly of metal ions and organic linkers in a 3D periodic network. However, our understanding of this self-assembly process is still too poor to predict which combinations will be able to be synthesized or which synthesis conditions will be optimal. Moosavi *et al.* recently used ML to capture chemical intuition from a set of successful and unsuccessful experiments to synthesize a metal–organic framework.[13] They used a random forest ensemble learner to evaluate the importance of variables in the synthesis of a MOF made from copper ions and 1,3,5-benzenetricarboxylic acid, thereby generating a material with unprecedented surface area. Moghadam *et al.* subsequently applied ML to predict the mechanical stability of MOFs that are suited to their applications.[14] They first conducted high-throughput molecular simulations for 3385 MOFs containing 41 distinct network topologies, which were then used to train an ML model to predict the mechanical properties of MOFs. Their model trained using five accessible descriptors, could rapidly estimate the bulk moduli of new or existing MOFs, which are representative of the critical pressure these materials can withstand under typical operating conditions, a valuable tool for the MOF community.

In the energy applications area, Thornton and co-workers reported the use of ML to design zeolite catalysts for CO_2 reduction.[15] They employed statistical and machine learning (QSPR) modelling techniques to identify quantitative relationships between the zeolite structural characteristics and the adsorption properties for CO_2 and hydrogen simulated by accurate but computationally demanding grand canonical Monte Carlo (GCMC) calculations. Robust structure–property relationships were found that linked structural descriptors such as pore size, surface area and void fraction to zeolite performance. Useful

catalysts were identified from the computational screening of >300 000 zeolite structures. Several of the top candidates were very promising energy-efficient templates for catalysis with predicted performance 50% higher than conventional reactors.

They subsequently showed how ML can be used to leverage GCMC calculations of hydrogen adsorption by nanoporous materials to identify their performance limits.[16] Hydrogen is an attractive transportation fuel with zero carbon emissions, but it is difficult to store. MOFs have shown promise as physical adsorbents of hydrogen. They exploited the Nanoporous Materials Genome, containing over 850 000 materials, and GCMC calculations results used to train an ML model to explore the limits of hydrogen storage. The also incorporated limited evolutionary optimization that used ML models of the fitness of materials to accelerate the discovery of higher performance examples in successive rounds of optimization. Figure 9.1 shows the results of selected GCMC calculations used to train neural net models of the fitness landscape for nanoporous materials, and improvements in performance over three optimization cycles. Maximum net capacity at room temperature was achieved with pore sizes of around 6 Å and void fractions of 0.1, while at cryogenic temperatures pore sizes of 10 Å and void fractions of 0.5 are optimal. The top candidates were commercially attractive cryo-adsorbents, with promising storage capacities at 77 K and 100 bar of 40 g L^{-1}, a promising alternative to liquefaction.

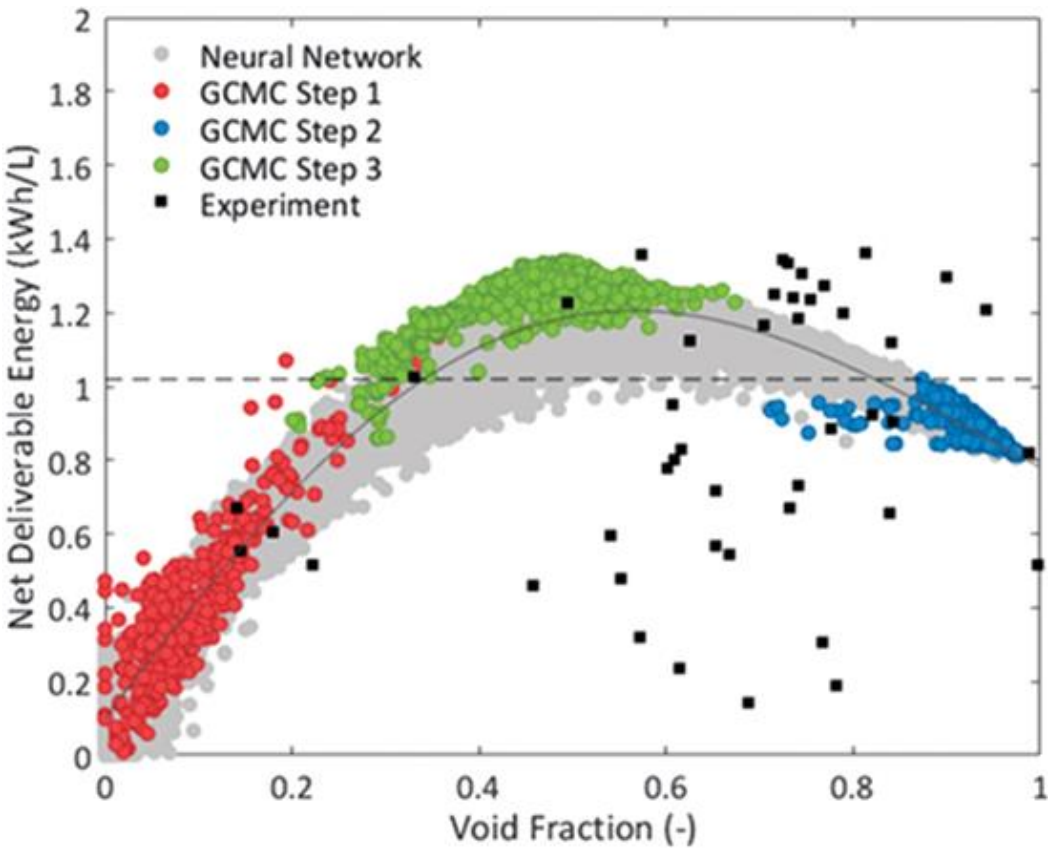


Figure 9.1 Selected GCMC calculations (grey filled circles), ML predictions (light grey points), and experimental data (black points) for net deliverable energy as a function of void fraction at 77 K, cycling between 100 and 1 bar, for the complete genome (∼850 000 porous materials). GCMC calculations were used to train neural net models of the fitness landscape for nanoporous materials that were used to generate improvements in performance over three optimization cycles. Experimental data are from the literature, the dashed line represents the predicted bare tank performance, and the solid dark line is the fitted Langmuir model. Reproduced from ref. 16, https://pubs.acs.org/doi/full/10.1021/acs.chemmater.6b04933, with permission from American Chemical Society, Copyright 2017.

2D materials are another very strong research area of contemporary research. These materials are essentially 1 atom thick; layered materials derived from them exhibit novel and useful electro-optic properties. They are also being developed as components of wearable and implantable electronics such as continuous sensors. Their properties can be predicted using computationally expensive quantum chemical calculations. Recently a flurry of papers has shown that ML models can be used to predict properties of large libraries of 2D and layered 2D materials orders of magnitude faster than QM calculations, with similar accuracy.

There are ∼2000 2D materials known or predicted to be synthesizable, and they exhibit a wide diversity of potentially useful properties. For example, a recent study[17] identified 1825 2D materials that could be exfoliated from known inorganic compounds. Hybrid layered 2D materials (two layers of different 2D materials) are of great interest for photovoltaic and photonic applications, as they have much larger scope for being tuned in application-specific ways. Known 2D materials can theoretically generate ∼2 million bilayer structures, and around a billion trilayers. Clearly, such a massive number of hybrid materials cannot be generated and tested experimentally and, even though DFT calculations can predict their properties, the computational demands make it also infeasible to predict the properties of more than about a hundred materials. ML methods can be used to leverage a relatively small number of accurate DFT calculations of hybrid 2D materials properties into a much larger material space. For example, Tawfik *et al.* reported the prediction of the interlayer distance and band gap for almost 1500 bilayer heterostructures. Their models could predict the interlayer spacing for hybrid materials in a test set with a mean error of 4% and an RMSE of 0.2 eV for the band gap. Use of hybrid DFT and ML methods allow predictions to be made for large numbers of materials much faster than those from DFT calculations alone.

9.4 Machine Learning Examples in Nanomaterials

Materials in nanoparticulate form often exhibit useful (but also potentially detrimental) biological properties compared to the same material in the bulk phase. This is largely due to the extremely high surface to volume ratio of small particles compared to larger ones. Materials surfaces interact directly with biology so are very important. Nanomaterials have shown great promise in delivering drugs and gene therapies to target tissues in the body, and as imaging agents for diagnosis. ML methods have been used quite widely to understand the relationship between nanoparticle physicochemical properties and how they affect biology. Computational models of nanoparticle behaviour are very difficult to devise for several key reasons: nanoparticles exhibit a distribution of sizes and shapes, so are more difficult to characterize than small molecules; they are much larger than molecules, so it is not clear how to encode their chemical properties in meaningful ways; in biological systems they become coated by complex layers of proteins (the corona) that significantly affect the

types of biological responses that they generate; experiments to map their distribution *in vivo*, uptake through lungs, GI tract, and skin are very difficult; and although *in vitro* assays are easier they often do not correlate well with *in vivo* responses.

ML methods can simplify some of these difficulties and they do not require a complete understanding of mechanisms involved in nanoparticle biological responses. They have been an important element in understanding the potential adverse biological properties of nanomaterials used commercially. The aim is to use ML models as part of a 'safe-by-design' paradigm to generate commercially useful materials with minimal adverse effects. For example, Le *et al.* described an experimental and computational approach for the development of nanoparticles that are safe-by-design.[18] They generated a library of 45 ZnO nanoparticles with varying particle size, aspect ratio, doping type, doping concentration, and surface coatings and measured cell damage in human umbilical vein endothelial cells or human hepatocellular liver carcinoma cells. They successfully generated quantitative and predictive ML models mapping nanoparticle properties to cell viability, membrane integrity, and oxidative stress. Figure 9.2 shows the results of ML models for these potentially adverse biological effects of nanomaterials as function of the size, shape, core material, doping, and coating type.

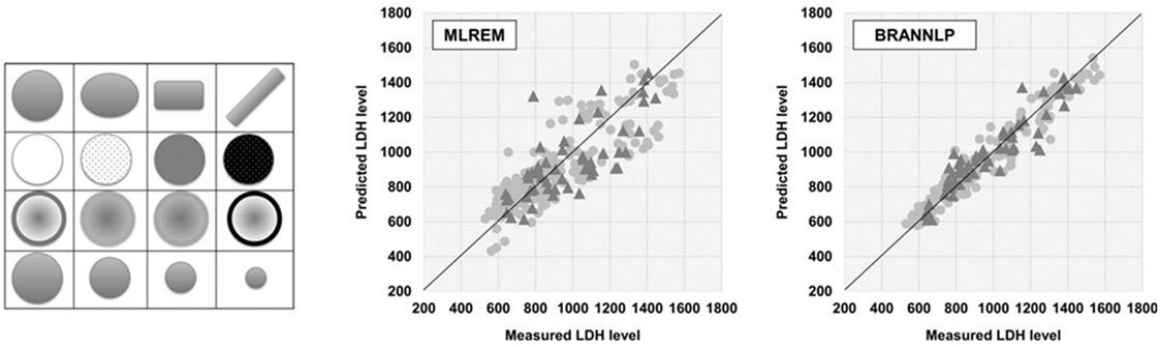


Figure 9.2 Performance of ML models for LDH level (a measure of cell membrane integrity, right) for a range of ZnO-based nanomaterials with different sizes, shapes, core materials, doping, and coating types (left). Reproduced from ref. 18 with permission from John Wiley and Sons, © 2016 WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim.

The most important factors driving cell damage were the concentration of nanoparticles the cells were exposed to, the type of surface coating, the nature and extent of the ZnO core doping, and the aspect ratio of the particles. Oksel *et*

al. also reported several ML models of nanoparticle biological properties and recently described the application of random forest ML methods to modelling nanoparticle properties.[19] They found RF methods generated models that were easier to interpret in terms of ranges and limits of nanoparticle properties that most influence any adverse effects of nanomaterials.

As ML methods are data-driven, one of the main issues with them is obtaining sufficient high-quality experimental data to train them. This is particularly difficult with nanomaterials, as there are still relatively few high throughput synthesis and characterization studies on these materials, and they are more difficult to study experimentally. This urgent need was identified in recent reviews on the application of ML methods to predicting the adverse properties of nanomaterials[20,21] and in a paper calling for more systematic studies of these properties.[22]

Amphiphilic, self-assembling nanomaterials such cubosomes and hexosomes are a promising class of drug delivery materials (see Figure 9.3). However, they are tricky to design as the types of nanophases they assemble into can be dependent on the temperature, time, drug loading and drug physicochemical properties, amongst others. Only specific nanophases are useful as delivery vehicles. Predicting the effects of these complex parameters on the resultant nanophase was considered intractable until recently. Le at al. showed how ML methods can accurately predict which of several (sometimes coexisting) nanophases will form for a given set of conditions.[23,24] They validated their ML models predicting nanophases by a subsequent *de novo* synthesis of loaded amphiphilic nanoparticles and experimental determination of the resulting nanophase using small angle x-ray scattering on the Australian synchrotron (see Figure 9.4). The predictions were 90% accurate. They subsequently showed how ML models could accurately predict the time evolution of nanoparticulate amphiphilic drug delivery systems.[25]

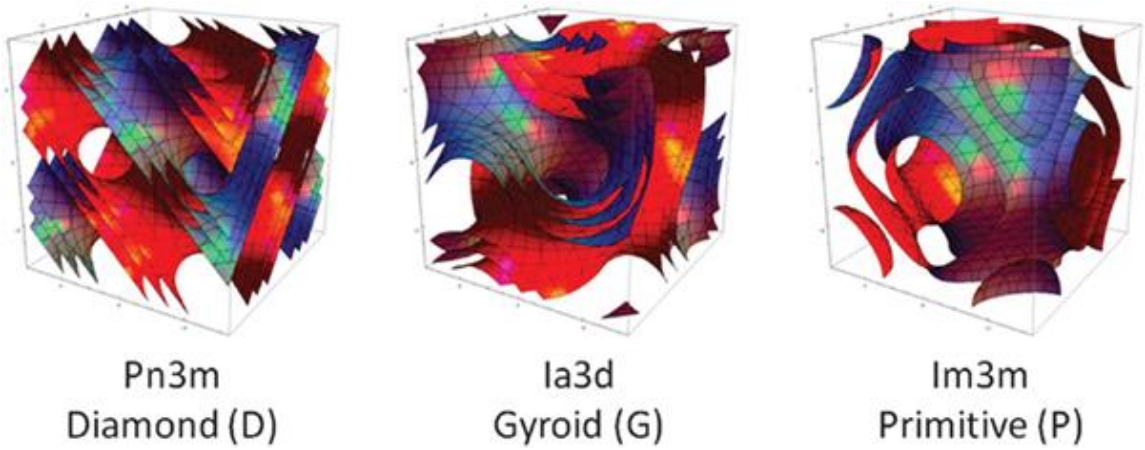
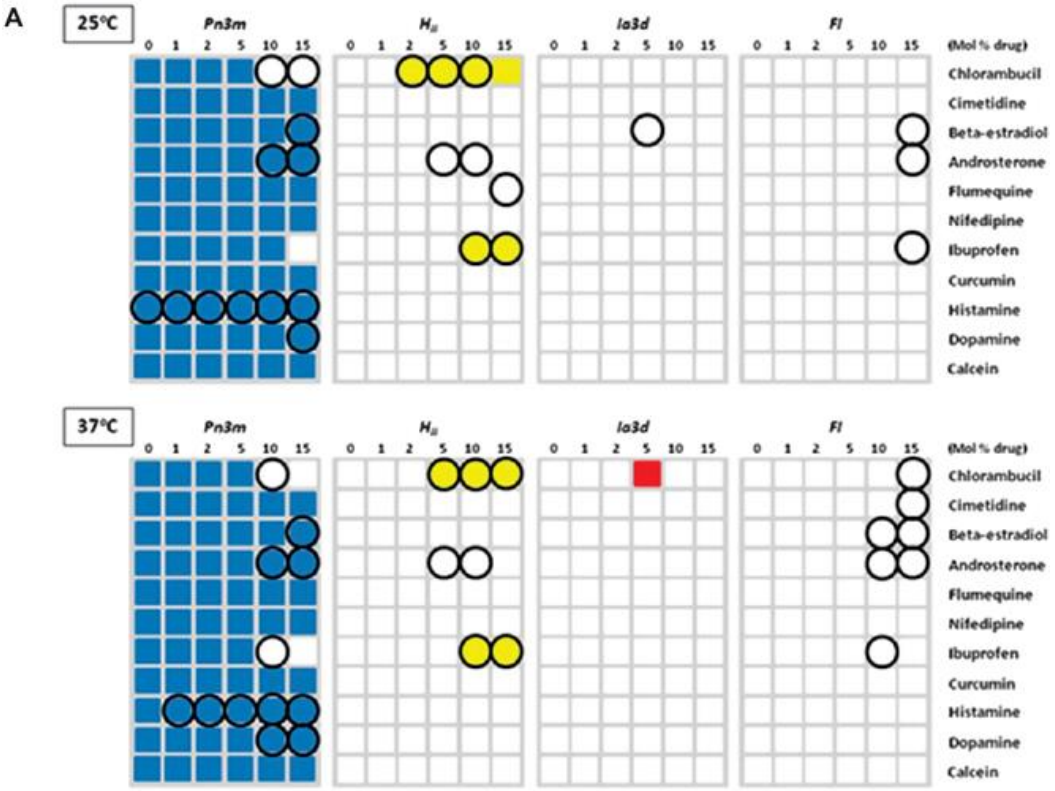


Figure 9.3 Amphiphilic, self-assembling nanomaterials, such cubosomes and hexosomes, adopt many kinds of nanostructures, not all of which are suitable as drug delivery vehicles. Here, we show example structures of the inverse-bicontinuous diamond QIID (Pn3m), gyroid QIIG (Ia3d), and primitive QIIP (Im3m) cubic phases. Reproduced from ref. 23, https://pubs.acs.org/doi/10.1021/mp3006402, with permission from American Chemical Society, Copyright 2013.


A
25°C
Pn3m
H_II
Ia3d
Fl
0 1 2 5 10 15
(Mol % drug)
Chlorambucil
Cimetidine
Beta-estradiol
Androsterone
Flumequine
Nifedipine
Ibuprofen
Curcumin
Histamine
Dopamine
Calcein
37°C
Pn3m
H_II
Ia3d
Fl
0 1 2 5 10 15
(Mol % drug)
Chlorambucil
Cimetidine
Beta-estradiol
Androsterone
Flumequine
Nifedipine
Ibuprofen
Curcumin
Histamine
Dopamine
Calcein

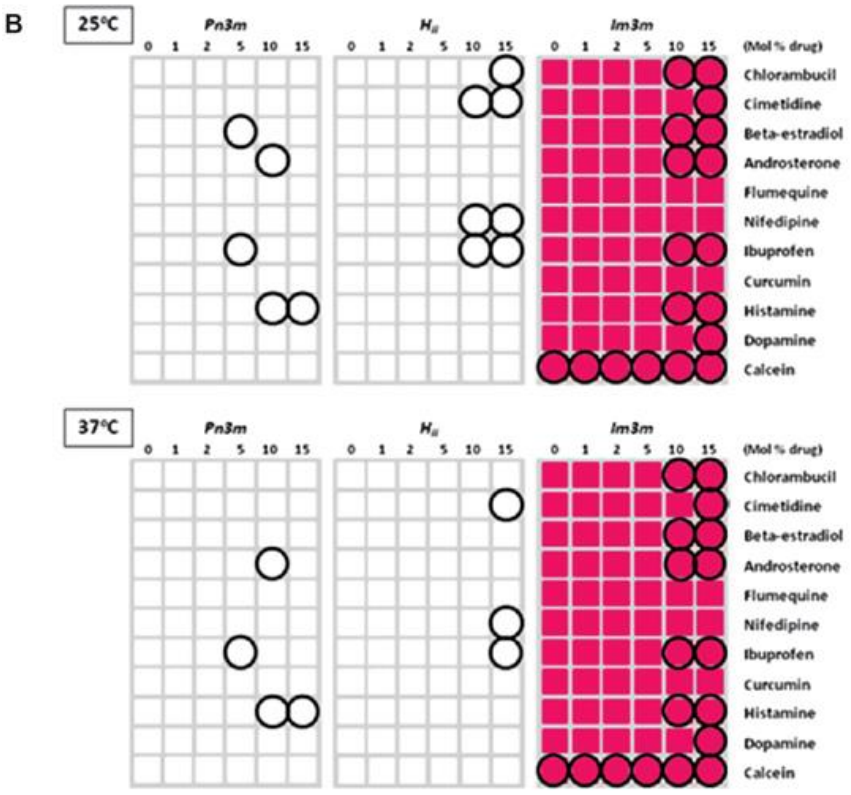

B
25°C
Pn3m
H_II
Im3m
0 1 2 5 10 15
(Mol % drug)
Chlorambucil
Cimetidine
Beta-estradiol
Androsterone
Flumequine
Nifedipine
Ibuprofen
Curcumin
Histamine
Dopamine
Calcein
37°C
Pn3m
H_II
Im3m
0 1 2 5 10 15
(Mol % drug)
Chlorambucil
Cimetidine
Beta-estradiol
Androsterone
Flumequine
Nifedipine
Ibuprofen
Curcumin
Histamine
Dopamine
Calcein

Figure 9.4 ML models predicting the existence of individual phases (inverse-bicontinuous diamond QIID (Pn3m) cubic (blue, to the left of Figure 9.4A and B), hexagonal HII (yellow, filled circles), gyroid QIIG (Ia3d) cubic (red), primitive QIIP (Im3m) cubic (purple, to the right in Figure 9.4C and D), and fluid isotropic FI (brown)) for a range of new drugs incorporated in the nanophases. Amphiphiles used to generate the nanoparticles were (A) phytantriol and (B) monoolein nanoparticles. These particles were loaded with new drugs at 25 and 37 °C. Circles indicate incorrect predictions for the nanophases determined subsequently by small angle X-ray scattering at the Australian Synchrotron. The predictions were 90% accurate. This showed how ML models could accurately predict nanophase behaviour as a function of amphiphile, drug type, drug loading, and temperature. Reproduced from ref. 23, https://pubs.acs.org/doi/10.1021/mp3006402, with permission from American Chemical Society, Copyright 2013.

siRNA can also be used for therapeutic purposes and ML has been used to design siRNA libraries and more effective materials to deliver these highly charged molecules. A 2011 review described the various ML methods used to assist with the rational design of siRNA sequences and delivery systems.[26] For example, Huesken and coworkers trained a neural network on 2182 randomly selected siRNAs targeted to 34 mRNA species using a high-throughput reporter gene system.[27] The ML model predicted the activity of 249 siRNAs in an independent test set with a modest r^2 value of 0.44. This ML model was used to design a genome-wide siRNA library consisting of two siRNAs per gene. This collection of 50 000 siRNAs identified genes involved in the cellular response to hypoxia. The dramatic increase in the capability of ML methods subsequently has reignited interest in this important area. The prediction of efficacy of siRNA sequences for gene silencing has improved significantly and ML methods are making much better predictions on the efficiency of nanoparticulate gene delivery systems. For example, more recently Han *et al.* used deep learning methods to improve the accuracy of prediction of siRNA efficacy. Their DNN extracted hidden features from sequences and thermodynamic properties. The model predicted siRNA efficacy slightly better, with an r^2 value of 0.53. Subsequent experiments showed that their model was more accurate than several other siRNA prediction methods. Anderson's team at MIT reported the effects of a small library of nanoparticles synthesized from lipid-modified aminoglycosides and siRNA.

9.5 Machine Learning Examples in Biomaterials and Regenerative Medicine

Biomaterials are an important element of regenerative medicine that promises hitherto inaccessible treatments for injury and disease. Biomaterials can replace damaged tissues, stimulate the growth of nerves, provide environments for implanted cells in cell therapies aimed at generating essential hormones such as insulin, replace bone damaged by accidents or disease, precisely deliver drugs and gene therapy agents to diseased tissues, and hide otherwise biologically incompatible implanted components such as pacemakers and cochlear implants from the body's immune response. Given the almost infinite variety of chemically accessible materials to choose from, it is essential to conduct experiments to select them using high throughput techniques, and to employ powerful computational methods such as ML and evolutionary algorithms to extract design rules from large data sets and to explore materials space as efficiently as possible. To date, the materials most widely used for these purposes are polymers and nanomaterials.

Given the rapid growth in implantable and indwelling medical devices, it is particularly important to design or discover materials that resist bacterial attachment and biofilm formation, and those that reduce or eliminate the foreign body response (FBR) that often causes device failure. FBR to an implanted device generates a fibrous capsule that insulates the device or implanted cell therapy from the body, compromising function. Clearly, materials surfaces and interfaces are often the main drivers of biological response, so the ability to analyze, modify and model the chemistry and topography of surfaces is very important for the development and optimization of new biomaterials. However, interactions between surfaces and biology are poorly understood at the molecular and mechanistic level.

ML methods have been used to model the propensity of polymers to support the growth of pathogenic microbes. Clever generation of large arrays of different polymers using dip pen and inkjet printing methods has provided the means for measuring the responses of diverse cell types to the polymers. Building on the seminal work from the Langer group at MIT, Hook *et al.* described this experimental technique in a recent publication.[12,28] They found relationships between the degree of attachment of three important hospital pathogens, Staphylococcus aureus, Pseudomonas aeruginosa, and uropathogenic Escherichia coli, and surface chemistry (as determined by experimental surface analysis) for several hundred diverse polymers. Subsequent ML modelling of these data using surface analysis data as descriptors, or computational generated molecular descriptors for the

monomers, resulted in robust and predictive models of pathogen attachment.[29] A subsequent study showed how a single ML model could simultaneously predict the attachment of all three pathogens,[30] making it much simpler to conduct virtual screens of thousands of accessible but not yet synthesized polymers to identify those with low general pathogen attachment (summarized in Figure 9.5).

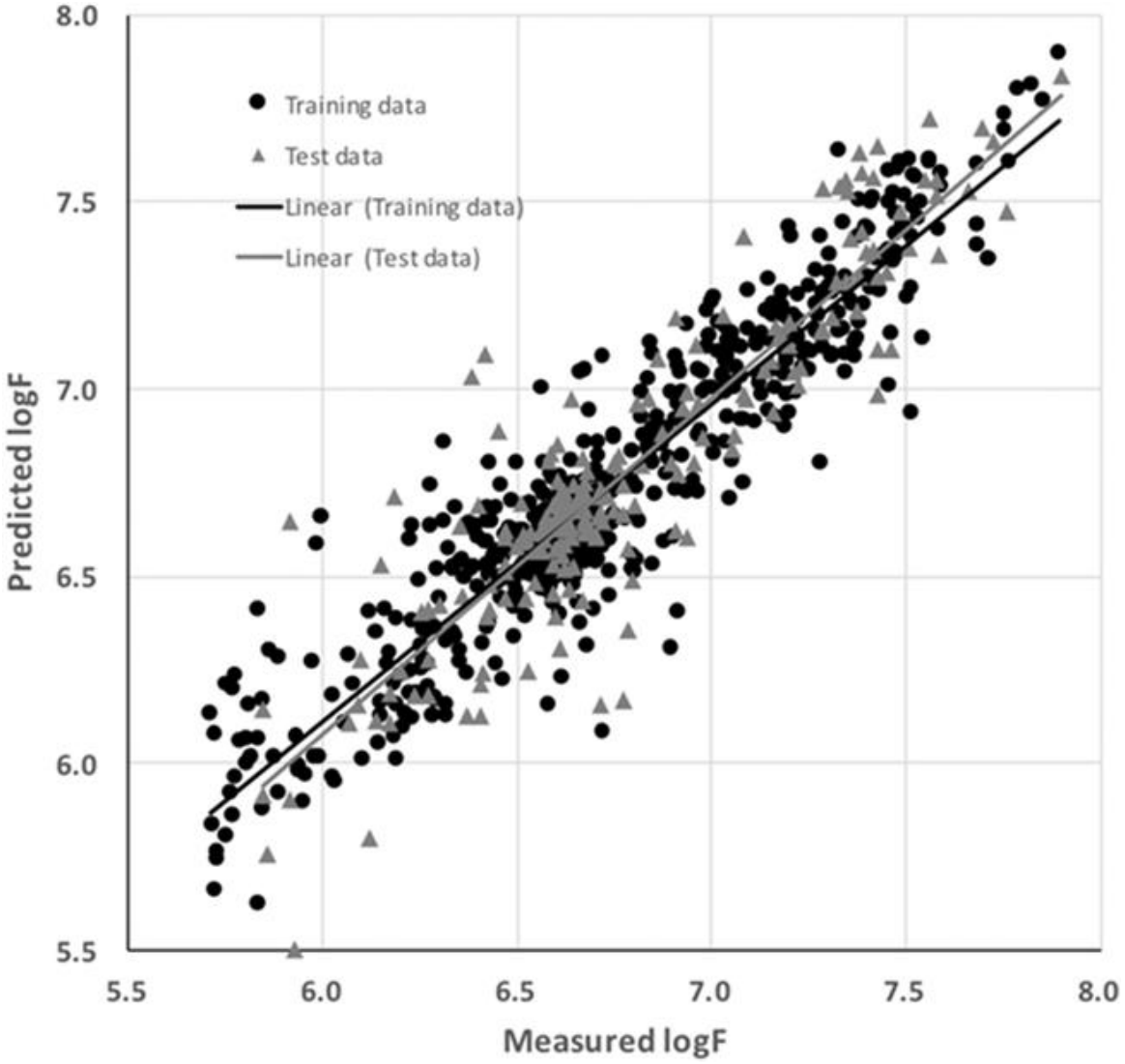


Figure 9.5 A single ML model simultaneously predicts the attachment of three important hospital-acquired pathogens,[30] Measured and predicted attachment (estimated using the log of the GFP fluorescence, logF) of the multi-pathogen attachment model employed computed molecular descriptors. Reproduced from ref. 30 with permission from American Chemical Society, Copyright 2018.

As ML models of biomaterials (and drugs too) have historically been trained using efficient but rather arcane (hard to understand) descriptors, they subsequently showed how a specific type of molecular fragment descriptor could generate models that were easier for experimental materials scientists to interpret.[31] A similar ML modelling approach was used to find relationships between the attachment of stem cells to polymer libraries.[32,33] More recently, ML methods have been used to model a library of 141 homopolymers and 400 diverse copolymers for their abilities to support attachment, proliferation, and differentiation of human dental pulp stem cells.[33] In this study, a Bayesian regularized neural network with a sparsity inducing Laplacian prior was used to model the three stem cell responses separately. Computational models were generated, relating polymer properties to cellular responses. The models correctly predicted the attachment of stem cells 92–95% of the time, proliferation 85–92% of the time, and differentiation 82–85% of the time. The models also identified several key functional groups that support specific cell responses.

Surface analysis methods like time of flight secondary ion mass spectrometry (ToF-SIMS) and Raman spectroscopy are very important for understanding how biomaterials interact with biology. This is because the surfaces of bulk materials and nanomaterials are the primary sites of contact with cells and other biological entities. ToF-SIMS impacts materials surfaces with atoms or clusters of atoms to fragment the constituent molecules, providing a kind of fingerprint that is specific to the materials surface. Surprisingly, analysis methods for the large data sets that surface analysis methods generate have involved relatively simple statistical methods like principle components analysis (PCA). When used as mathematical descriptors that encode surface chemistry, ion peaks from ToF-SIMS, for example, are also very useful for generating quantitative relationships between surface chemistry and biological responses.[12,28] Recently, we and others have employed more sophisticated informatics and ML methods to analyze these data sets and have shown that much more information can be obtained this way. For example, a recent study found that self-organizing maps (SOMs), a specialized form of neural network used for nonlinear clustering of complex data, are more effective at extracting information from data from ToF-SIMS experiments on polymers than principal components analysis (PCA), which is commonly employed.[34,35]Figure 9.6 shows how SOMs can distinguish between nylon samples with very similar chemical structures, a task that PCA cannot achieve. The nonlinear neural network clustering methods could detect

small quantities of materials such as antibodies better than traditional methods but could also discriminate between materials with high degrees of chemical similarity.[34,36]

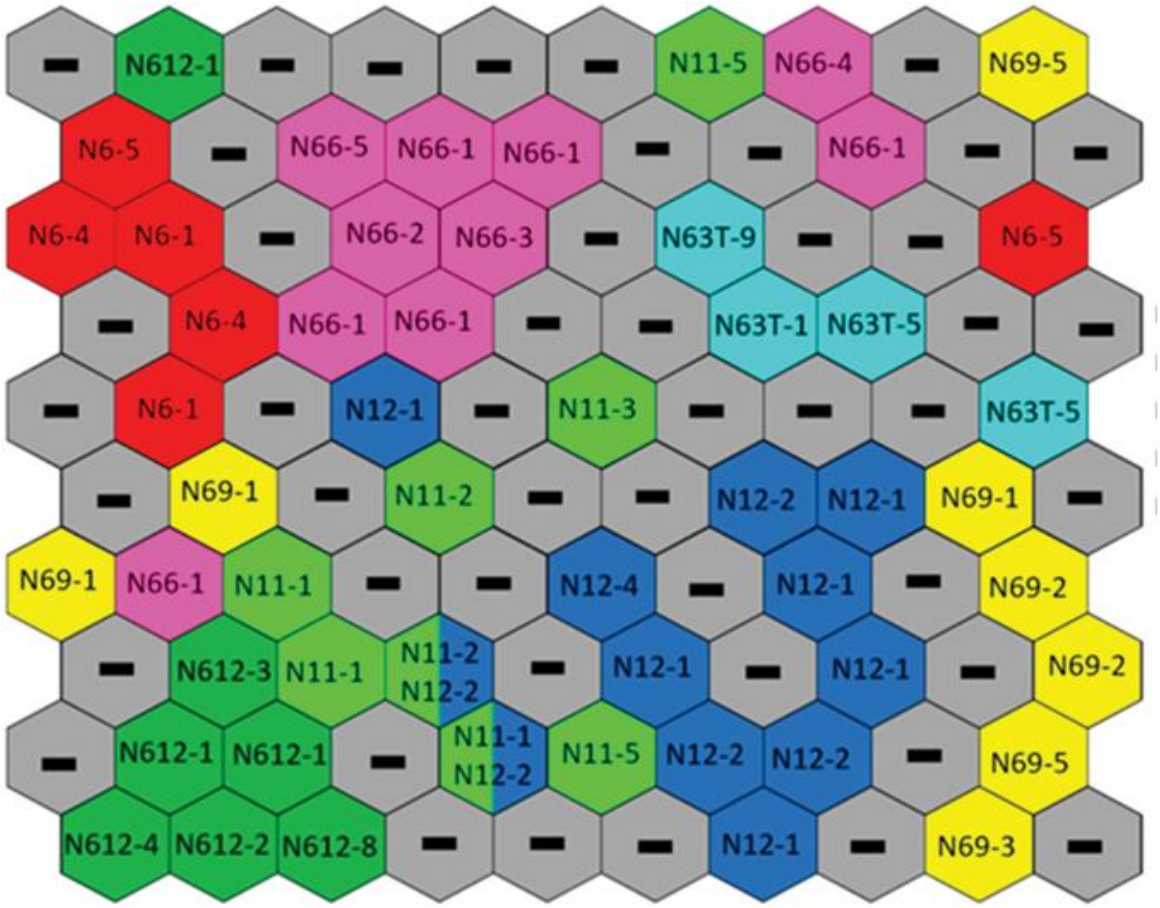


Figure 9.6 Discrimination between nylon samples with very similar chemical structures using Self-Organizing Maps (SOMs). The 10×10 supervised Kohonen network (SKN) was calculated for positive ion ToF-SIMS data. False-colouring is used to clarify sample locations on the network according to their sample group – see online version of the figure: (red, Nylon-6; green, Nylon-11; navy, Nylon-12; cyan, Nylon-6(3)T; pink, Nylon-6/6; yellow, Nylon-6/9; and dark green, Nylon-6/12). This ML model had a 98% class assignment accuracy. Reproduced from ref. 34 with permission from American Chemical Society, Copyright 2018.

We and others have also shown the importance of using modern sparse feature selection methods to remove chemical features from models that do not drive biological responses. These methods remove (set to zero) any features that are less relevant to the biological response using a variety of mathematical methods; the most widely used are the LASSO regression and multiple linear

regression using an expectation algorithm and a sparse Bayesian prior.[37,38] This makes models more interpretable, and also improves the predictive power of them because optimally sparse models generalize the best to new data. These methods have been very effective in identifying small numbers of genes in gene expression microarray data that correlate with biological responses. Tens of thousands of genes on a microarray can be reduced in a context-dependent way to a handful of those most relevant. For example, Autefage and co-workers used gene expression data from experiments in which mesenchymal stem cells were driven down the osteogenic (bone) pathway by strontium ions leaching from an implanted biomaterial. These implants have been shown to slow the loss of calcium in bone, reducing fractures and osteoporosis. Subsequent analysis using sparse feature selection identified a small number of related fatty acid and steroid biosynthesis genes as being important in osteogenesis. Subsequent experimental testing of these predictions using qPCR, lipid raft and gene expression measurements validated the importance of these hitherto unknown pathways for strontium-induced differentiation of MDS down the osteogenic pathway.[39] In another study, stem cells were caused to divide symmetrically or asymmetrically using a variety of chemical and physical methods. In symmetric division, they divide into two stem cells and in asymmetric division they divide into one stem cell and one progenitor cell. This division process was a long-standing unexplained problem in stem cell biology. Subsequent analysis of the gene expression microarray profiles identified 5 potential markers for the symmetry of stem cell division. Experimental validation using fluorescent antibodies for these gene products shows that these genes were indeed markers of symmetry of division (see Figure 9.7).[40] Some of these markers are now sold commercially by Asymmetrex.

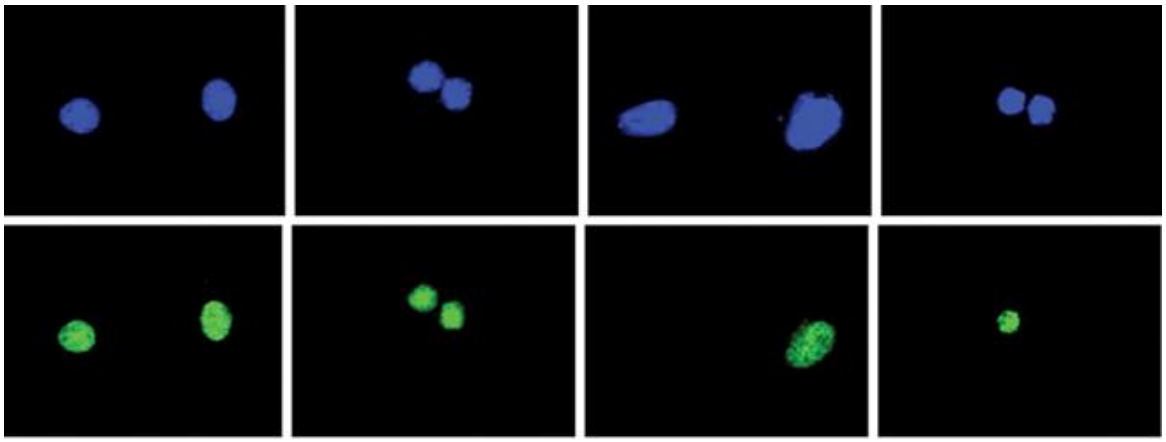


Figure 9.7 Experimental validation using fluorescent antibodies for these gene products shows that these genes were indeed markers of symmetry of division. The top row shows the positions of cell nuclei marked by DAPI stain, the lower panels show the marker H2A.Z that only identifies one of the two cells when asymmetric division has occurred (lower right two panels) and both cells when symmetric division occurs (lower left two panels). Reproduced from ref. 40 with permission from Elsevier, Copyright 2015.

9.6 Materials for Cell Therapies, Bioreactors, and Implantable Cells

Cell therapies (implanting of therapeutic cells into the body) are important medical interventions *e.g.* bone marrow transplants for leukaemia (current) and islet cells for diabetes (future). Key to the success of these therapies is the ability to culture large quantities of cells in artificial bioreactors.[41,42] Materials science and ML methods are playing increasingly important roles. For example, ML methods successfully modelled the relationships between bioreactor parameters such as initial cell type and seeding density, cytokine concentrations, reactor types, media *etc.* on subsequent cell fold increases (how many times the initial seed concentration of cell increased).[1] This study by Winkler and Burden modelled the increase in total number of cells and 5 different stem cell or progenitor cell types belonging to the haematopoietic (blood) differentiation pathway. Data from over 300 bioreactor experiments were successfully and quantitatively modelled using MLR and sparse Bayesian neural network methods. Apart from predicting the fold increase for each cell type within a factor of two (three for nucleated cells), their sparse feature selection methods also identified the most important chemical factors and bioreactor parameters driving each type of cell expansion, very useful information for controlling the specific types of cells the bioreactors generate.

9.7 ML and Evolutionary Methods

Although ML is attracting most of the attention, as we stated in the Introduction, the estimated 10^{100} possible accessible materials mean that even the most optimistic predictions of the capabilities of high throughput experimentation and ML modelling suggest that these methods cannot possibly explore more than a minute fraction of this effectively infinite space unaided.

Evolutionary algorithms (EA) are one of the most efficient ways to explore very large search spaces. It is therefore logical that evolutionary methods be employed to explore materials spaces better to discover entirely novel materials and to optimize those already discovered. There have been a limited number of studies on using evolutionary methods to 'evolve' materials, primarily catalysts and phosphors, as these represent 'low hanging fruit', more easily represented as materials 'genomes'. Applications of evolutionary methods to bioactive compounds and materials discovery are only now emerging, and progress is summarized in recent reviews.[43,44]

White light LEDs that have made possible energy-efficient lighting potentially lowers greenhouse gas emissions. These devices rely on phosphors, often made by high-temperature sintering of rare earth oxides. Phosphors are useful to demonstrate how materials 'genomes' are generated and the basic operation of *in silico* simulated Darwinian evolution. Phosphors can be very simply presented by a 'genome', for example, a vector containing the relative mole fractions of each rare earth oxide component. After sintering, phosphor brightness and colour are measured (equivalent to species fitness in Darwinian evolution) and the best phosphors are 'bred' to generate a new and higher fitness population of phosphors. Mutation operations such as elitism (passing on the best phosphors into the next generation unchanged), changing the concentration of one rare earth (point mutations), and splitting the genomes of two phosphors and combining the parts into new ones (crossover operators) are applied to the fittest materials in a given cycle of evolution. This evolutionary loop is iterated for a number of cycles, until a specified performance level is reached, or the algorithm fails to generate improved examples. In this way, Sharma and co-workers[45] steadily improved the fitness (brightness and colour) of mixed metal oxide seven-dimensional library ($MnO–Na_2O–Li_2O–MgO–ZnO–CaO–GeO_2$) by evolution to find a promising green phosphor for use in backlit liquid crystal display applications. (see Figure 9.8). They synthesized only a few hundred phosphors but explored a much larger space of phosphor composition to find locally optimum materials.

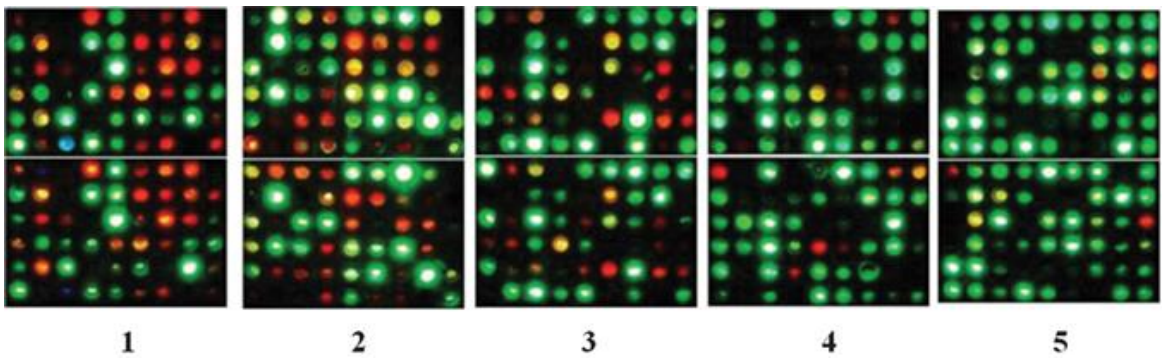


Figure 9.8 By iterating around this evolutionary loop for five cycles, Sharma *et al.* improved the fitness of the phosphors (brightness and green colour). The upper and lower panels are replicates. Reproduced from ref. 45 with permission from John Wiley and Sons, Copyright © 2009 WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim.

Consequently, it is very likely that the next disruptive technology to be adopted by materials scientists after ML will be genetic technologies. This is likely to play out in two ways: more efficient searching of materials spaces using high throughput experiments coupled with EAs, and even more effective searching using synergistic combinations of ML and EA. The latter involves building ML models of the fitness landscape from experimental data in successive initial rounds of high throughput experiments and using these to replace experimentation in later stages of the search. It has been shown that such 'adaptive' evolution (an in-silico equivalent of the biological Baldwin Effect) allows systems to evolve more quickly than those that do not use it.[46,47]

Evolutionary methods also make possible, for the first time, the possibility of autonomous discovery methods, fully automated invention machines with no 'human in the loop'. In broad terms, such a machine would consist of a general purpose synthesis robot, automated characterization instruments for quality control and to measure a specified fitness, an evolutionary program that could select the fittest members of an initial population of molecules or materials and mutate them into a new population, and a method for specifying the new molecules to be synthesized to the robotic synthesizer. Several groups are developing autonomous chemists and autonomous materials scientists. Some of these are focused on automating organic synthesis,[48] and others materials discovery.[49] There is major work towards universal organic synthesis machines, a prerequisite for an autonomous organic chemist, at the Max Planck Institute for Colloids and Interfaces in Potsdam. The US Defense Advanced Research Projects Agency (DARPA) has funded several international initiatives. to help

remove barriers to autonomous synthesis systems. The research teams are pursuing both hardware and software solutions for automation.[50] MIT is using preloaded cartridges to automate synthetic steps such as heating, mixing and separating chemicals.[51] They trained an ML model on millions of reactions from the Reaxys database and the U.S. Patent and Trademark Office. The ML model learned to apply retrosynthetic transformations, identify relevant reaction conditions, and to estimate reaction feasibility. It was taught to generalize from specific known chemical reactions to new substrates. A team at the University of Glasgow led by Prof. Lee Cronin have developed a Chemputer, a modular desktop-sized robotic synthesizer that compiles text-based synthetic instructions into machine instructions to drive synthesis robots costing £25 000–30 000 (US$33 000–39 500) each.[52] This approach has successfully synthesized generic versions of common drugs in an order of magnitude faster than conventional manual synthesis methods. His online platform, Chemify, allows chemists to download Chemputer assembly instructions.

9.8 The Path Forward

ML is passing through at least the second of the well-known hype cycles. Claims of ML's capabilities are often exaggerated, but it is clear that there are also many examples across broad fields of science, technology, medicine, manufacturing, and business of spectacular outcomes. ML is a disruptive technology that will change the world as we know it in the next few decades. It has the potential to remove much of the tedious, repetitive, and sometimes dangerous work of chemical and materials researchers, freeing them up for more creative endeavours. The main roadblocks to future development are:

- the ability to describe complex materials mathematically to train ML methods,
- a paucity of data to train deep learning models,
- a better understanding of how to reduce bias in ML models learning from examples,
- development of chemical synthesis languages that will completely specify reactions to improve reproducibility and allow robots to be programmed with synthesis information,

- development of synthesis machines with broad applicability (*e.g.* by breaking down molecule or materials syntheses into a finite number of unit operations),
- and optimal synergy between automation, ML, and evolution.

ML and AI will also generate profound changes to the workforce, with even intellectual jobs such as accounting, law, and some aspects of medicine being impacted. As foreshadowed, the combination of AI, ML, robotics, and evolutionary methods will potentially unleash new ways of doing science that can accelerate our understanding of materials properties, their interactions with biology, and the discovery of novel molecules with highly novel and useful properties. As well as robots helping design and synthesize new molecules and materials, we are likely to see materials evolving collaboratively with robot morphologies and controllers to generate new fit-for-function robots,[11] another neat 'closing of the loop' in technology.

References

1. D. A. Winkler and F. R. Burden, *Mol. BioSyst.*, 2012, **8**, 913–920.
2. S. S. Young, F. Yuan and M. Zhu, *Mol. Inf.*, 2012, **31**, 707–710.
3. B. Sanchez-Lengeling and A. Aspuru-Guzik, *Science*, 2018, **361**, 360–365.
4. T. Fujita and D. A. Winkler, *J. Chem. Inf. Model.*, 2016, **56**, 269–274.
5. J. L. Faulon, D. P. Visco and R. S. Pophale, *J. Chem. Inf. Comp. Sci.*, 2003, **43**, 707–720.
6. A. Mauri, V. Consonni, M. Pavan and R. Todeschini, *MATCH-Commun. Math. Comp. Chem.*, 2006, **56**, 237–248.
7. I. W. Davies, *Nature*, 2019, **570**, 175–181.
8. A. Senior, J. Jumper and D. Hassabis, AlphaFold: Using AI for scientific discovery, https://deepmind.com/blog/article/alphafold.
9. S. Abdulkader Tawfik, M. J. Ford, O. Isayev, D. A. Winkler and C. Stampfl, *Adv. Theory Simul*, 2019, **2**, 1800128.
10. T. Le, V. C. Epa, F. R. Burden and D. A. Winkler, *Chem. Rev.*, 2012, **112**, 2889–2919.
11. D. Howard, A. E. Eiben, D. F. Kennedy, J.-B. Mouret, P. Valencia and D. A. Winkler, *Nat. Mach. Intell.*, 2019, **1**, 12–19.
12. A. L. Hook, C. Y. Chang, J. Yang, J. Luckett, A. Cockayne, S. Atkinson, Y. Mei, R. Bayston, D. J. Irvine, R. Langer, D. G. Anderson, P. Williams,

M. C. Davies and M. R. Alexander, *Nat. Biotechnol.*, 2012, **30**, 868–U899.
13. S. M. Moosavi, A. Chidambaram, L. Talirz, M. Haranczyk, K. C. Stylianou and B. Smit, *Nat. Commun.*, 2019, **10**, 539.
14. P. Z. Moghadam, S. M. J. Rogge, A. Li, C.-M. Chow, J. Wieme, N. Moharrami, M. Aragones-Anglada, G. Conduit, D. A. Gomez-Gualdron, V. Van Speybroeck and D. Fairen-Jimenez, *Matter*, 2019, **1**, 219–234.
15. A. W. Thornton, D. A. Winkler, M. S. Liu, M. Haranczyk and D. F. Kennedy, *RSC Adv.*, 2015, **5**, 44361–44370.
16. A. W. Thornton, C. M. Simon, J. Kim, O. Kwon, K. S. Deeg, K. Konstas, S. J. Pas, M. R. Hill, D. A. Winkler, M. Haranczyk and B. Smit, *Chem. Mater.*, 2017, **29**, 2844–2854.
17. N. Mounet, M. Gibertini, P. Schwaller, D. Campi, A. Merkys, A. Marrazzo, T. Sohier, I. E. Castelli, A. Cepellotti, G. Pizzi and N. Marzari, *Nat. Nanotechnol.*, 2018, **13**, 246–252.
18. T. C. Le, H. Yin, R. Chen, Y. D. Chen, L. Zhao, P. S. Casey, C. Y. Chen and D. A. Winkler, *Small*, 2016, **12**, 3568–3577.
19. C. Oksel, D. A. Winkler, C. Y. Ma, T. Wilkins and X. Z. Wang, *Nanotoxicology*, 2016, **10**, 1001–1012.
20. D. A. Winkler, *Toxicol. Appl. Pharmacol.*, 2016, **299**, 96–100.
21. D. A. Winkler, E. Mombelli, A. Pietroiusti, L. Tran, A. Worth, B. Fadeel and M. J. McCall, *Toxicology*, 2013, **313**, 15–23.
22. X. Bai, F. Liu, Y. Liu, C. Li, S. Q. Wang, H. Y. Zhou, W. Y. Wang, H. Zhu, D. A. Winkler and B. Yan, *Toxicol. Appl. Pharmacol.*, 2017, **323**, 66–73.
23. T. C. Le, X. Mulet, F. R. Burden and D. A. Winkler, *Mol. Pharmaceutics*, 2013, **10**, 1368–1377.
24. T. C. Le, C. E. Conn, F. R. Burden and D. A. Winkler, *Cryst. Growth Des.*, 2013, **13**, 3126–3137.
25. T. C. Le, C. E. Conn, F. R. Burden and D. A. Winkler, *Cryst. Growth Des.*, 2013, **13**, 1267–1276.
26. J. O. Ebalunode, C. Jagun and W. Zheng, *Methods Mol. Biol.*, 2011, **672**, 341–358.
27. D. Huesken, J. Lange, C. Mickanin, J. Weiler, F. Asselbergs, J. Warner, B. Meloon, S. Engel, A. Rosenberg, D. Cohen, M. Labow, M. Reinhardt, F. Natt and J. Hall, *Nat. Biotechnol.*, 2005, **23**, 995–1001.
28. A. L. Hook, C. Y. Chang, J. Yang, S. Atkinson, R. Langer, D. G. Anderson, M. C. Davies, P. Williams and M. R. Alexander, *Adv. Mater.*, 2013, **25**, 2542–2547.

29. V. C. Epa, A. L. Hook, C. Chang, J. Yang, R. Langer, D. G. Anderson, P. Williams, M. C. Davies, M. R. Alexander and D. A. Winkler, *Adv. Funct. Mater.*, 2014, **24**, 2085–2093.
30. P. Mikulskis, A. L. Hook, M. H. Alexander and D. A. Winkler, *ACS Appl. Mater. Interfaces*, 2018, **10**, 139–149.
31. P. Mikulskis, M. Alexander and D. A. Winkler, *Advanced Intelligent Systems*, 2019.
32. V. C. Epa, J. Yang, Y. Mei, A. L. Hook, R. Langer, D. G. Anderson, M. C. Davies, M. R. Alexander and D. A. Winkler, *J. Mater. Chem.*, 2012, **22**, 20902–20906.
33. S. R. Ghaemi, B. Delalat, S. Gronthos, M. R. Alexander, D. A. Winkler, A. L. Hook and N. H. Voelcker, *ACS Appl. Mater. Interfaces*, 2018, **10**, 38739–38748.
34. R. M. T. Madiona, S. E. Bamford, D. A. Winkler, B. W. Muir and P. J. Pigram, *Anal. Chem.*, 2018, **90**, 12475–12484.
35. R. M. T. Madiona, D. A. Winkler, B. W. Muir and P. J. Pigram, *Appl. Surf. Sci.*, 2019, **487**, 773–783.
36. R. M. T. Madiona, N. Welch, B. W. Muir, D. A. Winkler and P. J. Pigram, *Biointerf*, 2019, **14**, 061002.
37. F. R. Burden and D. A. Winkler, *QSAR Comb. Sci.*, 2009, **28**, 645–653.
38. R. Tibshirani, *J. R. Stat. Soc. Series B*, 1996, **58**, 267–288.
39. H. Autefage, E. Gentleman, E. Littmann, M. A. B. Hedegaard, T. Von Erlach, M. O'Donnell, F. R. Burden, D. A. Winkler and M. M. Stevens, *Proc. Natl. Acad. Sci. U. S. A.*, 2015, **112**, 4280–4285.
40. Y. H. Huh, M. Noh, F. R. Burden, J. C. Chen, D. A. Winkler and J. L. Sherley, *Stem Cell Res.*, 2015, **14**, 144–154.
41. A. Celiz, J. Smith, A. Patel, A. Hook, D. Rajamohan, V. George, M. Patel, V. Epa, T. Singh, R. Langer, D. Anderson, N. Allen, D. Hay, D. Winkler, D. Barrett, M. Davies, L. Young, C. Denning and M. Alexander, *Tissue Eng., Part A*, 2015, **21**, S270.
42. A. D. Celiz, J. G. W. Smith, R. Langer, D. G. Anderson, D. A. Winkler, D. A. Barrett, M. C. Davies, L. E. Young, C. Denning and M. R. Alexander, *Nat. Mater.*, 2014, **13**, 570–579.
43. T. C. Le and D. A. Winkler, *Chem. Rev.*, 2016, **116**, 6107–6132.
44. T. C. Le and D. A. Winkler, *ChemMedChem*, 2015, **10**, 1296–1300.
45. A. K. Sharma, C. Kulshreshtha and K.-S. Sohn, *Adv. Funct. Mater.*, 2009, **19**, 1705–1712.

46. G. E. Hinton and S. J. Nowlan, in *Adaptive Individuals in Evolving Populations*, ed. K. B. Richard and M. Melanie, Addison-Wesley Longman Publishing Co., Inc., 1996, pp. 447–454.
47. J. M. Smith, *Nature*, 1987, **329**, 761–762.
48. V. Dragone, V. Sans, A. B. Henson, J. M. Granda and L. Cronin, *Nat. Commun.*, 2017, **8**, 15733.
49. F. Häse, L. M. Roch and A. Aspuru-Guzik, *Trends Chem.*, 2019, **1**, 282–291.
50. K. Sanderson, *Nature*, 2019, **568**, 577–579.
51. C. W. Coley, D. A. Thomas, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen, V. Schultz, T. Hart, J. S. Fishman, L. Rogers, H. Gao, R. W. Hicklin, P. P. Plehiers, J. Byington, J. S. Piotti, W. H. Green, A. J. Hart, T. F. Jamison and K. F. Jensen, *Science*, 2019, **365**, eaax1566.
52. S. Steiner, J. Wolf, S. Glatzel, A. Andreou, J. M. Granda, G. Keenan, T. Hinkley, G. Aragon-Camarasa, P. J. Kitson, D. Angelone and L. Cronin, *Science*, 2019, **363**, eaav2211.

CHAPTER 10

Machine Learning Techniques Applied to a Complex Polymerization Process

SILVIA CURTEANU(0000-0002-5281-1265)

"Gheorghe Asachi" Technical University of Iasi, Faculty of Chemical Engineering and Environmental Protection "Cristofor Simionescu", Iasi, Romania
Email: silvia_curteanu@yahoo.com

10.1 Difficulties in Chemical Process Modeling

In chemical reactions engineering, there are many processes and systems for which the physical and chemical laws that govern them are not known, or limited, or only uncertain knowledge is available. In such cases, either phenomenological models cannot be developed or those models are subject to significant errors because of the approximations included. Empirical modeling, based on experimental data, then becomes a viable tool, with a real chance of providing accurate models. Artificial neural networks, belonging to the machine learning (ML) domain, fall into this category, operating on input–output data in a "black box" fashion.

In complex, highly non-linear, chemical processes, in addition to the difficulty of obtaining a mechanistic model, further problems arise related to solving the mathematical models, which are often represented by systems of differential equations. Modeling the output–input dependence using neural networks represents a possible way in which one might avoid the problem of solving intractable mathematical models.

Another major difficulty in controlling the quality of products in a chemical synthesis is the lack of "online" measurements for some variables and the substantial delays that may accompany other measurements.

Further difficulties in process modeling exist in the complex field of polymerization processes. Polymer products often contain numerous specially added materials to enhance their use characteristics. The most widely used process for obtaining these materials is discontinuous polymerization, which is also useful if it becomes necessary to change the production specifications for the manufacture of polymers with different properties. Discontinuous operation is a dynamic process, with reaction conditions often varying in a fairly wide range. Consequently, both data collection for modeling and configuring the reactors present problems.

In contrast to discontinuous processes, continuous polymerizations benefit from the availability of significant amounts of data and the possibility of making precise adjustments to

process conditions, corresponding to narrow operating ranges of the key adjustable parameters.[1]

One of the most important features of free radical polymerization is the significant increase in viscosity that occurs as the reaction proceeds, especially in homogeneous systems such as mass and solution polymerizations. Under these conditions, the heat transfer, associated with the appreciable exothermicity of the reaction, becomes a difficult factor to control. From a kinetic point of view, the viscosity changes cause the initiation, propagation and, especially, the termination steps of the chain reaction(s) to become diffusion-controlled. These phenomena are defined in terms of cage, glass, and gel effects; they represent the most difficult part of a polymerization process to model.

Another problem related to polymerization reactions is the possibility of multiple solutions. Differences in operating conditions (such as feed concentrations, temperature, pressure, or catalyst addition rates) can lead to the same type of polymer (same molecular mass, density, and composition), but with different productivity.

Ensuring the quality of product is a much more difficult task in polymerization processes than in conventional reactions involving small molecules, because the morphological and molecular properties of a polymeric product strongly influence its physical, chemical, thermal, rheological, and mechanical properties, as well as the final applications of the polymer. From this point of view, the development of mathematical models that can predict the quality of the polymer from a knowledge of the operating conditions in a polymerization reactor is usually the key to obtaining an efficient and high quality production and improving the operation of the plant.[1] Many researchers have conducted detailed studies on the mechanistic modeling of polymerization reactors. However, a complete mechanistic model would contain a very large number of differential equations, the solution of which would require considerable time. In addition, the values of many kinetic parameters that appear in the model may be unknown or difficult to determine precisely. Furthermore, development of the model, especially for diffusion-controlled reactions, under discontinuous or semi-discontinuous operating conditions, is extremely difficult when several polymers of different product specifications are manufactured in the same installation.

Developing a kinetic model for free radical polymerization is not an easy task, especially when polymerization takes place at high temperatures and pressures. These difficulties are not only due to the complex reactions that occur simultaneously in the reactor but also to the limited understanding of the physical and chemical phenomena related to the mixtures including polymers.

An alternative to mechanistic modeling is provided by empirical modeling, which requires less knowledge specific to the process than is needed for phenomenological modeling. Empirical modeling requires data (measurements) on the variables considered to be representative for the behavior of the process and for the quality or properties of the product or system. Statistical regression techniques and neural networks have been increasingly used to build empirical models.

10.2 Phenomenological Modeling of the Free-radical Polymerization Process

This section is necessary before approaching the application of ML techniques in free radical polymerization for at least two reasons. First, the difficulties related to establishing and solving the mathematical model of the process, as well as problems related to the accuracy of the modeling, are demonstrated here. Second, the mechanistic models inserted here may constitute data generators required for techniques based on ML.

Among the polymerizations that take place through free radicals, the polymerization of methyl methacrylate (MMA) and styrene (St), similar in many aspects, but with specific features, were chosen as examples. We will try to show the usefulness of ML techniques to overcome the difficulties of classical modeling, but also to make a significant contribution to the accuracy of the results. Also, it is not difficult to demonstrate that a user who may not be familiar with these techniques (such as a chemical engineer) can readily manipulate them to obtain the predictions (s)he needs, especially when the software products are provided with a graphical user interface. It should be noted that most of the software products used in the case studies presented here belong to the author of this chapter (and to her group) and some of them being patented.

In a free radical polymerization reaction, a phenomenological model starts from the kinetic scheme of the process; for MMA this scheme is:

$$\text{Initiation} \qquad I \xrightarrow{k_d} 2R^*$$

$$R^* + M \xrightarrow{k_i} P_1{}^*$$

$$\text{Propagation} \quad P_n{}^* + M \xrightarrow{k_p} P_{n+1}{}^*$$

$$\text{Termination by disproportionation:} \quad P_n{}^* + P_m{}^* \xrightarrow{k_t} D_n + D_m$$

where I is the initiator; R^* is the initiating radical; M is the monomer; $P_n{}^*$ is the polymer growing chain of length n; D_n is the dead polymer of length n; k_d, k_i, k_p, and k_t represent the rate constants for initiator decomposition, initiation, propagation, and termination by disproportionation, respectively.

Assuming the steady-state and that no monomer is consumed in the initiation stage, one obtains the following equations based on the above kinetic scheme.[2]

$$\frac{\mathrm{d}I}{\mathrm{d}t} = -k_d I - I\varepsilon \frac{1-x}{1+\varepsilon x} \lambda_0 k_p \tag{10.1}$$

$$\frac{\mathrm{d}x}{\mathrm{d}t} = k_p (1-x) \lambda_0 \tag{10.2}$$

$$\frac{\mathrm{d}\lambda_0}{\mathrm{d}t} = 2fk_d I - k_t \lambda_0^2 - \lambda_0^2 \varepsilon \frac{1-x}{1+\varepsilon x} k_p \tag{10.3}$$

$$\frac{d\lambda_1}{dt} = k_p M_0 \frac{1-x}{1+\varepsilon x}\lambda_0 - k_t\lambda_0\lambda_1 - \lambda_0\lambda_1\varepsilon\frac{1-x}{1+\varepsilon x}k_p \tag{10.4}$$

$$\frac{d\lambda_2}{dt} = k_p M_0 \frac{1-x}{1+\varepsilon x}(2\lambda_1 + \lambda_0) - k_t\lambda_0\lambda_2 - \lambda_2\lambda_0\varepsilon\frac{1-x}{1+\varepsilon x}k_p \tag{10.5}$$

$$\frac{d\mu_0}{dt} = k_t\lambda_0^2 - \mu_0\lambda_0\varepsilon\frac{1-x}{1+\varepsilon x}k_p \tag{10.6}$$

$$\frac{d\mu_1}{dt} = k_t\lambda_0\lambda_1 - \mu_1\lambda_0\varepsilon\frac{1-x}{1+\varepsilon x}k_p \tag{10.7}$$

$$\frac{d\mu_2}{dt} = k_t\lambda_0\lambda_2 - \mu_2\lambda_0\varepsilon\frac{1-x}{1+\varepsilon x}k_p \tag{10.8}$$

where x is the conversion of monomer; λ_k and μ_k are the moments of macroradicals and dead polymer, respectively, that give the molecular weight distribution (k=0, 1, 2); ε takes into account the volume variation during the reaction; M_0 represents the concentration of monomer at time t=0, and f is the efficiency of the initiator. The system composed from eqn (10.1)–(10.8) was obtained with the moment method, detailed in ref. 3.

A major feature of homogeneous free radical polymerization is the significant increase in viscosity of the reaction mass as the reaction progresses (as monomer conversion increases). Under these conditions, a deviation from the normal kinetic behavior and changes of mass and heat transfer in the reaction medium are observed.

The highest sensitivity to the increase of viscosity is presented by the rate constant of the termination step, k_t. The severe reduction of the mobility of the macroradicals due to the increase of viscosity causes a decrease in the value of k_t, resulting in a self-acceleration of the reaction (the *gel effect* or Trommsdorff effect).

Normally, when polymerization is conducted at low temperatures, at a certain level of conversion, a transition state is reached, and the temperature corresponding to this moment is called the glass transition temperature. The rate constant k_p decreases before the monomer is completely consumed. This is the *glass effect* in radical polymerization, being also the effect of the viscosity increases.

Finally, at high conversions, due to the increase in viscosity, the efficiency of the initiator also decreases, appearing as the so-called *cage effect*.

Therefore, one of the specific problems of free radical polymerization is that it is diffusion-controlled, which leads to the manifestation of gel, glass, and cage effects. Process modeling is directly affected by these because, as Figure 10.1 shows, the above mathematical model formulated in terms of moments (master curve) fails to reproduce the experimental data.

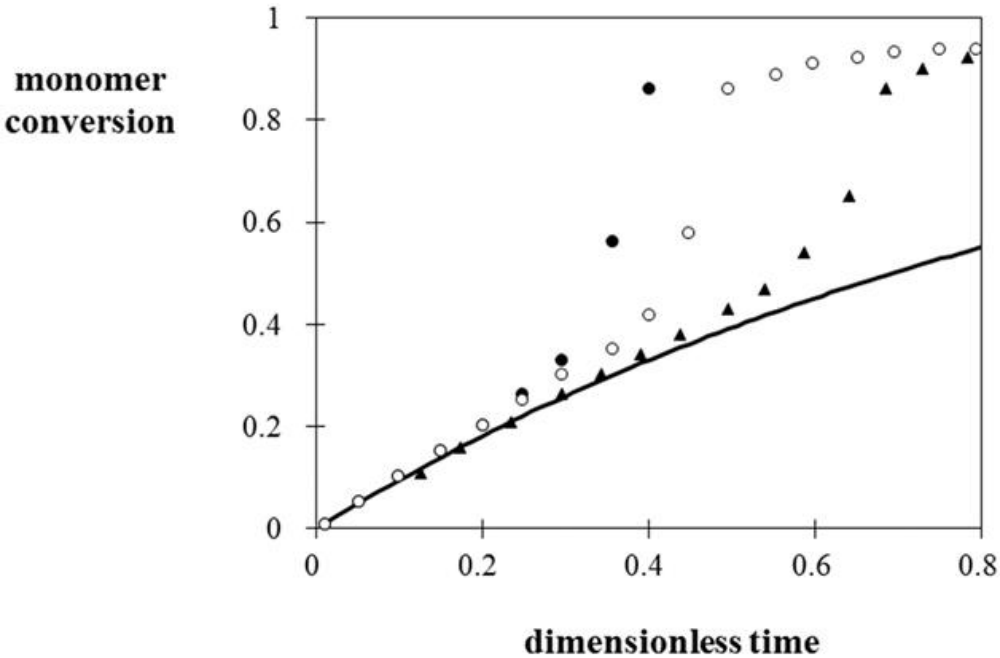


Figure 10.1 Master curve and experimental data obtained at 50, 70, and 90 °C for MMA conversion.

From a certain value of conversion, called the critical conversion (corresponding to the deviation of the experimental data from the kinetic curve of Figure 10.1), significant structural changes occur in the reaction mass due to the transition to diffusion control. To obtain a realtsic model we require a means of capturing these changes in a mathematical formalism.[4]

The most widely used model that considers the variations of propagation and termination rate constants (gel and glass effects) was proposed by Chiu *et al.*[5]

$$\frac{1}{k_t} = \frac{1}{k_{t0}} + \theta_t(T, I_0)\frac{\lambda_0}{\exp\left[\dfrac{2.303\Phi_m}{A(T) + B\Phi_m}\right]} \tag{10.9}$$

$$\frac{1}{k_p} = \frac{1}{k_{p0}} + \theta_p(T)\frac{\lambda_0}{\exp\left[\dfrac{2.303\Phi_m}{A(T) + B\Phi_m}\right]} \tag{10.10}$$

$$\Phi_m = \frac{1 - x}{1 + \varepsilon x} \tag{10.11}$$

$$\theta_t = \frac{\theta_t^0}{I_0}\exp[E_{\theta t}/(RT)] \tag{10.12}$$

$$\theta_p = \theta_p^0 \exp[E_{\theta p}/(RT)] \tag{10.13}$$

$$A(T) = C_1 - C_2(T - T_{gp})^2 \tag{10.14}$$

where k_{t0} and k_{p0} are the rate constants for termination and propagation in the absence of gel and glass effects; A, B, C_1, and C_2 represent terms in diffusion equations for propagation and termination rate constants; I_0 is the concentration of the initiator at time t=0; θ_t and θ_p are characteristic migration times; θ^0_t, θ^0_p are pre-exponential factors for θ_t and θ_p; ϕ_m is the volume fraction of the monomer; $E_{\theta t}$ and $E_{\theta p}$ are activation energies for θ_t and θ_p; T is the temperature in Kelvin; T_{gp} is the glass transition temperature of the polymer, and R is the universal gas constant.

Another example of gel and glass effects models was proposed by Curteanu *et al.*[6] These models are represented by simple dependencies between the termination and propagation rate constants and the monomer conversion, thus quantifying their variation as the reaction progresses:

$$k_t = k_{t0} \exp(A_t + B_t x + C_t x^2 + D_t x^3) \tag{10.15}$$

$$k_p = k_{p0} \exp(A_p + B_p x + C_p x^2 + D_p x^3) \tag{10.16}$$

in which k_{t0} and k_{p0} represent the rate constants of termination and propagation in the absence of gel and glass effects, and A, B, C, and D, with t and p indices for termination and propagation, respectively, are dependent on time (conversion) and reaction conditions (temperature and reactant concentrations); these are determined from experimental data. Thus, compared with Figure 10.1, much better results are obtained by adding the model that includes diffusion-controlled effects (see Figure 10.2).

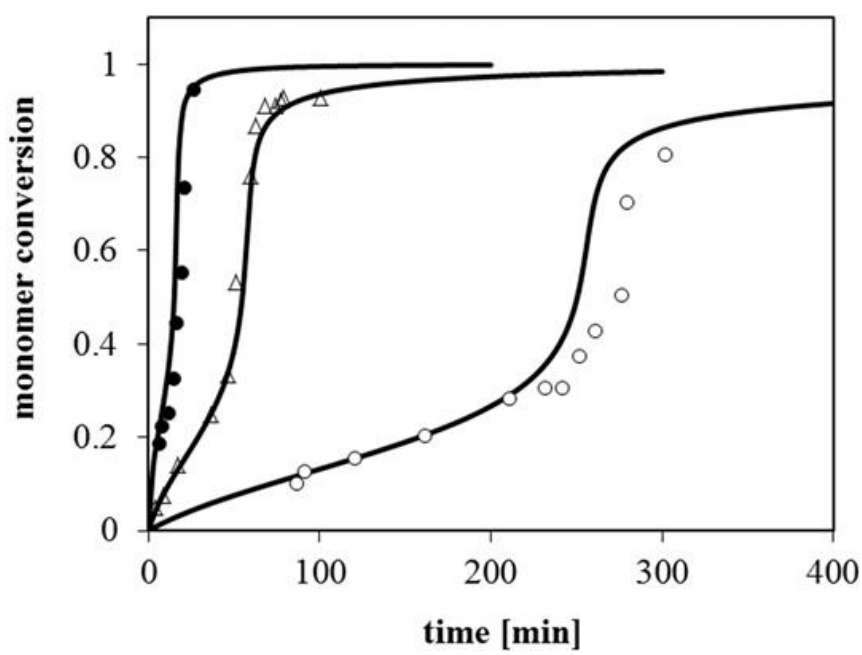


Figure 10.2 Variation of monomer conversion with time at different temperatures for MMA polymerization.

The main criteria in evaluating a mathematical model are accuracy of results and ease of handling. The ideal case corresponds to the quantification of the phenomenology of the process (physical and chemical laws that govern the process) through mathematical relations. However, in many cases in models of diffusion-controlled effects, empirical constants are needed to match the experimental data or these models are entirely of the black box type. Thus, the use of artificial neural networks for free radical polymerization can be considered as a favorable alternative; the advantages it presents overcome the so-called deficiency of their empirical character. A third possibility is to combine two types of models, phenomenological and empirical, as will be described later in this chapter.

10.3 Artificial Neural Networks Applied to Polymerization Processes

Given the difficulties of modeling the polymerization processes mentioned in section 1, Artificial Neural Networks (ANN) prove to be useful predictive models in a wide range of applications in the field of polymer engineering. They possess the ability to learn what happens in a process without quantifying the physical and chemical laws of the system. Therefore, they are recommended modeling tools in the case of complex non-linear processes for which phenomenological understanding is limited.

The manipulation of the ANNs applied to the modeling of polymerization processes implies collecting training data from experiments or by simulation on mechanistic models, setting up the training and validation data sets according to the type of neural network, determining the architecture and parameters of the network (training) and, finally, evaluating the generalization capacity of the neural model when the network produces results for data that are not included in the training set. This last step really establishes the quality of the model, namely its generalization ability.

The input variables in an ANN are chosen depending on the process, either based on the experience and knowledge of the engineer, or by applying a sensitivity analysis that allows the parameters to be ranked according to their importance. The outputs correspond to the purpose pursued, being correlated with the input variables.

The quality of a neural model depends on the quantity of data available for modeling, on their representativeness, on the uniformity of their distribution in the investigated space, and on the methods used to develop the neural model. This last consideration refers to the choice of the best type of model depending on the process subject to modeling and the training method with the highest chance of leading to an optimal or near-optimal model. Regarding the quantity of data, a significant number of data samples is recommended, more important than their number being a relatively uniform coverage of the domain to which they belong. A larger amount of data and, consequently, a longer training time, is required in the case of complex, strongly non-linear systems. Obviously, the accuracy of the experimental data is a key factor in determining the accuracy of the modeling results.

Developing the topology of a neural network is a critical and sometimes difficult step, especially when it comes to determining the optimal network architecture. In addition to trial-

and-error, different techniques have been tested, of which evolutionary algorithms have proven to be the most efficient variants.

Concerning the polymerization processes, one can solve problems such as estimating the reaction characteristics (conversion, reaction yield) depending on the working conditions (time, temperature, concentrations of reactants, *etc.*), predicting the final properties depending on the initial compound structures and/or reaction conditions, identifying the optimal working conditions that lead to pre-defined structures or characteristics of the reaction, identifying the structures that will generate pre-established final properties (molecular design), or determining the most influential input parameters.

Numerous articles in the specialized literature cover modeling, optimizing, predicting and controlling some parameters from polymerization reactions using various ML techniques, of which, in particular, individual neural networks or hybrid soft-computing techniques are notable. Noor *et al.*[7] present a number of important issues related to the application of ANNs in polymerization reactions including vinyl acetate polymerization, continuous copolymerization processes, gas-phase olefin polymerization, continuous free-radical polymerization of styrene in a tubular reactor, free-radical polymerization of MMA, semibatch polymerization, and polymerization of nylon-6.6 in a twin-screw extruder reactor. The first part of Curteanu's article[8] is a review that also associates neural networks with polymerization processes. Fernandes and Lona[9] provide a brief tutorial on the use of ANNs for polymerization processes, a tutorial which contains a number of practical considerations regarding neural network selection and training.

10.3.1 Types of Applications Based on Neural Networks in Polymerization Reaction Engineering

When using neural networks, some considerations are the type of network to be employed, how this relates to different types of applications and different processes, determination of the best network architecture, and how they might be included in different soft-computing configurations.

ANNs have proven to be suitable tools for direct and inverse neural network modeling, monitoring (soft sensors), and control procedures. In addition to these applications, two types of approaches can be highlighted: 1) determining how the process performance (efficiency, yield *etc.*) or the properties of the final product (chemical, physical, mechanical, optical, surface properties *etc.*) depend on the reaction conditions; and 2) analysis of quantitative structure property/activity relationships (QSPR/QSAR) in an attempt to correlate structural descriptors of compounds with their physico-chemical properties and biological activities.

In Direct modeling, a correlation is derived between the output values of one or more variables and a series of input variables that significantly influence the outputs. For example, Chan and Nascimento[10] model olefin polymerization in high pressure tubular reactors using feedforward neural networks. The selection of the best network topology was based on a sensitivity analysis relating the mean square error to the number of intermediate neurons.

For a chemical reactor, optimization involves determining the reaction conditions that lead to suitable values for properties that are required in the final product or to a maximum efficiency of the process. Inverse neuronal modeling can respond to the first optimization variant and, even if it is not easy to perform, compared to the classical procedures that require the existence

(determination) of a good mathematical model and the selection of an optimization technique, it is an acceptable alternative.[11] Fernandes and Lona[9] designed a neural network that determines the operating conditions (comprising monomer and comonomer feed rates, catalyst feed rates, surface gas velocity, and porosity) of gas-phase ethylene polymerization in a fluidized bed reactor leading to the desired properties of the copolymer (molecular mass, polydispersity, composition, and production). Another example related to the synthesis of polyacrylamide-based hydrogels by inverse neuronal modeling consists in determining, in an easy manner and with good results, the initial reaction conditions which lead to a pre-established reaction yield and maximum swelling degree.[12]

In many processes in chemical engineering, "difficult-to-measure" variables are estimated using "easy-to-measure" variables, based on a two-way correspondence between them. One solution to such a problem is the use of inferential models or soft sensors represented by neural models. For example, Zhang and co-workers[13] apply inferential software sensors for a batch reactor in which MMA polymerization takes place. The molecular mass of the polymer is estimated based on online measurements of reactor temperature, jacket inlet temperature, jacket outlet temperature, monomer conversion and coolant flow rate. Gonzaga *et al.*[14] describe the implementation of a virtual sensor (soft-sensor) in the polyethylene terephthalate (PET) production process. The sensor, based on an ANN, is used to estimate the viscosity of PET, using both online and redundant measurements. Thus, an efficient strategy for controlling this industrial process is designed, with neural networks overcoming the disadvantages of difficult measurements.

Another type of application is represented by the use of ANNs for the identification of systems and design of controllers, through the inclusion of neural models in optimal control strategies. Wei *et al.*[15] proposed an optimal control strategy for a propylene polymerization process, integrating a simplified mechanistic model based on a polymer property estimator and a feedforward neural network model based on a predictive control model for melt index. Lifgtbody and Irwin[16] present a neural network modeling application designed to improve the control of a polymerization reactor, highlighting the problem of precise control of polymer viscosity based on delayed measurements. A non-linear predictor consisting of a feed-forward neural network is used to remove limitations that arise from delays in measurements.

10.3.2 Different Types of Neural Network Models Used in Modeling of Polymerization Processes

Neural networks can be divided into two broad categories: *static*, which includes feedforward neural networks and *dynamic*, which includes global recurrent, local recurrent, Elman and dynamic filter networks. Static neural networks are suitable for short-term predictions and are relatively easy to develop and train, while dynamic ones are recommended for long-range prediction models.

Feed-forward (multilayer perceptron) neural networks are often used for modeling chemical processes, and their universal approximator qualities give very good results in most cases. One or two hidden layers containing a reasonable number of neurons are usually sufficient. But, in some cases, other types of networks have proven effective in modeling chemical engineering processes, exceeding the performance of feed-forward networks. In this section, this will be

illustrated, contrasting with the examples included in the previous sections which relate mainly to the multilayer perceptron.

A feed-forward network that models styrene polymerization under semi-batch and non-isothermal conditions has been reported by Curteanu and Petrila.[17] This approach demonstrates the usefulness of ANNs in modeling a complex process, under special operating conditions, semi-discontinuous and non-isothermal. The supplementary addition of initiator or monomer at certain stages of the reaction (dependent on critical conversion) contributes to the control of polymer molecular mass and, therefore, its properties. The same effect is obtained in non-isothermal conditions, represented by increasing temperature steps. Simple neural networks were used, of feed-forward type with two intermediate layers, the same models being valid for different operating conditions; differences are achieved by the way the models are used, not by changing their structure. Figure 10.3 shows the dependence of the monomer conversion on intermediate addition of an initiator, and Figure 10.4 shows the molecular mass of the polymer resulting from the application of a non-isothermal program.

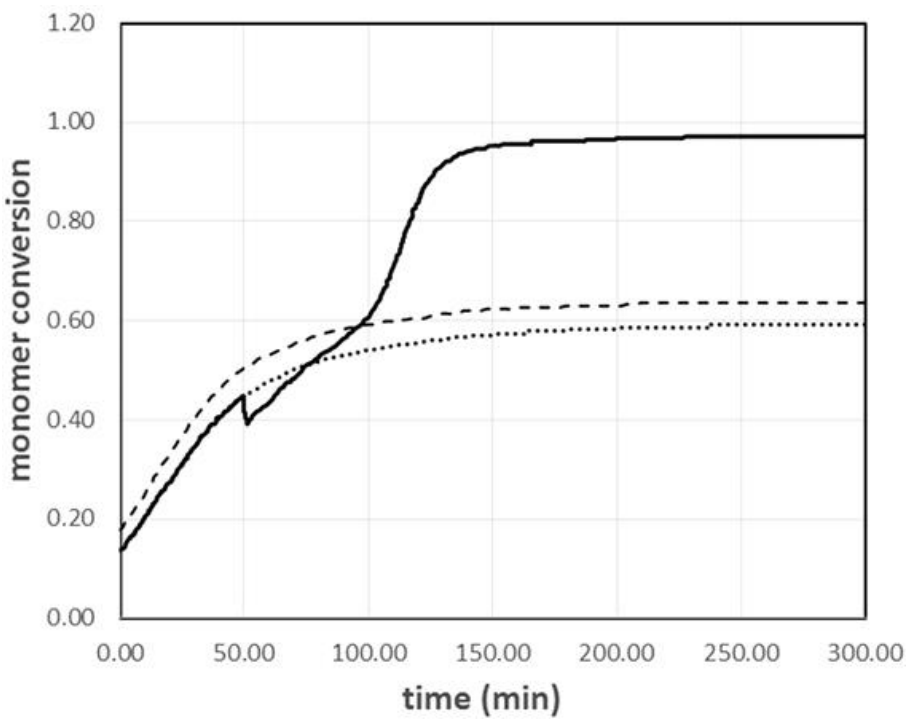


Figure 10.3 Monomer conversion obtained with a feed-forward neural network with intermediate addition of 5 mol m^{-3} of initiator, I_0 =15 mol m^{-3}, before the gel effect (continuous line); dotted curve I_0=15 mol m^{-3} and dashed curve I_0=20 mol m^{-3}.

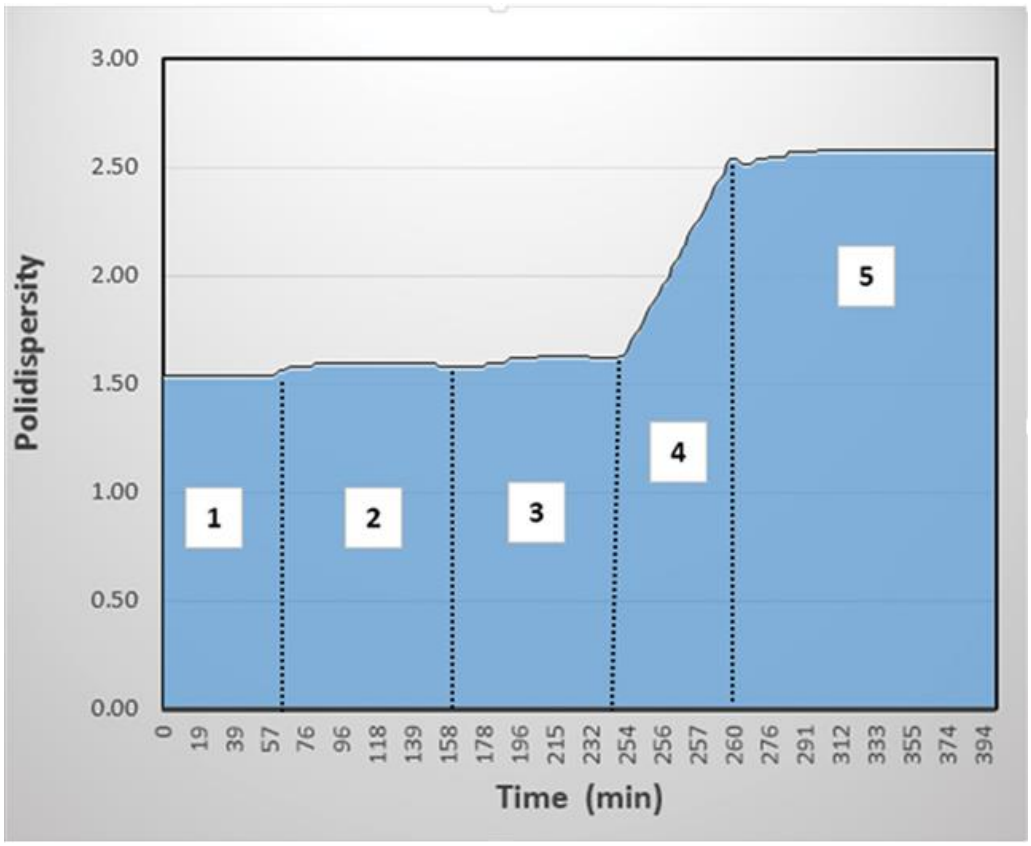


Figure 10.4 Polidispersity index obtained with a feed-forward neural network for five step increasing temperature: 80, 90, 95, 100, 110 °C.

Recurrent neural networks have the advantage that they provide long-term predictions, which are useful for many processes in chemical engineering. For example, in discontinuous processes the aim is to optimize the quality of the final product (this is a frequent requirement of the polymerization processes). Therefore, precise, reliable long-term predictions are essential in such cases.

Tian and co-workers[18] present the development of a recurrent neural network model for an emulsion copolymerization reactor in which the essential parameters are the average numerical molecular mass (M_n), the copolymer composition (C), the polymer conversion (X), and the temperature profile (T). Dynamic modeling assumes that these quantities occur not only at the output, but are added to the inputs by values at previous times, because these previous values influence the final results. For example, a neural network developed for modeling M_n has inputs $T\,(t-1)$, $T\,(t-2)$, $X\,(t-1)$, $X\,(t-2)$, $M_n\,(t-1)$ in which t is time.

Jordan-Elman neural networks are recurrent neural networks that supply MLP with context units, representing processing elements that remember past activity. Context units provide the network with the ability to extract temporal information from the data.[19] An example is the modeling of the synthesis and swelling behavior of polyacrylamide gels with inclusion polymer.[20] To evaluate the quantitative influence of the reaction conditions on the yield and

swelling degree, neural networks of different types were used, of which Jordan-Elman networks proved to be the most efficient.

Curteanu and co-workers[21] used modular neural networks to predict the liquid crystal property of poly (siloxane-azomethine) based on a series of molecular and structural descriptors including fully extended length, diameter of the structural unit, and the dipole moment. This type of neural network uses several MLP networks in parallel, combining their results. There is no total interconnectivity, so the number of weights is smaller, and the structure benefits from some flexibility due to the diversity of ways in which an MLP can be segmented into modules. The applied method provides very good results, thus proving the utility of neural networks in classification problems, in particular for quantifying the structure–property relationship.

Using as a case study the polymerization of styrene carried out in a CSTR, Haiyan *et al.*[22] applied a novel method for modeling and controlling the molecular mass distribution of the polymer. Their technique is based on combining the orthogonal polynomial feedforward neural network (OPFNN) with a recurrent neural network (RNN), resulting in what the authors call a gray-box model. Satisfactory results are obtained when the molecular mass distribution of a polymer is difficult to accurately model with classical tools.

An alternative to the use of individual neural networks is represented by stacks of neural networks, which, in many cases, leads to an increase in the performance of neural models, as it combines different neural networks that capture different aspects of the modeled process. An additional advantage of using different neural networks arises because of the various possibilities of combining the outputs of the individual networks, as well as varying the number of networks included in the stack. The flexibility of the stacking technique represents an advantage of these types of models that give real chances of increasing the accuracy of the results.

Zhang *et al.*[13] used stacks of neural networks to evaluate the properties of a polymer by estimating polymer quality variables from process measurable variables. For each set of data, a neural model is designed which, subsequently using PCA (Principal Component Analysis), is aggregated into a stack. In a further example[23] neural network stacks were used to model the synthesis of multicomponent hydrogels based on polyacrylamide. The swelling degree of the product and reaction yield were correlated with working conditions, which included the quantities of monomer, initiator, cross-linking agent, inclusion polymer, time and temperature. The performance of the stack model (see Figure 10.5) compared favorably with that of individual neural networks.

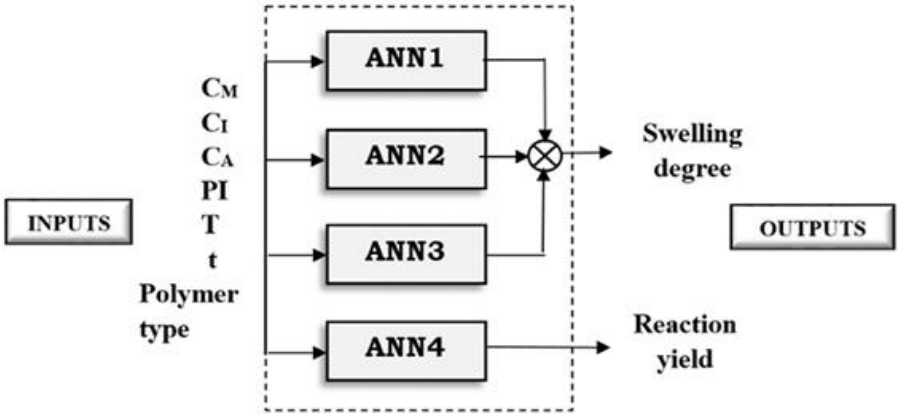


Figure 10.5 Neural network stack applied to modeling of the polyamide–hydrogel synthesis.

When choosing between phenomenological modeling and black box techniques such as ANNs, several factors should be taken into account. Phenomenological modeling has the advantage that it is based on the physical and chemical laws that govern the process, but, in most cases, the approximations they include lead to inaccurate results. On the other hand, empirical modeling, conducted through an appropriate procedure, can provide precise results, and yet does not clarify the process phenomenology. The two classes of methods represent viable modeling alternatives, with applicability depending on the process and the purpose pursued. Following the previous discussion, the utility of hybrid neural models that include a simplified (approximate) phenomenological model and one or more neural networks whose role is to correct the outputs of the phenomenological model or to model difficult parts of the process is evident. Such hybrid modeling offers the possibility of extracting the best features of both approaches, phenomenological and empirical, which leads to precise results.

In the polymerization processes, there are many parts or variables that are difficult to model: the effects of gel, glass and cage, reaction rates, kinetic constants, fluid and flow behavior, physical properties, *etc.* As a result, these processes are good candidates for hybrid neural modeling, especially for on-line advanced model-based control design strategies.[24] Some examples are given below.

Tian and co-workers[1] approach the discontinuous polymerization of MMA with a hybrid model composed of a simplified phenomenological model (that ignores the gel effect) and stacks of recurrent neural networks (that correct the residuals of the mechanistic model). A similar strategy is applied by Nascimento *et al.*[25] to the finishing stage of nylon-6.6 polycondensation in a twin-screw extruder reactor. Within the hybrid model, the information on the final amino and carboxyl groups comes from a mechanistic model; subsequently, these results are used to calculate the relative viscosity with a neural network model. Another example of a hybrid model is designed for an industrial polyethylene process.[26] The mechanistic model part in this example is developed to predict the temperature in the reactor, the conversion and the distribution of the polymer molecular mass. For this latter parameter, important for the evaluation of the end-use properties of the polymer, no satisfactory results are obtained through a classical modeling based on fundamental material and energy balances. However, the empirical component represented by neural networks of feed-forward type, added

to the phenomenological model, makes it possible to obtain precise predictions for the distribution of the polymer molecular mass. Vega *et al.*[24] developed a hybrid-neural model for a loop-tubular polymerization reactor for polystyrene production. An approximate mechanistic model was combined with neural networks that have the role of modeling reaction rates and, in addition, correcting the final errors in modeling.

10.4 Soft-computing Hybrid Configurations Applied to Polymerization Processes

Neural networks can be combined with other ML tools, leading to the development of improved performance methodologies, with possibilities in a variety of different applications: modeling, optimization or automatic control. The most commonly used combinations include ANNs and evolutionary algorithms or neural networks and fuzzy systems. Obviously, the number of soft – computing techniques that can be combined is not limited to two.

For example, evolutionary algorithms can be used to 1) optimize the architecture and parameters of a neural network, 2) estimate the optimal set of parameters at the input of a neural model, or 3) optimize the processes modeled with neural networks.

Regarding the first option, several researchers argue that evolutionary methods are more effective in determining the topology of neural networks,[27,28] than other techniques such as trial and error, empirical or statistical, fuzzy inference, and constructive or destructive methods.

An evolutionary algorithm searches a population of chromosomes whose genes encode the elements of a neural network, looking for the best (adapted) individuals who satisfy an objective function (known as the fitness function). The mechanism that determines the search progress is based on the genetic operators of selection, recombination and mutation. Such a problem is identified with a multi-objective optimization in which the space of the solutions consists of different topologies of the neural networks from which the algorithm will select the best one. The process is iterative, after each step the better solutions being maintained[29] with, in some variants, the solution of highest fitness being protected so that it cannot be lost. The use of neuro-evolutionary techniques applied to the design of neural networks is presented in more detail in ref. 30.

An example is fluorescence modeling[31] of the polydimethylsiloxane/silica composites containing lanthanum, which was performed using a methodology based on ANNs, individual or aggregated in stack, designed with a genetic algorithm (GA). The GA, applied here to determine the topology of neural models, has proven to be an efficient, flexible method, leading to a near optimal configuration with high probability.

Differential Evolution (DE) is another commonly used evolutionary algorithm that can be applied to global search within the space of a typical feed-forward neural network. Curteanu *et al.* applied DE to the design of a neural network that modeled the free radical polymerization of styrene.[32] The good results provided by this algorithm suggest that this method is an efficient one.

Another example,[33] in which a hybrid model consisting of a simplified phenomenological model and a neural network developed using the DE algorithm, addresses the free radical polymerization of styrene. What is noteworthy in this approach is the different ways in which

two sub-models connect. The best variant that leads to accurate predictions for monomer conversion and molecular mass of the polymer is assumed to have been accomplished by executing the following steps: 1) for each combination of variables, the moment when the gel effect appears was automatically determined; 2) until this critical moment, the phenomenological model was applied; 3) after the critical moment, an optimal neural model was designed by applying the DE algorithm.

The optimization of discontinuous polymerization processes raises problems related to the proposed objectives, the decision variables, the mathematical solving method, the model that will be included in the optimization procedure, and the specific aspects of the multi-objective optimization, within which the objective function is formulated – either scalar or vectorial. The difficulties related to the development of the mathematical model were discussed in the previous sections. The disadvantages of traditional optimization methods are well recognized:[34] 1) convergence to the optimal solution depends on the choice of the initial solution, so is not guaranteed; 2) most algorithms tend to block in suboptimal solutions; 3) the efficiency of an algorithm depends on the specificity of the problem addressed.

In addition to these difficulties, polymerization processes generally involve multi-objective optimization, which, in turn, brings new problems. Formulating a scalar objective function that involves the weighted combination of several objectives allows the use of simple algorithms for solving, but the results depend significantly on the values of the weights and there is a risk of losing optimal solutions. The use of an objective function formulated as a vector has the advantage of providing the Pareto set of optimal solutions.

As a result of these advantages, techniques based on evolutionary algorithms are gaining ground. They provide a means to overcome the difficulties outlined above, both because of the principles on which their functioning is based and their flexibility in handling different variants or parameters that improve their performances.

An ANN+GA combination, with a scalar multiobjective function, has been applied to optimize a polysiloxane synthesis process.[35] Within this methodology, the neural model evaluates the dependence of the conversion and molecular mass of the polymer on reaction conditions: temperature, time, amounts of catalyst and co-catalyst. The objective of the optimization was to maximize conversion and to obtain a molecular mass with an imposed value. A simple GA was used which, besides optimum values of the decision variables (reaction conditions), also calculated the weights with which the two objectives were scalarly combined.

A similar problem, but based on a more elaborate variant of machine learning, was applied to determine the reaction conditions (temperature, catalyst concentration, reaction time, and initial composition) that lead to maximizing the conversion and obtaining a desired molecular mass in the synthesis of dimethyl-methylvinylsiloxane copolymers.[36] The variation in time of the main reaction parameters was modeled with a feedforward neural network that subsequently calculated the fitness function for the GA, conceived in the elitist non-dominated sorting GA (NSGA-II) variant. The objective function was in vector form, so that the result of the optimization problem is represented by the non-dominated Pareto front containing optimal values for the decision variables.

Neuro-fuzzy hybrid systems combine the advantages of the two types of techniques, namely the explicit, easy to understand knowledge of fuzzy systems with the implicit knowledge, acquired by a neural network through learning. An example for this type of machine learning method is the online monitoring of the melting index, an important variable that determines the

quality of the product obtained in the process of propylene polymerization.[37] The soft sensor consists of two variants: adaptive fuzzy neural network and adaptive fuzzy neural network associated with support vector regression, added for parameter tuning.

The same problem of melting index prediction of polypropylene[38] uses another association of algorithms: neural networks, particle swarm optimization and simulated annealing. Modeling is performed with the RBF neural network; its parameters are optimized with a modified swarm optimization particle variant and combined with simulated annealing, thus overcoming the disadvantages of the two individual optimization methods.

Leon and co-workers discussed the use of several machine learning techniques in predicting the liquid crystal property of copolymers with mesogenic groups in the main chain, depending on a series of molecular and structural descriptors.[39] A comparison was made between the results obtained using feedforward neural networks and categorization algorithms, such as decision trees, nearest-neighbor and Bayesian induction, implemented in some variants: C4.5 pruned, C4.5 unpruned, random tree, random forest, naive Bayes, nearest-neighbor, k-nearest-neighbor and non-nested generalized exemplars. The best results were provided by neural networks and a k-nearest-neighbor method. The observation that the efficiency of the method is primarily related to the choice of the set of input parameters that determine the liquid crystal property was important.

10.5 Machine Learning Techniques Applied to Methyl Methacrylate Free Radical Polymerization

In this section, we will illustrate the application of different machine learning techniques to MMA modeling and optimization. Previously, the phenomenological modeling was presented based on the kinetic scheme of the reaction and the quantification of diffusion-controlled effects. We also highlighted the difficulties of phenomenological modeling, along with the advantages of methodologies that include artificial intelligence tools. To the examples mentioned so far, a more complete example will be added, involving a process for which different variants of modeling and optimization (different machine learning techniques) are compared and graded, with the well-defined purpose of improving the results.

In a discussion of direct and inverse neural modeling of MMA,[10] direct modeling follows the correlation between the monomer conversion, the molecular masses of the polymer and the viscosity of the reaction mass with time, monomer concentration and temperature. Inverse modeling, as an optimization variant, provides the reaction conditions that lead to the predetermined values for conversion and molecular masses. As ML techniques, feed-forward neural networks are used, with one or two hidden layers, determined by trial-and-error. The networks have one or more output variables, being designed as separate models for conversion, molecular mass or viscosity, or a single model with three outputs, given that input variables that have a significant influence on all three outputs can be chosen.

An interesting example is shown in Figure 10.6 in which the monomer conversion obtained with the MLP neural model (3 : 9 : 3 : 1), the phenomenological model, and the experimental validation data are compared. The neural network better captures the behavior of the system.

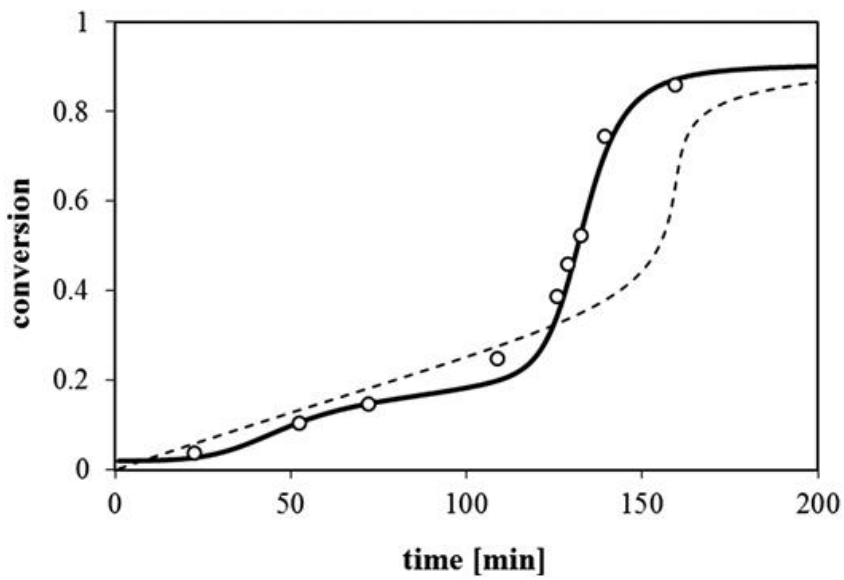


Figure 10.6 Conversion of the monomer in the MMA polymerization obtained with MLP neural model (3 : 9 : 3 : 1) (continuous line), phenomenological model (dotted line), and experimental validation data (○).

Table 10.1 presents values of temperature, T, and initiator concentration, I_0, leading to predetermined values of conversion, x, and numerical average polymerization degree, DP_n, the results being obtained by two optimization methods: sequential quadratic programming (SQP) and by inverse neural modeling (INN). The results of these procedures have as reference values T and I_0 provided by the complete kinetic model. It should be noted that INN values are very close to the reference values, the method thus proving superior to the classical optimization variant.

Table 10.1 Optimization results provided by SQP and INN.

x	DP_n	T_kinetic model(°C)	I_0_kinetic model(mol m^{-3})	T_net (INN) (°C)	I_0_net (INN) (mol m^{-3})	T_SQP(°C)	I_0_SQP(mol m^{-3})
0.9354	3761.9	50	40	50.4061	39.8552	55.1	19.4
0.9354	3761.9	50	40	50.4061	39.8552	50	40
0.9549	2092.9	60	30	59.3373	31.2988	64	18.1
0.9549	2092.9	60	30	59.3373	31.2988	60	30
0.9665	708.74	70	50	70.3659	49.4917	89.9	11.6
0.9665	708.74	70	50	70.3659	49.4917	70	50
0.9973	95.607	90	100	89.4078	95.9585	90	100

A hybrid model[40] has proved to be very efficient for MMA modeling, the improvement being especially evident in the distribution of molecular masses. In this approach, a simplified phenomenological model that does not include quantification of the diffusion-controlled effects was combined with neural networks in different ways: 1) The errors resulting from applying the phenomenological model, with higher values as conversion increases, have been corrected with

a feed-forward neural network; 2) The ANN was used as a model for the gel and glass effects, providing the mechanistic model with adequate (correct) values for the rate constants of propagation, termination and transfer to the monomer, these being influenced by diffusion; 3) The two models were used alternately, first the phenomenological model, then the neural network, starting from the so-called critical conversion – the point at which the effects of diffusion become significant.

For the first method, Figures 10.7 and 10.8 show how the residuals of the approximate phenomenological model were corrected for gravimetric polymerization degree (one of the variables that is difficult to model) with a neural network of feedforward type in the training and validation stages.

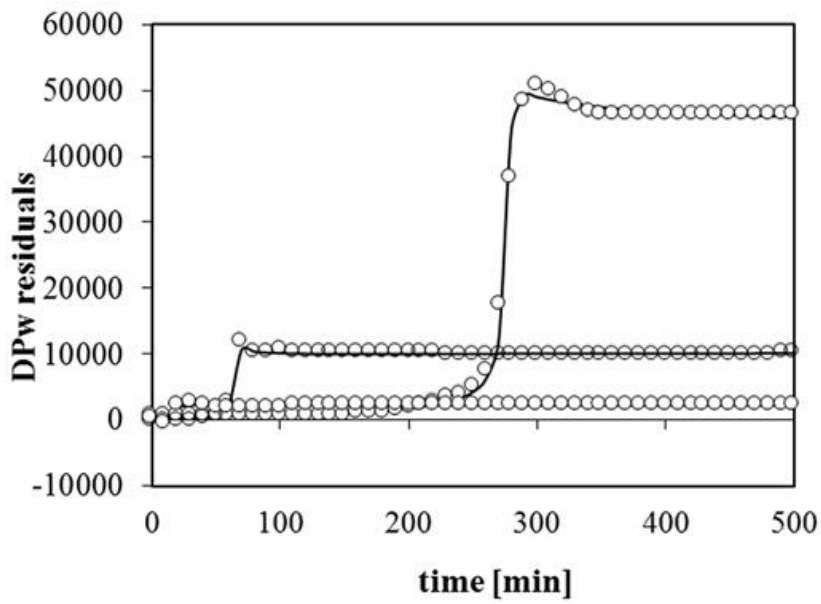


Figure 10.7 Training results of a feed-forward neural network obtained at different temperatures and initiator concentrations for the residuals of gravimetric average polymerization degree; continuous line – training data; ○ – neural model.

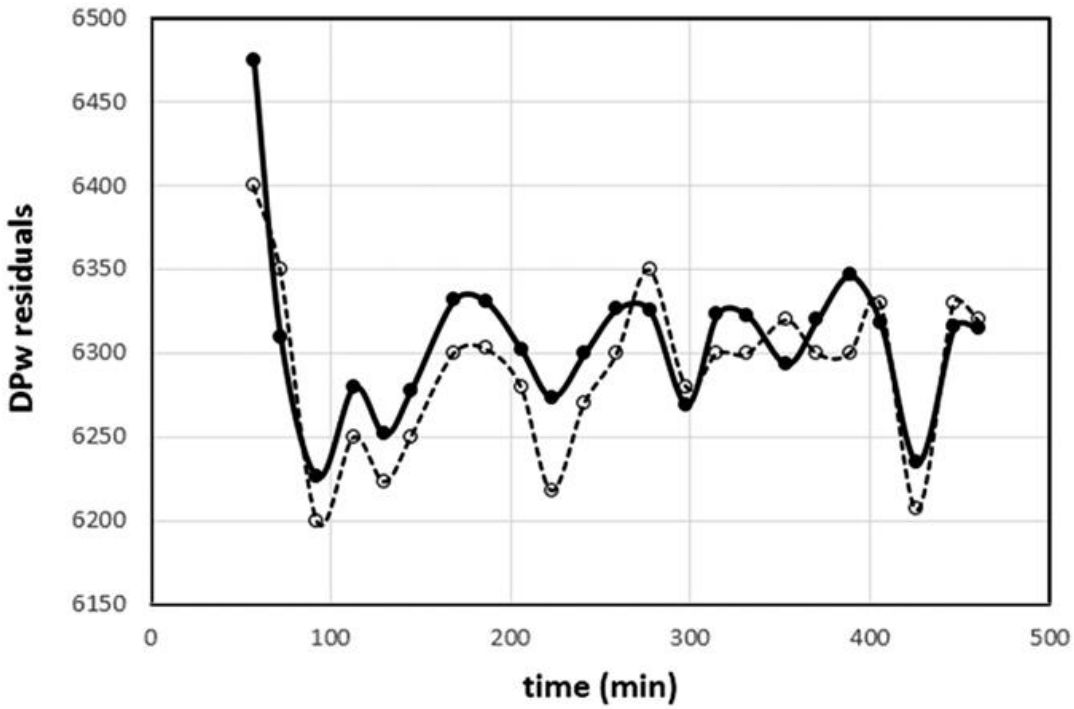


Figure 10.8 Validation results of a feed-forward neural network for the residuals of gravimetric average polymerization degree; continuous and dotted lines – simulation data; ○ – neural network predictions.

For the third case, Figure 10.9 illustrates the successive use of the two models, highlighting the very good match with the reference data.

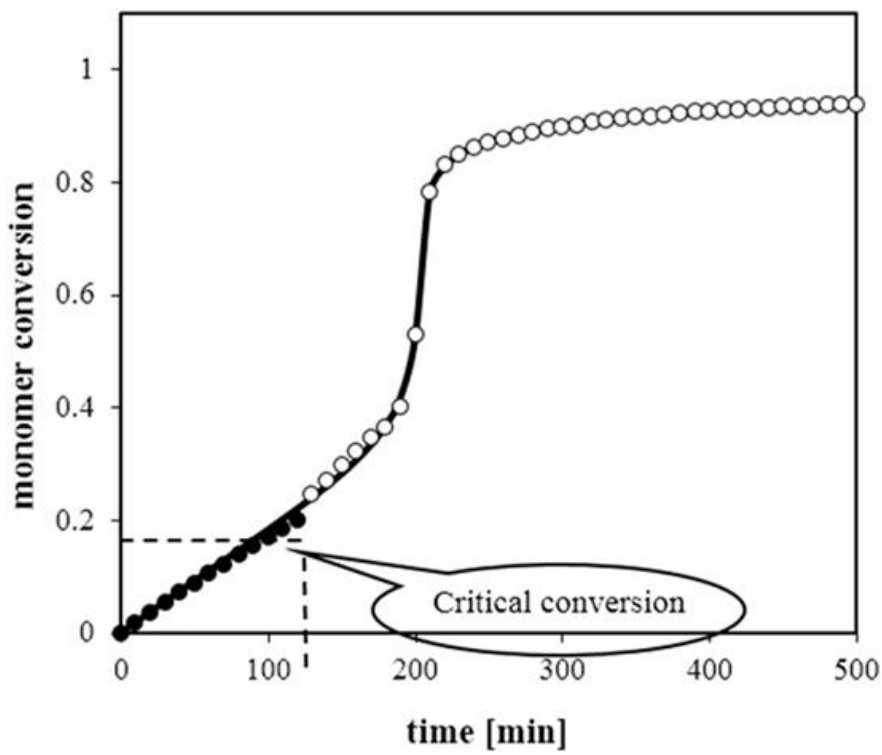


Figure 10.9 Monomer conversion obtained in MMA polymerization with a hybrid model composed from a simplified phenomenological model (●) and neural network (○); continuous line – complete phenomenological model.

Applying one of the three mentioned variants, respectively using a hybrid model, the results obtained in MMA modeling are much better than when individual or even stacked neural networks are used.

Optimization of the MMA polymerization process can be illustrated by some examples in which phenomenological or neural models, traditional optimization or evolutionary algorithms have been used, but what is more interesting in relation to the improvement of the optimization procedure's performance is how different methods can be combined.

A first example[41] starts from the use of a complete phenomenological model for discontinuous polymerization of MMA (presented in section 2), a scalar formulated multi-objective function that includes minimizing reaction time (t), minimizing the polydispersity index (Q), and realizing imposed values for conversion (x) and polymerization degree (DP_n). The decision variables were the initial concentration of the initiator (I_0) and the temperature (T), represented by several isothermal steps. For optimization, three methods were applied, and their results were compared: a classical method based on sequential quadratic programming (SQP), a GA, and a hybrid technique (HM) that combines SQP and GA: GA provides initial values for SQP. The comparison showed the validity of all three methods, because, in a multi-objective optimization, the result is a compromise between the partial objectives; thus, depending on the situation and the priority objective, different results can be chosen. Taking into account aspects such as ease of handling, the accuracy of results, as well as the timing of the procedure, the hybrid method seems to be the recommended one.

Here is an example: imposing the conversion x=0.96 and the numerical average polymerization degree DP_{nd}=1800, the three methods provide the results in Table 10.2.

Table 10.2 Results of three optimization techniques applied to MMA polymerization.

Method	I_0 (mol m^{-3})	T (°C)	x	t (min)	Q	DP_n
SQP	31.1	66.1	0.47	65.8	3.65	1840
		80	0.74	68.7		
		66	**0.94**	123.5		
GA	10	86.7	0.89	33.9	3.67	1797
		43.3	**0.99**	400		
HM	26.6	70.1	0.48	52.3	2.37	1799
		90	0.85	56.1		
		48.2	**0.88**	88.9		

Another optimization strategy applied to MMA polymerization[40] uses neural networks and GAs. The process is modeled with a feedforward neural network developed and trained with a GA to ensure a near optimal topology. Subsequently, the method of solving consists of applying a GA that, besides the optimum values for decision variables, calculates the weights of the objective function, which overcomes the difficulty of a scalar objective function. The aim of the optimization was to minimize the polydispersity index, to maximize monomer conversion and obtain a preset value for the numerical average polymerization degree. The broader aim of the approach was to develop a general methodology for modeling and optimization, based on machine learning techniques, which could be easily adapted to other complex, non-linear processes. Figure 10.10 shows some details of this methodology.

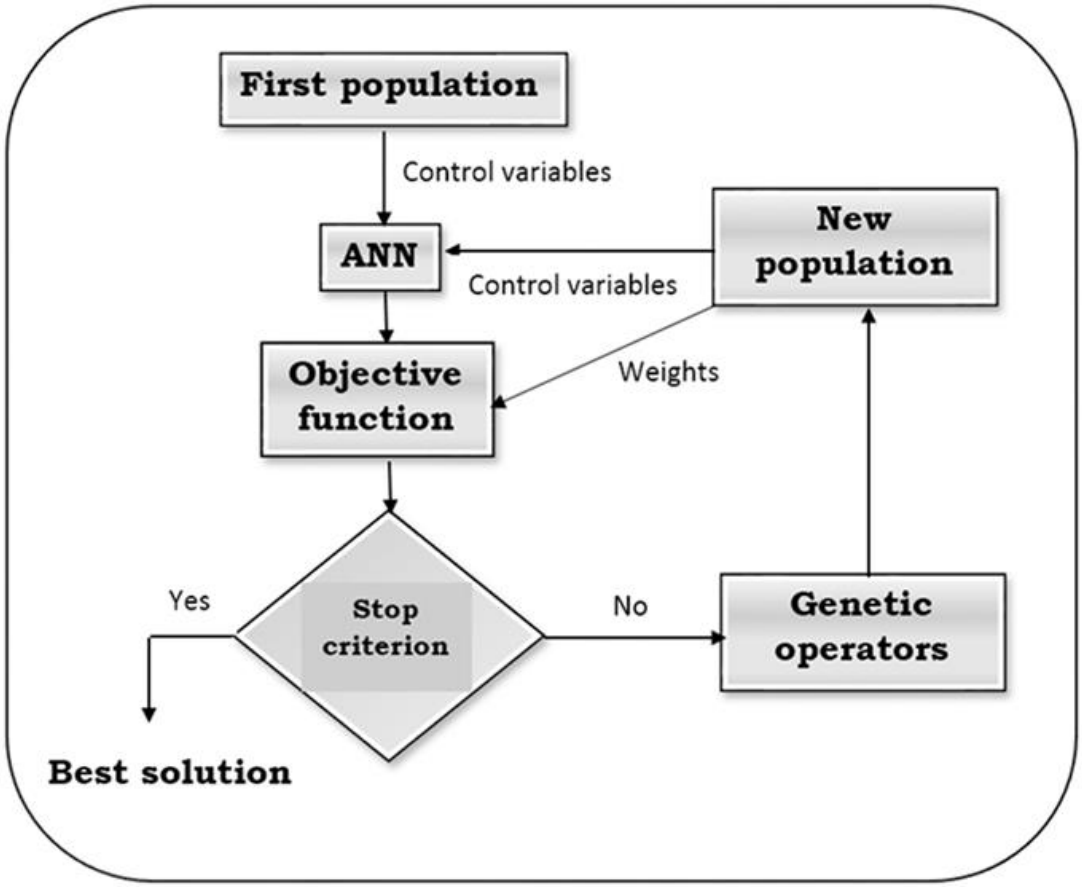


Figure 10.10 Optimization procedure based on a neural model and genetic algorithm.

The predictions provided by neural networks are typically of great practical utility, being able to supplement or replace experiments that may be costly in time, material, and energy. In view of the way that these models work, the prediction time, and the accuracy of the results, they are recommended for online optimal control procedures. A further advantage is presented by the various possibilities offered by neural network types, structure, or by the possibility of combining with other instruments to form efficient hybrid configurations.

References

1. Y. Tian, J. Zhang and J. Morris, *Ind. Eng. Chem. Res.*, 2001, **40**, 4525.
2. S. Curteanu and V. Bulacovschi, *Modelarea si simularea reactiilor controlate de difuzie in polimerizarea radicalica*, Matrix Rom, Bucuresti, 2000.
3. S. Curteanu and V. Bulacovschi, *Mater. Plast.*, 2001, **38**(3), 168.
4. S. Curteanu, V. Bulacovschi and M. Constantinescu, *Hung. J. Ind. Chem*, 1999, **27**, 287.
5. W. Y. Chiu, G. M. Carrat and D. S. Soong, *Macromolecules*, 1983, **16**, 348.
6. S. Curteanu, V. Bulacovschi and C. Lisa, *Polym.-Plast. Technol. Eng.*, 1999, **38**(5), 1121.

7. R. A. M. Noor, Z. Ahmad, M. M. Don and M. H. Uzir, *Can. J. Chem. Eng.*, 2010, **88**, 1065.
8. S. Curteanu, *Cent. Eur. J. Chem.*, 2004, **2**(1), 113.
9. F. A. N. Fernandes and L. M. F. Lona, *Braz. J. Chem. Eng.*, 2005, **22**, 323.
10. W. M. Chan and C. A. O. Nascimento, *J. Polym. Sci*, 1994, **53**, 1277.
11. S. Curteanu and H. M. Cartwright, *J. Chemom.*, 2011, **25**(10), 527.
12. S. Curteanu, A. Dumitrescu, C. Mihăilescu and B. Simionescu, *Polym.-Plast. Technol. Eng.*, 2008, **47**, 1061.
13. J. Zhang, E. B. Martin, A. J. Morris and C. Kiparissides, *Comput. Chem. Eng.*, 1997, **21**, s1025.
14. J. C. B. Gonzaga, L. A. C. Meleiro and C. R. Kiang, *Comput. Chem. Eng.*, 2009, **33**(1), 43.
15. J. Wei, Y. Xu and J. Zhang, *Proceedings of the 2002 IEEE International Conference on Control Applications*, Glasgow, Scotland, U.K., 2002, p. 397.
16. E. Fiesler and R. Beale, *Handbook of Neural Computation*, IOP Publishing Ltd. and Oxford University Press, 1997.
17. S. Curteanu and C. Petrila, *Int. J. Quantum Chem.*, 2006, **106**, 1445.
18. Y. Tian, J. Zhang and J. Morris, *Chem. Eng. Process.*, 2002, **41**, 531.
19. J. Principe, N. Euliano and C. Lefevre, *Neural and Adaptive Systems: Fundamentals Through Simulations*, Wiley & Sons, 2000.
20. S. Curteanu, A. Dumitrescu, C. Mihailescu and B. Simionescu, *J. Macromol. Sci., Part A*, 2009, **A46**, 368.
21. S. Curteanu, C. Racles and V. Cozan, *J. Optoelectron. Adv. Mater.*, 2008, **109120**, 3382.
22. W. Haiyan, C. Liulin and J. Wang, *J. Process Control*, 2012, **22**(9), 1624.
23. S. Curteanu, F. Leon and C. G. Piuleac, *Macromol. React. Eng.*, 2010, **4**(9–10), 591.
24. M. P. Vega, E. L. Lima and J. C. Pinto, *Braz. J. Chem. Eng.*, 2000, **17**(04–07), 471.
25. C. A. O. Nascimento, R. Giudici and N. Scherbakoff, *J. Appl. Polym. Sci.*, 1999, **72**, 905.
26. M. Hinchliffe, G. Montague, M. Willis and A. Burke, *AIChE J.*, 2003, **49**, 3127.
27. M. Dam and N. D. Saraf, *Comput. Chem. Eng.*, 2006, **30**, 722–729.
28. P. G. Benardos and G. C. Vosniakov, *Robotics Comput. Int. Manufact.*, 2002, **18**, 343.
29. P. G. Benardos and G. C. Vosniakov, *Eng. Appl. Artif. Intell*, 2007, **20**, 365.
30. H. M. Cartwright and S. Curteanu, *Ind. Eng. Chem. Res.*, 2013, **52**(36), 12673.
31. S. Curteanu, A. Nistor, N. Curteanu, A. Airinei and M. Cazacu, *J. Appl. Polym. Sci.*, 2010, **117**, 3160.
32. S. Curteanu, F. Leon, R. Furtuna, E. N. Dragoi and N. Curteanu, in *IEEE World Congress on Computational Intelligence*, International Joint Conference of Neural Networks, 18–23 July, 2010, Barcelona, Spain, p. 1293. IEEE Cat. Number: CFP10IJS – DVD, ISBN: 978-1-4244-6917-8.
33. L. Ghiba, E. N. Dragoi and S. Curteanu, *J. Applied Polym. Sci.*, in press.
34. K. Deb, *Optimization for Engineering Design: Algorithms and Examples*, Prentice Hall, New Delhi, India, 1995.
35. S. Curteanu and M. Cazacu, *Rev. Roum. Chim.*, 2008, **33**(2), 1141.
36. R. Furtuna, S. Curteanu and F. Leon, *Buletinul Universitatii Petrol-Gaze, Ploiesti*, 2009, **3**, 161.
37. M. Zhang and X. Liu, *Chemom. Intell. Lab. Syst.*, 2013, **126**, 83.
38. J. Li and X. Liu, *Neurocomputing*, 2011, **74950**, 735.

39. F. Leon, S. Curteanu and N. Hurduc, *Mol. Cryst. Liq. Cryst.*, 2007, **469**, 1.
40. S. Curteanu and F. Leon, *Polym.-Plast. Technol. Eng.*, 2006, **45**, 1013.
41. S. Curteanu, F. Leon and D. Galea, *J. Appl. Polym. Sci.*, 2006, **100**, 3680.

CHAPTER 11

Machine Learning and Scoring Functions (SFs) for Molecular Drug Discovery: Prediction and Characterisation of Druggable Drugs and Targets

I. L. HUDSON,*[a] S. Y. LEEMAQZ[b] AND A. D. ABELL[c]

[a] Mathematical Sciences, College of Science, Engineering and Health, Royal Melbourne Institute of Technology (RMIT), Melbourne, Victoria, Australia
[b] Robinson Research Institute, Adelaide Medical School, University of Adelaide, Adelaide, South Australia
[c] Department of Chemistry, Adelaide Node Director Centre for Nanoscale BioPhotonics (CNBP), University of Adelaide, Adelaide, South Australia
*Email: irene.hudson@rmit.edu.au

11.1 Introduction

The chemical structure of a drug molecule affects its physicochemical properties and biological activities. Lipinski's[1–4] Rule of Five (Ro5) revolutionised the way scientists grappled with exploring the molecular landscape. The introduction of the Ro5 sparked a lot of debate within the scientific community as to the best way to divide the chemical space using molecular parameters (MPs) and other factors. To date, many different rules, filters, and models have emerged that attempt to partition molecules and targets based on a variety of different parameters and factors, with some proving more effective than others. The Ro5 is a filter based on four simple molecular properties (MPs): lipophilicity (a molecule's ability to dissolve in fats or lipids) measured as the octanol/water partition coefficient (logP)≤5, molecular weight (MW)≤500 Da, number of hydrogen bond donors (HBD)≤5, and number of hydrogen bond acceptors (HBA)≤10. These cut-off points were founded on the results from Pfizer's high-throughput screening, which indicated that molecules with lower MW that were more hydrophilic (larger polarity) were more likely to be successful at progressing to the next stage of drug development.[5] Lipinski advanced these results by indicating that molecules that violated two or more of Ro5's rules were much less likely to have the properties for successful development as an orally bioavailable medicine.[6,7] This has had a large impact on the direction and nature of exploration methods used in the pharmaceutical industry, even though the Ro5 filter can only be applied to drugs that use passively transported absorption methods.[5] Despite this, the Ro5 classification is still a commonly used measure of a molecule's druggable potential.

Numerous pharmacologically active molecules, however, have failed to reach the market or have a problematic route of administration due in part to their inherent chemical structure. Some light was recently shown on the reasons surrounding this deficit with chemists now focussing on new drugs with improved potency.[8] This was further confirmed by Rafferty[9] given his discussion of the discrepancy between drug exploratory techniques taught to those in industry and academia. Drug candidates that exhibited poor physico-chemical properties were considered more likely to be lost to attrition due to poor AMDET (absorption, distribution, metabolism, excretion, and toxicity) properties and pharmacokinetics. It was thought that by reducing the number of potential drug candidates that are lost to attrition during pre-clinical development, more time and resources could be invested in developing candidates that progress to the later stages of the discovery process.

Further to Lipinski's work, recent studies have explored the physico-chemical parameters of molecules that are most important to the druggable classification. A number of researchers have reported that lipophilicity is an

important MP as it contributes to a number of other parameters such as solubility, permeability, promiscuity, and selectivity of a compound.[10] Its measurement has also sparked debate as to which calculation (distribution coefficient (logD) or Lipinski's originally proposed logP) most accurately captures its molecular value.[5,11,12] In a study,[13] it was reported that, although logP is a useful descriptor of lipophilicity, it does not account for the ionic variation that molecules possess when exposed to different biological pH values. Many of the drugs currently available on the market possess ionic qualities,[13] which supports the use of logD. However, to date, there remain studies that have used logP as a measure for lipophilicity. Further investigation of the more appropriate measure of lipophilicity and its effect on accurately determining drug-like classification was recently reported.[11,12,14–16] These studies showed that logP values switch magnitude in relation to MW across the druggable status of a molecule. To account for this change, that exists due to the interaction between pH values and ionic levels, logD was proposed as a more accurate biological measure of lipophilicity.[11,12]

Efforts by chemists and data scientists to reliably expand areas in the chemical space that are likely to yield viable drugs have been constant and growing. Indeed, recently, there has been a significant momentum to develop drug candidates beyond the so-called drug-likeness space, outside the Ro5 space,[3] to the so-called beyond Ro5 (bRo5) space, or "middle space",[17] and the expanded Ro5 (eRo5)[17] to tackle difficult targets – with particular focus on up-to-the-minute therapeutic candidates and targets.[8,18] Recently many disease targets have been classified as "undruggable" due to their lack of oral bioavailability.[19–21] Generally, such targets have binding sites, which are large, highly lipophilic, flexible, highly polar and featureless. An ongoing major issue in drug discovery is to select compounds for difficult targets that have higher MW and are more lipophilic.[22] It has been suggested that significant expansions in the range of so-called druggable chemo-space for MW, polar surface area (PSA), logP, HBD and HBA, can provide oral clinical candidates for difficult targets.[17] Recent analyses of candidate molecules have shown that the bRo5 space allows for increased likelihood of discovering orally active and cell permeable compounds and has highlighted their ability to modulate difficult targets.[23] Definitions of the molecular cutpoints (or cut-offs) per MP used in the vector describing "druggability" for such expansions of chemo-space are given in Table 11.1.

Table 11.1 List of main ADMET-based rules of druggability. Table adapted from Mignani *et al.* [48].[a]

Main rules (and authors)	Related parameters with MP cutpoints
Ghose *et al.* rules[71]	MW 160–480, logP −0.4–5.6, ROT 20–70, molar refractivity range 40–130
Lipinski's Ro5[6]	MW (Da)≤500, clogP≤5, HBA≤10, HBD≤5
Oprea & Gottfries[52]	0≤HBD≤2, 2≤HBA≤9, 2≤ROT≤8, 1≤NATOM≤4
Extension of Ro5 (Veber *et al.*[49])	Lipinski cut offs +PSA(A^0)≤140, ROT≤10
Ro4 (Congreve *et al.*[72])	MW(Da)≤400, clogP≤4, HBA≤8, HBD≤4, PSA≤120
Ro3 (Congreve *et al.*[72])	MW(Da)≤300, clogP≤3, HBA≤6, HBD≤3, PSA≤60
Ro2 (Goldberg *et al.*[73])	MW(Da)≤200, clogP ≤2, HBA≤4, HBD≤2
Golden Triangle (Johnson *et al.*[74])	200≤MW≤500, −2≤clogD≤5
Pfizer rule (Lewis *et al.*[75])	clogP>3, PSA<75
Hudson *et al.*[47]	MW(Da)≤305, logP≤1.9, HBD≤4, HBA≤7, HBD+ HBA≤11, PSA(A^0)≤65, ROT≤7, NATOMS≤40, NRING≤2, Halogens≤2
bRo5 (Doak *et al.*[19,20])	MW(Da)> 500, clogP/clogD <0 or >7.5, HBA>10, HBD>5, PSA(A^0)>200, ROT>20
CNS drug space (Wager *et al.*[76])	MW=305.3Da, clogP/clogD= 2.8/1.7c, HBD=1, PSA=44.8 (A^0), formal charge=8.4c.
eRo5 (Doak *et al.*[17])	MW 500–700(Da), clogP range 0–7.5, HBD≤5, HBA≤10, PSA(A^0)≤200, ROT≤20.
Ursu *et al.*[42]	MW(Da)≤324.8, clogP≤2.43, HBA≤5, HBD≤1, PSA(A^0)≤67.2, ROT≤4
Hudson *et al.*[14]	MW (Da)≤305, logD≤3.5, logP≤1.9, HBD≤4, HBA≤7, PSA(A^0)≤65, ROT≤7, NATOMS≤40, NRING≤2, Halogens≤2

[a] *bRo5* beyond Ro5, *CNS* Central nervous system, *eRo5* expanded Ro5, *Halogens* number of halogens, *HBA* number of hydrogen bond acceptors, *HBD* number of hydrogen bond donors, *logD* distribution coefficient, *logP* octanol/water partition coefficient, *MW* molecular weight, *NATOMs* number of N and O atoms, *NRING* number of rings, *PSA* polar surface area, *ROT* number of rotatable bonds, *RoX* Rule of *X*.

We refer the reader to two reviews. First is an earlier review of computational tools to set up a suitable formulation strategy for bRo5 compounds and molecular requirements for different targets. These authors suggested that enabling a formulation strategy for the prediction of 'formulate-ability' may increase the number of orally tractable targets.[24] Second is the recent review of chemical and pharmaceutical approaches to help overcome structure-related problems and achieve acceptable ADMET properties, this *via* chemical modifications and pharmaceutical technology applications to improve improving drug-likeness[25] (see also the recent review by Hudson[16]).

In recent years, machine learning has played an increasing role in determining the robust classification of druggable molecules.[26] In particular, research has shown that recursive partitioning (RP), Random Forests (RFs),[27] support vector machines (SVMs),[28] and Artificial Neural Networks (ANNs)[5] have been commonly employed in this arena.[16,29–38] Classification methods such as RP, naïve Bayesian,[39] and SVM[34] approaches have been used to categorise small molecules, into druggable and non-druggable compounds. In ADMET evaluation[40] naïve Bayesian methods were shown to provide more accurate results than RP and SVM.[39] In contrast, SVM showed very good classification results compared with other classification methods in early pharmaceutical research.[41]

High throughput docking of small molecule ligands (candidate drugs) into high-resolution protein structures is now standard in computational approaches to drug discovery.[42] High throughput docking does well in correctly identifying the binding modes of ligands, but associated scoring functions have poor success rates in predicting binding affinities.[43] Interestingly, a hybrid Self Organising Map ANN approach was recently suggested as a computational tool for the evaluation of docking experiments of calpain ligands (small drug molecules) for the treatment of cataracts.[44,45] These authors showed the value of adding chemical–physical structure for improved identification of good candidate drugs for the classification of calpains for the treatment of cataracts. More recently Hudson *et al.*[14] studied the performance of SVM and RP methods based on 10 physico-chemical MPs, to successfully classify small molecules (ligands) with low or high violator scores (defined by an optimal cutpoint, C=5, as based on skew normal mixtures[46] and on logit models. RP and SVM were then used to classify the high-score molecular violators from the low-score ones (<5). They also showed that SVM used in combination with simple molecular descriptors provided a reliable assessment of a scoring function of counts of violations of MPs which were suggested earlier by Hudson *et al.*[47] to partition molecular chemo-space into so-called druggable *versus* non-druggable molecules.

We refer the reader to a summary given in Table 11.2 of a selection of machine learning (a subset of artificial intelligence (AI)) studies for druggability prediction based on physico-chemical MPs. Noteworthy also is that while AI and the more recent development of deep learning (DL) are not new, recent advances in high-performance computing, the availability of large annotated data sets required for training, and the development of novel frameworks for implementing deep neural networks, have led to an unprecedented acceleration of the field of both pharmacogenomics and molecular (network) biology. We refer the reader also to a recent review of DL methods in drug discovery[16] and a review on machine learning methods used for predicting drug-likeness of molecules and their disease or organ category.[26]

Table 11.2 Machine Learning for druggability prediction based on physico-chemical molecular properties.[a]

References	Dataset	Machine learning algorithm	Results
Nayal & Honig[77]	PDB DrugBank	RF	Balance Error rate=7.2%, C=77%, Accuracy=87.02%, C=74%
Li *et al.*[78]			
Hajduk *et al.*[79]	NMR-based screening dataset	Regression analysis	R^2=72, adjusted R^2=0.65, leave one out Q^2=0.56
Zhu *et al.*[80]	DrugBank	SVM	AUC=69.21%, Accuracy=90%, SD <2%
Volkamer *et al.*[81]	NRDD		

Costa *et al.*[82]	Yıldırım *et al.*[83]	Decision tree-based meta-analysis	AUC=82%, Precision=74.8%
Jeon *et al.*[35]	DrugBank, Therapeutics Target Database	SVM-recursive feature elimination (SVM-REF)	Avg. accuracy=91.69%, Avg. specificity=91.9%, Avg. AUC=78%
Jamali *et al.*[34] Mayr *et al.*[36]	Drug Bank	SVM, Neural Network, k-Nearest Neighbourhood, NaïveBayes, RF and Decision Tree	Accuracy=89.78–95%%, MCC=0.79, AUC=95.4%
Ofran *et al.*[84]	PDB	FragFEATURE	Precision=72%, recall=82% on average.
Prasanthi *et al.*[85]	PDB	SVM and decision trees	Classification accuracies for druggable proteins in the range 64%–71% and for non-druggable proteins, in the range 85%- 85.8%.
Hudson *et al.*[47,86]	DrugBank	SOMs of MPs	Score based on 9 predictors excluding logD but using logP correctly classified 66.43% of druggable (Ro5 compliant) molecules and 99% of non-druggables. Accuracy=82.18%
Hudson *et al.*[47]	DrugBank	Gaussian Mixture and Hidden Markov clustering models, Bayesian clustering and SVM	Best non-Bayesian mixture models exhibit 88% SE and 95% SP. Bayesian models – 88% SE and 100% SP. Best SVM model – 99% SE and 98% SP.
Chatterjee *et al.*[87]	nrDSSP	Feed forward neural networks	Prediction accuracy range, 75.58%–77.48%.
Hudson *et al.*[44,45]	Calpain Inhibitor dataset	SOMs and ANN, K means hybrid clustering	SOMs significantly reduced the number of false negatives by 64% and false positives by 26%.
Hudson *et al.*[14]	DrugBank	Skew normal mixture classification, SVM & RP	SVM achieved an AUC=98.1%, CI (97% – 99.2%), SE=93%, SP=98%; *C*=0.82. RP achieved an AUC=95.3% CI (93.1% – 97.5%), SE=92%, SP=89%, C=0.809.

[a] *ANN* Artificial Neural Network, *AUC* area under the curve, *logD* distribution coefficient, *logP* octanol/water partition coefficient, *MPs* molecular parameters, *RF* Random Forest, *Ro5* Rule of Five, *RP* recursive partitioning, *SOMs* self-organising maps, *SVM* support vector machine.

As implied above, lead discovery or lead optimisation involves testing the structure–activity relationship of compounds, after which selected compounds are synthesised and screened experimentally. Analysis of the structural and physico-chemical properties of compounds has been shown to increase a candidate molecule's probability of activity for a specific target, as well as the molecule's eventual survivability through preclinical development. As such there have been a number of investigators that have sought to improve Lipinski's Ro5 filter by means of additional MPs and to create different violation cut-off points for each MP.[14,15,48] Numerous physico-chemical properties,[49–52] including Lipinski's Ro5,[3,6] the latter known as the rule of thumb, have been applied to improve ADMET properties.[10,40,53] It is important to note that the type and the length of the molecular descriptor vector influences the details of the chemical representation, and the choice of the feature set is driven by the expected depth of description. A recent study[14] studied the viability of scores involving the original 4 variables of Lipinski's Ro5 and an additional 6 MPs (PSA, number of rotatable bonds (ROT), number of rings (NRING), number of halogens (Halogens), number of N and O atoms (NATOMs), and logD). They compared the best cutpoint, *C*, for a rule violations testing *C*=3, 4 or 5 and it was determined that the most effective cutpoint that optimised drug-like classification was less or equal to 5 violations.[14] A list of main ADMET based rules of druggability is given in Table 11.1.

The aim of this chapter is two-fold: (i) to propose computational methods to identify druggable disease targets and (ii) to establish which of ANN or RF procedures and which scoring functions best partition disease target space. This has not been formally established for the data set studied earlier,[14] which showed indirectly that

molecules with more than 5 violations were empirically associated with specific disease targets, but this study did not test machine learning methods using the actual target scores associated with the 172 disease targets, as was recently performed in Hudson *et al.*[15] and extended to encompass Principal Component Analysis (PCA) based metrics of the targets in this chapter. Recent preliminary work by Leemaqz *et al.*,[54] using SVM and RPs directly on the median scores of molecules associated with specific disease targets, showed the superiority of SVM over RP to partition targets with a median logD based score of 4 or more. In this chapter, we extend this work to ANN and RF analyses and use both the non-PCA constructs (based on raw data) and find corresponding PCA constructs based on both the MPs and the targets. We use cutpoints which are aligned with the cutpoints as reported in Hudson *et al.*[14,15] and in part modify the cutpoints of Leemaqz *et al.*[54]

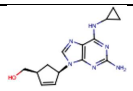
The score function evaluated in this chapter for both the raw and the PCA based MPs and based on the target median scores comprises the 4 traditional parameters of the Ro5,[1–3] plus 5 extra MPs PSA, ROT, NRING, NATOMs, and either logP and logD. Recently, logD was suggested as a preferable predictor for permeation[11,12] compared with the more traditionally used Lipinski's permeation coefficient logP.[13] Hudson *et al.*[14,47] developed druggability rules for molecules (scores counting violations) that account for physico-chemical properties, and derived novel cutpoints for each of the 10 possible MPs (which included both logP and logD, as candidates). These cutpoints, reported in Table 11.1, were based on a mixture clustering discriminant analysis approach,[55] and were shown to be in substantial agreement with recent cutpoints reported for a subset of MPs[14,42] (Table 11.1). We use these MP cutpoints[14,47] in this chapter to derive a scoring function of violations per molecule and for each of the targets, testing cutpoints C=3, 4 or 5.

11.2 Data and Methods

11.2.1 Data and Druggability Scoring for Molecules and Disease Targets

We analysed 1279 small molecules from the DrugBank database,[56] a unique bioinformatics and chemo-informatics resource combining detailed drug (*i.e.* chemical, pharmacological, and pharmaceutical) data with drug target (*i.e.* sequence, structure, and pathway) information, containing 6711 drug entries (a candidate molecule is shown in Table 11.3.). In total, there were 105 Ro5 non-compliant molecules, 681 with oral and 598 with non-oral delivery modes. In this study, 172 targets representing 1279 molecules were found and the median score value obtained for the molecules in the given target; with 99 targets predominantly oral and 73 non-oral in terms of modes of delivery.

Table 11.3 DrugBank3.0 information on one candidate molecule (DB01048 Abacavir).

DrugBank ID & Name & CAS number	Molecular weight formula	Chemical structure	Categories	Therapeutic indication
DB01048 Abacavir 136470-78-5	286.3323 $C_{14}H_{18}N_6O$		Anti-HIV Agents/Nucleoside and Nucleotide Reverse Transcriptase Inhibitors/Reverse Transcriptase Inhibitors	For the treatment of HIV-1 infection, in combination with other antiretroviral agents

Specifically, we investigated the MP score functions based on recent filter scores,[14] where molecules with violations of less or equal to 4 (or 5), were considered a druggable molecule. The score function per molecule sums the number of violations based on 8 or 9 MPs modelled. The filter scores were created as follows for each MP: lipophilicity logP≤1.9, hydrophobicity logD≤3.5, MW≤305 daltons, HBD≤4, HBA≤10, PSA≤140 angstroms, ROT≤7, ring numbers≤2, and NATOMs≤40 (see Table 11.1). Note the cutpoint for logD of 3.5 is smaller than 5.5 as suggested earlier.[13] In this chapter, we test models including either of logD or logP as lipophilicity variants and apply ANN and RF methods to test models to classify the score based partitions for both the raw MP data or raw target median data along with corresponding derived PCA based dimensions (Dim) of both targets and MPs.

11.2.2 Description of the MPs and Targets by Score Function Strata

Table 11.4 shows the description of the MPs analysed, stratified by score logD for cutpoints C=4 or C=5, where the MP parameter set excludes logP, as logD was found to be optimal. In addition, the number and percentage of molecules within the Ro5, bRo5, and eRo5 regions in chemo-space, as defined in Table 11.1, are given in Table 11.4, as are the number and percentage of molecules recently identified as good druggables *via* skew normal mixture analysis.[14] This latter study, summarised in this chapter, identified 928 and 351 molecules with score ≤4 and >4, respectively (Table 11.4). Of the 928 low score violator molecules, 86.2% were also identified as good druggables by skew normal mixtures.[14] Whereas of the 351 high score violator molecules only 53% were identified as good druggables by skew normal mixtures. While 1174 molecules follow the Ro5, only 34 (3%) of these molecules satisfied the eRo5 criteria, of which 11 molecules were outside the Ro5 region, 6 of which (18%) had a low score (≤4), with 82% being high scoring >4 violators (see Table 11.4). In the full data set 614 molecules were found to be in the bRo5 space, all of which satisfied the Ro5 criterion, 23% had a score ≤4 in contrast to 77% exhibiting high scores >4 (see Table 11.4).

Table 11.4 Description of the mean MPs stratified by score logD for cutpoints C=4 or 5.

Score logD	No. of molecule *n* (%)	MW	logP	logD	HBA	HBD	PSA	ROT	NATOM	Nring	Ro5 druggable *n* (%)	bRo5 *n* (%)	eRo5 *n* (%)	Skew normal good druggables *n* (%)
≤4	928 (72.6)	262.7	1.7	1.6	3.7	1.5	63.9	3.5	35.4	2.1	926 (99.8)	473 (51.0)	6 (0.6)	800 (86.2)
>4	351 (27.4)	519.5	1.8	1.4	8.7	3.6	146.7	8.2	68.7	3.7	248 (70.7)	141 (40.2)	28 (8.0)	186 (53.0)
≤5	1068 (83.5)	282.4	1.8	1.7	4.1	1.6	68.3	4.0	38.0	2.2	1058 (99.1)	559 (52.3)	14 (1.3)	904 (84.6)
>5	211 (16.5)	590.4	1.4	1.0	10.2	4.6	179.1	9.0	77.2	4.0	116 (55.0)	55 (26.1)	20 (9.5)	82 (38.9)

Table 11.5 reports the top twelve targets with the highest molecular representation stratified according to median score 4 logD and oral status. Figure 11.1 displays part of the molecular descriptors pattern for these targets, showing different patterns of distributions across targets.[14] A description of the target based medians stratified by score logD for cutpoints C=4 or 5 is given in Table 11.6. We identified 37 targets (21.5%) with high scores >4, which were invariably associated with significantly higher target medians for all parameters, except logD and logP, where the latter two ADMET properties are related to hydrophobicity or lipophilicity of molecules. This similarity reflects in part an earlier finding that showed that high scoring violators (whether targets or MPs) have a propensity for negative values for logD and/logP.[14,15] A similar pattern is observed for the 18 targets with score >5 (see Table 11.6).

Table 11.5 Description of the top 12 targets in terms of Score 4 logD and oral status.

Target	Total no. (*n*) of molecules	Median Score 4 logD	Number of oral molecules	Oral status
Adjuvant	25	1	14	Oral
Anti-anxiety	26	3	20	Oral
Antihypertensive	35	4	17	Non-oral
Dietary	40	1	11	Non-oral
Antineoplastic	57	3	28	Non-oral
Anti-inflammatory	61	3	40	Non-oral
Anti-Infective	40	1	23	Oral
Anti-Bacterial	92	6	42	Non-oral
Anaesthetics	30	1	6	Non-oral
Analgesic	48	2.5	30	Oral
Adrenergic	98	3	62	Oral
Anti-Allergic	27	4	12	Non-oral

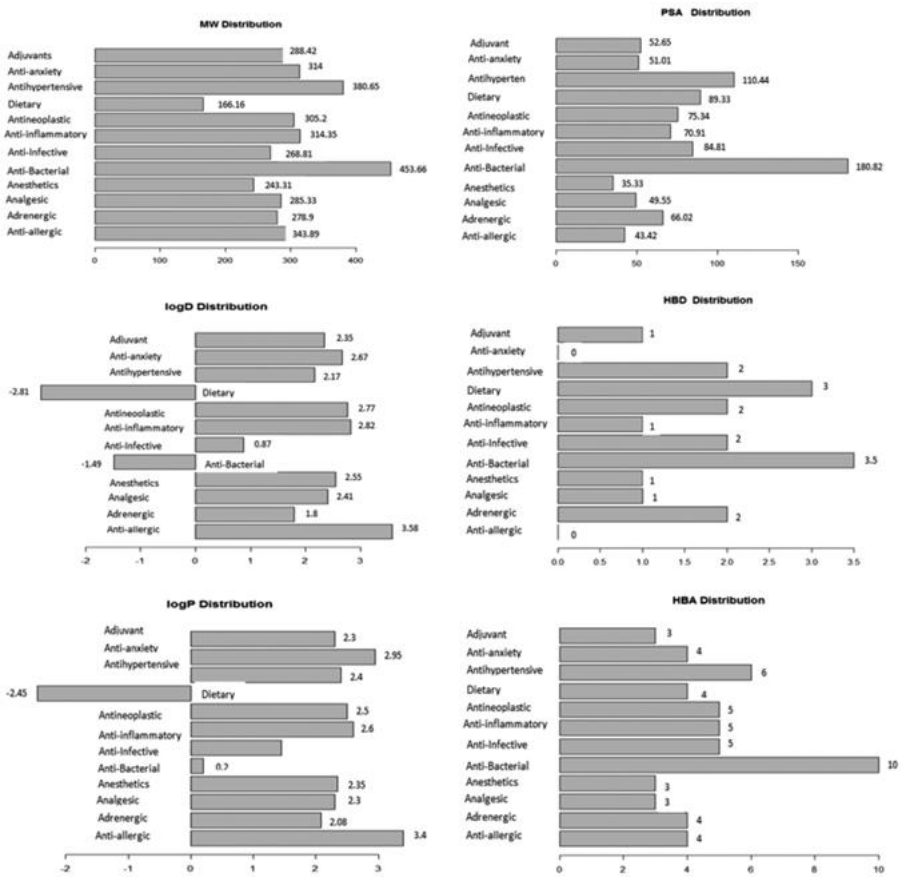


Figure 11.1 Molecular descriptor pattern for top 12 targets (sourced from Hudson *et al.*[14]).

Table 11.6 Description of the target-based medians stratified by score logD for cutpoints *C*=4 or 5.

Target Score logD	No. targets *n* (%)	Median MW	Median logP	Median logD	Median HBA	Median HBD	Median PSA	Median ROT	Median NATOMS	Median NRING
≤4	135 (78.5)	285.7	2.0	2.0	3.8	1.5	63.4	4.0	38.2	2.2
>4	37 (21.5)	475.3	2.3	2.1	7.2	2.9	122.4	8.4	64.3	3.5
≤5	154 (89.5)	303.5	2.1	2.1	4.0	1.5	66.1	4.5	41.0	2.3
>5	18 (10.5)	523.4	1.4	1.0	9.0	4.1	161.4	9.1	68.1	3.7

11.2.3 Statistical and Program Approach

11.2.3.1 PCA Analyses of the Molecular and Target Data

It has been appreciated for some time that chemical space visualisations invariably need to reduce the dimensionality inherent in the process of comparing molecular structures. This is often achieved *via* algorithms such as PCA and recently through *t*-distributed stochastic neighbour embedding.[57] To visualise the distribution of the molecules, and primarily the targets in chemical space, we performed a PCA[58] on the set of physico-chemical descriptors described above (see also Table 11.1). PCA is a statistical method for variable reduction that allows multidimensional data to be visualised using two- and three-dimensional plots, which incurs minimal loss of information from the original dataset. As detailed by Hudson *et al.*[14] it is well known that several of the molecular descriptors analysed in this chapter are correlated, in which case PCA is appropriate. PCA implements a linear transformation to rotate the matrix of variables onto a set of orthonormal axes that then define the dimensions of

greatest variance for the dataset.[59] The resultant axes are called principal components or dimensions (Dim) and represent linear combinations of the original variables (descriptors). The matrix rotation preserves Euclidean distances and maximises the fraction of total variance from the original dataset on each successive principal component. Via this transformation, the first principal component (PC1 or Dim1) retains the greatest fraction of variance from the original dataset, the second principal component (PC2 or Dim2) contains the next largest fraction, and so on. In this way, an n-dimensional dataset (in our analysis n=10) can be visualised using an m-dimensional plot of principal components (where $m<n$) with minimal loss of information. The PCA was performed to find new and reduced dimensions based on both the MP data and on the target medians data.

11.2.3.2 ANN and RF Approach

The dataset was partitioned into 70% and 30% for training and testing purposes, respectively, to avoid model overfitting. The success of the model was determined by assessing the area under the curve (AUC) of the receiver operating characteristic (ROC) curve, a measure used to predict model accuracy.[60] For replication, all models were created using a seed of 44. RF models were based on 10-fold cross validation on the training partition with a repetition of 5. As RF is considered an accumulative model algorithm, the only tuning parameter used was the number of trees. The averaged model across the specified number of trees was then assessed on the testing data. ANN models were created using a 10-fold cross validation on the training partition and repetition of 5, with models assessed *via* the training partition. All possible variable set scores were scaled and centred before being integrated in the model. The final model was created using a tune grid containing decay (to avoid overfitting *via* regularisation) parameters between 0.1 to 0.5 by a value of 0.1 and size parameters (the number of hidden nodes in the network) between 1 to 10 used. Each of these steps was conducted across each of the different score functions: score C logP and score C logD, for C=3, 4 or 5. This ensured the best machine learning method was identified and the most effective score function, whether based on raw data or the PCAs. RStudio, an integrated statistical software package, was utilised to complete all analyses. In particular, the Husson *et al.* FactoMineR package[61] in RStudio was used to complete all PCAs. The Kuhn Caret package[62] was used for all classification and the Beck *et al.* NeuralNetTools[63] used for the visual plots and ROC analyses for the neural network models.[64]

11.3 Results

11.3.1 PCA Results

Table 11.7 and Table 11.8 report the eigenvalues of the target-based PCA and the PCA loadings, respectively, for the 8 parameters analysed (the logD model, which excludes logP, and the latter is shown to be less optimal than logD). Taken together, the first three principal components (Dim1, Dim2, and Dim3) retain 89.24% of the full 8-dimensional target data (see Table 11.7). The PCA component loadings given in Table 11.8 aid in the assessment of the influence of the original 8 parameters on the distribution of the targets and are displayed in the loading plots of (Dim2 *vs* Dim1) and (Dim3 *vs* Dim1) in Figure 11.2. The distribution of the target violators with score 4 (>4) *versus* the non-violator targets in two-dimensional space is shown in the PCA plot of Dim2 *vs* Dim1 in Figure 11.3.

Table 11.7 Eigenvalues from PCA of target data.

PC DIMs	Eigenvalue	Percentage of variance	Cumulative percentage of variance
Dim 1	4.40	55.01	55.01
Dim 2	1.80	22.52	77.52
Dim 3	0.94	11.72	89.24
Dim 4	0.32	3.95	93.19
Dim 5	0.25	3.08	96.26
Dim 6	0.14	1.75	98.02
Dim 7	0.09	1.08	99.10
Dim 8	0.07	0.90	100.00

Table 11.8 PCA-based target loadings.

Variables	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
MW	0.41	0.31	−0.01	0.05	0.19
HBD	0.39	−0.30	−0.10	−0.38	0.60
HBA	0.41	−0.16	−0.03	0.79	−0.05
PSA	0.43	−0.24	−0.07	0.08	0.09
ROT	0.32	0.14	0.69	−0.10	−0.37
NATOM	0.39	0.38	0.08	−0.35	−0.10
NRING	0.19	0.45	−0.68	−0.04	−0.30
logD	−0.19	0.61	0.21	0.29	0.59

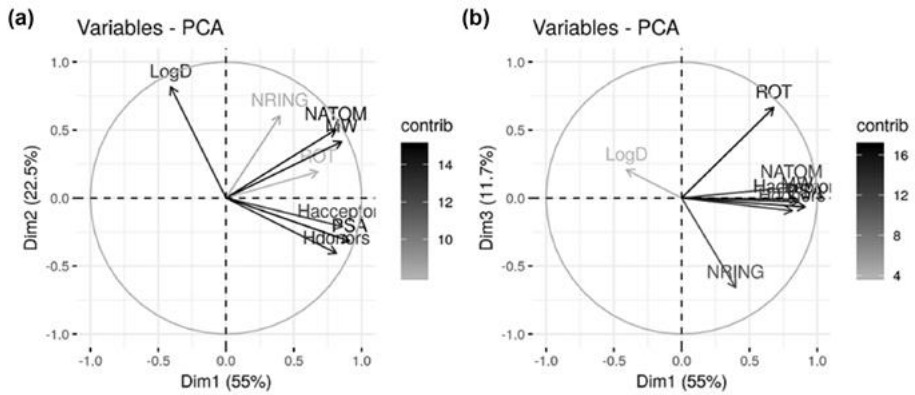


Figure 11.2 PCA Loading plots for the target median data. Vectors on the loading plot indicate the relative influence of each structural and physicochemical descriptor on the placement of targets on the plot of Dim2 *vs* Dim1.

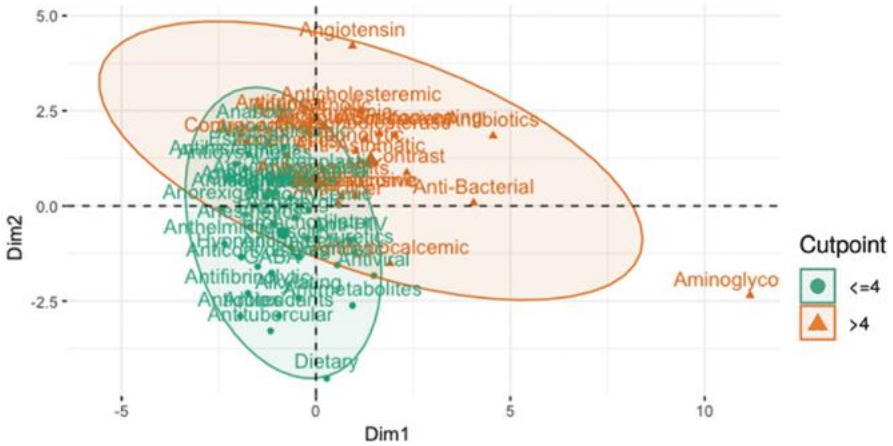


Figure 11.3 PCA plots of Dim2 *vs.* Dim1 for the target median data. Quadrants: A=top right, B=top left, C=bottom left, and D=bottom right.

As seen in Table 11.8, the variables that contributed most to Dim1 were MW, HBD, HBA, PSA, and NATOMs. logD contributed most to Dim2, and ROT and NRING contributed the most to Dim3. We conclude that Dim1 explains flexibility, Dim2 explains lipophilicity, and Dim3, due to the negative relationship between ROT and NRING, represents cell permeability.

The distribution of the score >4 targets is shown to be more spread along Dim1, and the range of the targets with score ≤4 seems more pronounced on Dim2 (see Figure 11.3). Positioning of compounds along Dim2 is governed largely by logD (and by high values of PSA, HBA, and HBD), which influence compounds in a positive (upward)

and negative (downward) direction, respectively (Figures 11.2 and 11.3). In addition, logD influences the positioning of targets negatively (leftward) along Dim1 and positively (rightwards) by (MW, NATOM, and ROT) along Dim1. A larger portion (70%) of the targets with high scores >4 cluster in the upper right (quadrant A) of the PCA plot (Figure 11.3, see also Table 11.9) and to a lesser extent (22%) are in the lower right region (quadrant D), 6 of which (75%) are oral targets (see Table 11.9). For the non-violator targets with scores ≤4, 35% and 37% cluster in the upper left (quadrant B) and lower left (quadrant C), respectively – with equivalent percentages being oral or non-oral low score targets in each quadrant B and C (see Figure 11.3, Table 11.9). PCA analysis successfully identified a group of disease targets with a specific score function range and position of ADMET descriptors in different quadrants. Indeed Table 11.9 confirms a significant association between the score 4 logD classes and the quadrants for all targets (P<0.0001) and also for targets stratified by both score 4 status and oral status (a significant score 4 status by oral status interaction effect, P<0.0001).

Table 11.9 Types of targets, which occupy different quadrants of Figure 11.3.

Disease categories status		QA	QB	QC	QD	
Total number of disease target per quadrant		46	50	50	26	Total number of targets
Total number of molecules		225	352	370	257	
Total number of oral and non-oral disease target per quadrant	Oral	26	27	24	11	88
	Non-oral	20	23	26	15	84
	P value	P>0.20				
Total number of disease target based on score 4	Score 4≤4	20	47	50	18	135
	Score 4>4	26	3	0	8	37
	P value	P<0.00001				
Total number of disease target based on score 4 oral by druggable status	Score 4≤4 oral	11	26	24	9	70
	Score 4>4 oral	15	1	0	2	18
	Score 4≤4 non-oral	9	21	26	9	65
	Score 4>4 non-oral	11	2	0	6	19
	P value	P<0.00001				

11.3.2 Machine Learning Classification Results

The AUC and 95%CI of each machine learning model (RF and ANN) for both the raw MPs data and the raw disease target data across score functions (with varying cutpoints and model variations, founded on whether logD or logP were included in the model) are given in Table 11.10. Based on earlier work,[14,15] only score 4 or 5 cutpoints were tested for the MPs, and only scores 3 and 4 reported for the raw target data (see Leemaqz *et al.*[54]). The corresponding RF and ANN AUC and 95%CI values based on the PCA constructs for MPs and the targets are given in Table 11.11.

Table 11.10 AUC (95% CI) for RF and ANN models for both the raw MPs and raw Disease Target data.

	RF on MPs	ANN on MPs	RF on Targets	ANN on Targets
Score functions	AUC (95% CI)	AUC (95% CI)	AUC (95% CI)	AUC (95% CI)
Score 3 logP	—	—	0.82 (0.71, 0.93)	0.82 (0.71, 0.93)
Score 3 logD	—	—	0.84 (0.73, 0.94)	0.80 (0.69, 0.91)
Score 4 logP	99.58 (99.18–99.98)	99.29 (98.72–99.85)	0.85 (0.70, 1.00)	0.72 (0.54, 0.90)
Score 4 logD	99.70 (99.42–99.99)	99.27 (98.59–99.70)	0.85 (0.70, 1.00)	0.89 (0.77, 1.00)
Score 5 logP	99.16 (98.51–99.82)	98.86 (98.04–99.69)	—	—
Score 5 logD	99.33 (98.73–99.92)	99.01 (98.22–99.80)	—	—

Table 11.11 AUC (95% CI) for RF and ANN models based on the PCA Dimensions for both the MPs and the Disease Target data.

Score functions	RF on PCAs of MPs	ANN on PCAs of MPs	RF on PCAs of Targets	ANN on PCAs of Targets
	AUC (95% CI)	AUC (95% CI)	AUC (95% CI)	AUC (95% CI)
Score 3 logP	0.92 (0.90,0.95)	0.94 (0.91,0.96)	0.79 (0.73, 0.93)	0.80 (0.69,0.91)
Score 3 logD	0.92 (0.90,0.95)	0.94 (0.91,0.96)	0.80 (0.70,0.90)	0.81 (0.67,0.90)
Score 4 logP	0.89 (0.85,0.93)	0.92 (0.89,0.96)	0.74 (0.58,0.93)	0.91 (0.79,1.0)
Score 4 logD	0.92 (0.89,0.95)	0.95 (0.87,0.94)	0.75 (0.58,0.93)	0.92 (0.81,1.0)
Score 5 logP	0.88 (0.83,0.93)	0.89 (0.84,0.94)	0.88 (0.63,1.00)	1.00 (1.00,1.00)
Score 5 logD	0.89 (0.84,0.94)	0.88 (0.83,0.93)	0.75 (0.47,1.00)	0.99 (1.00,1.00)

11.3.2.1 Results of the Analyses Based on the PCA of the MPs and the Raw Target Data

Table 11.10 reports the RF and ANN models on the raw target data and demonstrates that RF performed better than ANNs for the score 3 variant models, but that the ANN score 4 logD model was optimal overall, with AUC (95% CI) of 0.89 (0.77, 1.00). The neural network plots for the 4 models based on the raw target data for the score C logP and score C logD (C=3 and 4) partitions are given in Figure 11.4. Table 11.11 shows that overall the ANN classification was superior to the RF models for score 3 and score 4 logD models, with the ANN target based PCA with partition C=4 achieving best classification, with AUC (95% CI) of 0.92 (0.81, 1.00). Similarly, for the molecular parameter PCs, the ANN method performed better than RF classification, with the score 4 logD being optimal with AUC (95% CI) of 0.95 (0.87, 0.94) (see Table 11.11). Figure 11.5 displays the ANN plots for the drug target PCA constructs based on score 4 logP and score 4 logD models. Figure 11.6 shows the RF plots for the drug target PCA constructs based on the score 3 logP and logD variants.

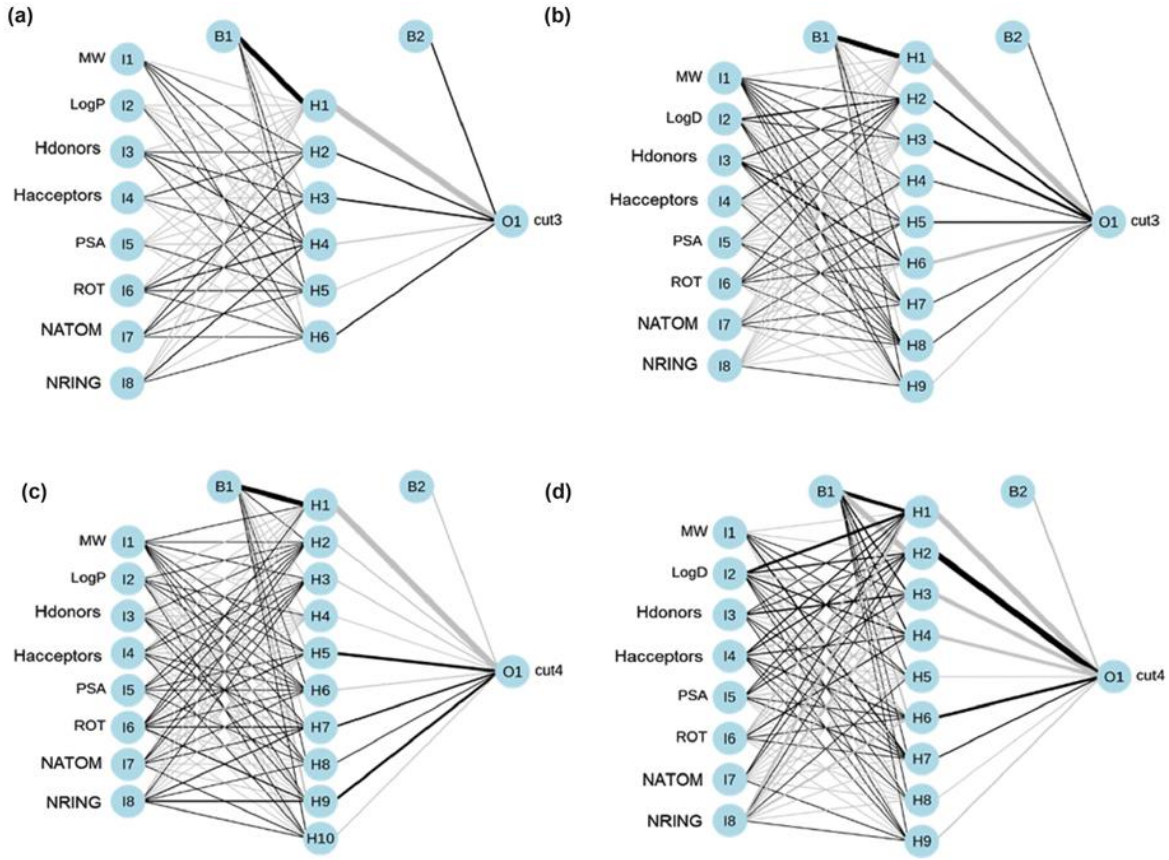


Figure 11.4 Artificial Neural Network plots for Drug Target model: (a) cutpoint 3 logP, (b) cutpoint 3 logD, (c) cutpoint 4 logP, and (d) cutpoint 4 logD. Sourced from Hudson *et al.*[14].

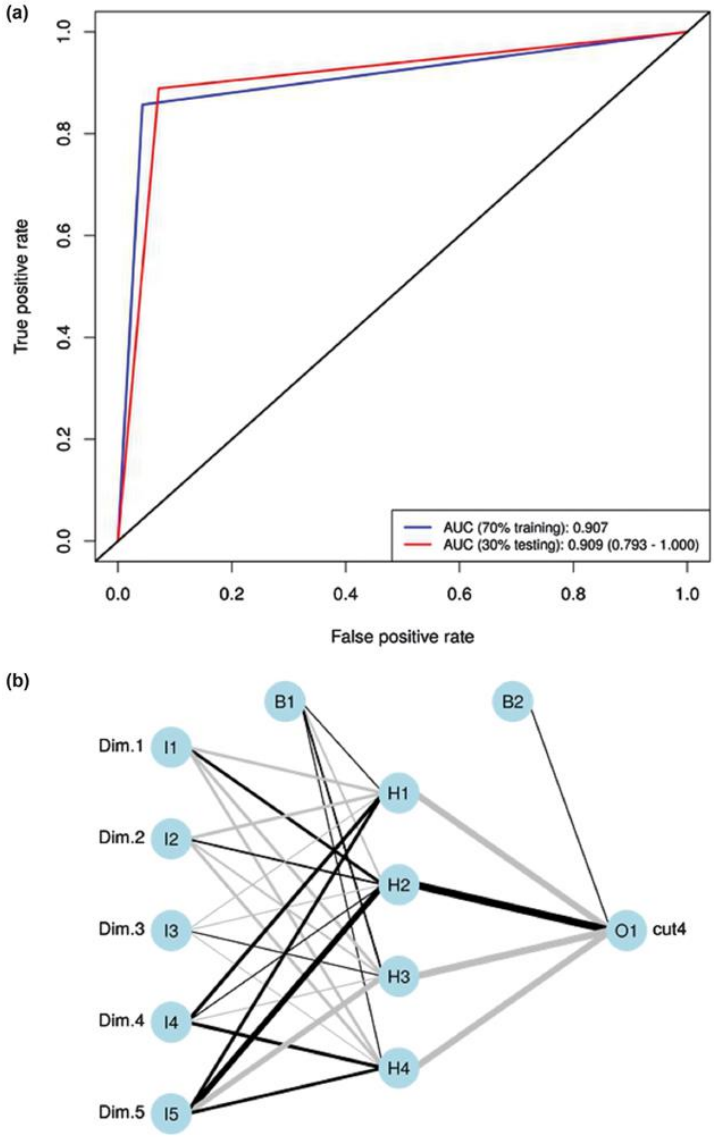


Figure 11.5 Neural Network plots for the optimal Drug Target PCA models: (a) cutpoint 4 logP and (b) cutpoint 4 logD.

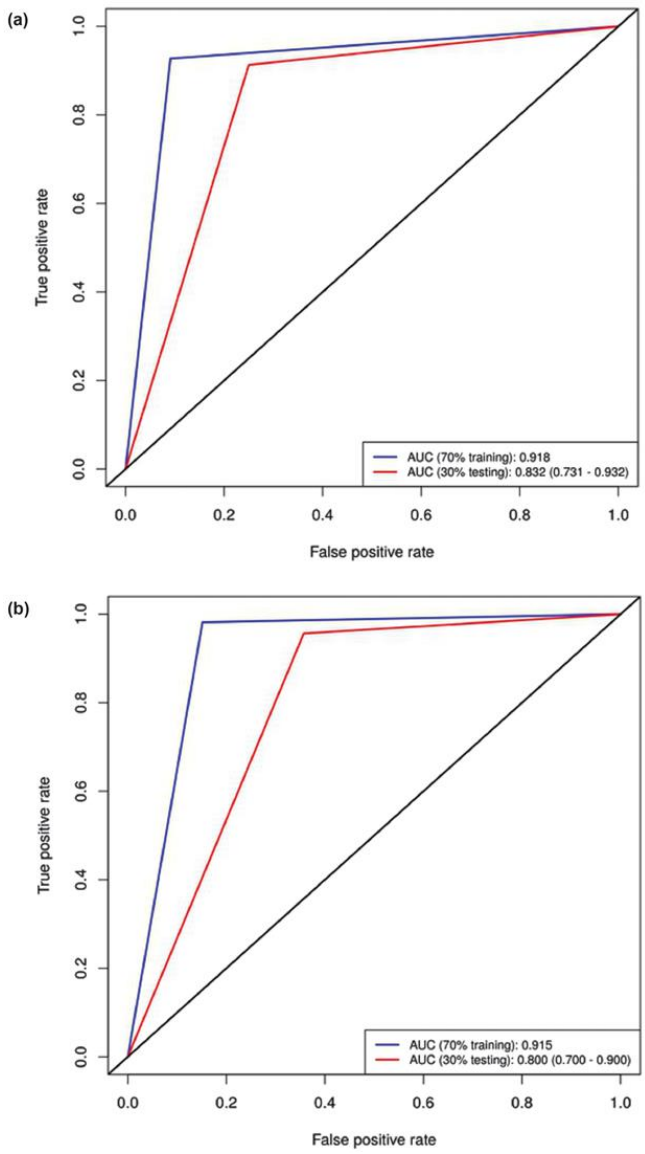


Figure 11.6 Random Forest plots for the optimal Drug Target PCA models: (a) cutpoint 3 logP and (b) cutpoint 3 logD.

11.3.2.2 Mapping Variable Importance from ANN Analyses to the PCA Loadings of the Target Data

For the superior ANN score 4 logD model based on the raw target data given in Table 11.10, the corresponding variable importance measures, in decreasing order of magnitude, were attributable to PSA, logD, NRING, HBA, MW, NATOMS, ROT, and HBD with a corresponding variable importance vector of 21.65, 19.53, 13.18, 11.48, 11.06, 9.5, 7.52, and 6.08, (see also Table 4 of Hudson *et al.*[14]). This revealed the stronger contribution in the ANN classification of both PSA and logD, compared to MW as is traditionally assumed. From Table 11.8, which reports the PCA loadings of the PCA dimensions founded on the target data studied in this chapter, we note that the variable importance measures above, mirror, in part, the sequence of high absolute values of the PCA loadings on the variables (see Table 11.8) for the dimensions as follows: adding strength to the value and interpretation of the PCA dimensions to classify targets into high *versus* low scorers as shown in Figure 11.3:

- **Dim2:** PSA (loading=−0.24), logD (0.61), NRING (0.45)
- **Dim1:** HBA (0.41), MW (0.41), NATOM (0.39)

- **Dim5:** ROT (−0.37), HBD (0.60).

11.4 Conclusion and Discussion

This chapter demonstrates how PCA and score functions of violations can be used to identify the hidden pattern of druggable molecules and disease targets. Our novel PC analytic MP and target constructs showed that molecules and targets lie in particular locations (quadrants) in molecular space and as stratified by our scoring function cutpoints. Overall violations above 4 mirrored poor targets, which could be plainly differentiated and located in reduced PCA dimensional space (see Figure 11.3) as can also be seen in both the 2D raw MP and 2D target data maps (see Figures 11.7 and 11.8) which, in addition, delineate the bRo5 and eRo5 regions. We found that in the full data set analysed, 614 molecules actually lie within the bRo5 space, all of which satisfied the Ro5 criterion, of which 23% had a score ≤4 in contrast to 77% exhibiting high scores >4. While 1174 molecules follow the traditional Ro5, only 34 (3%) satisfied the eRo5 criteria, of which 11 molecules were outside the Ro5 region, 6 of which 6 (18%) had a low score (≤4), while 82% were high scoring >4 violators.

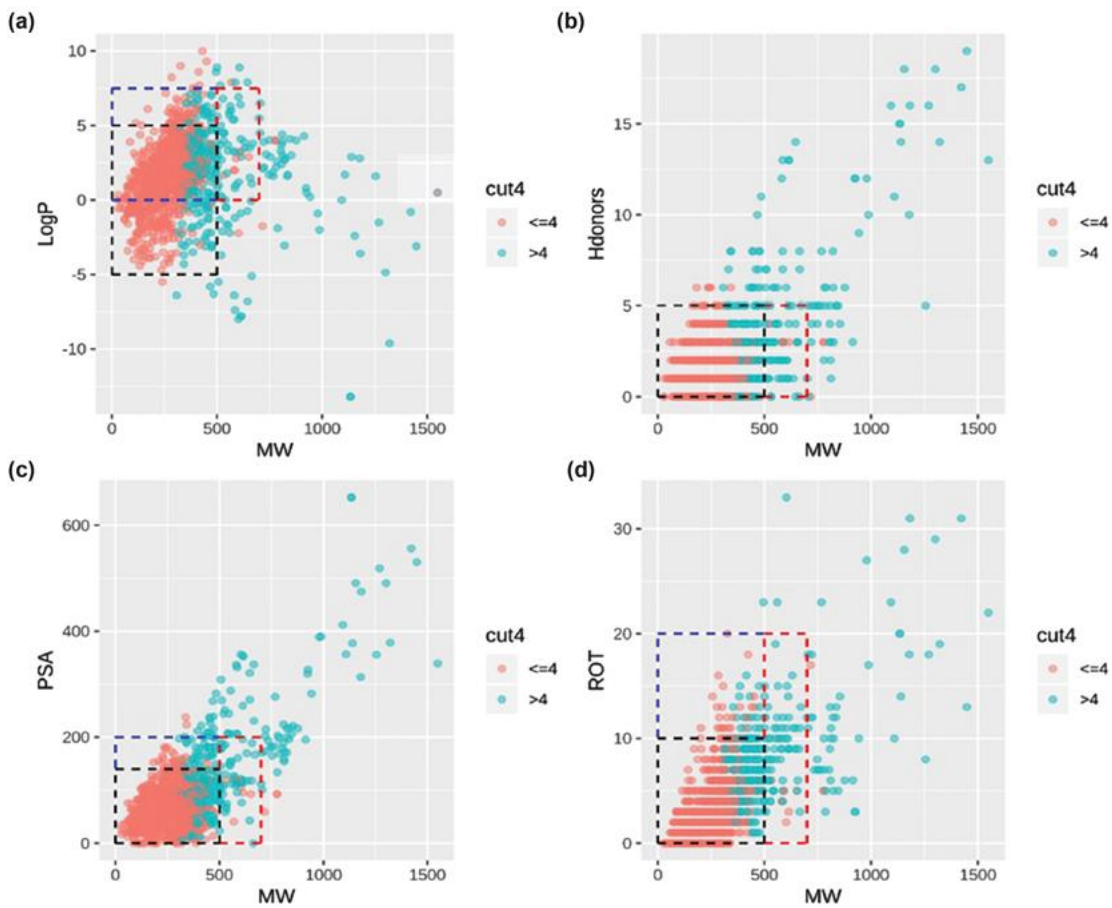


Figure 11.7 Molecular parameter 2D maps: black=Ro5 (with extension) region, blue (to the right of the Figure)=bRo5 region, and red (lighter colours to the left of the Figure)=eRo5 region.

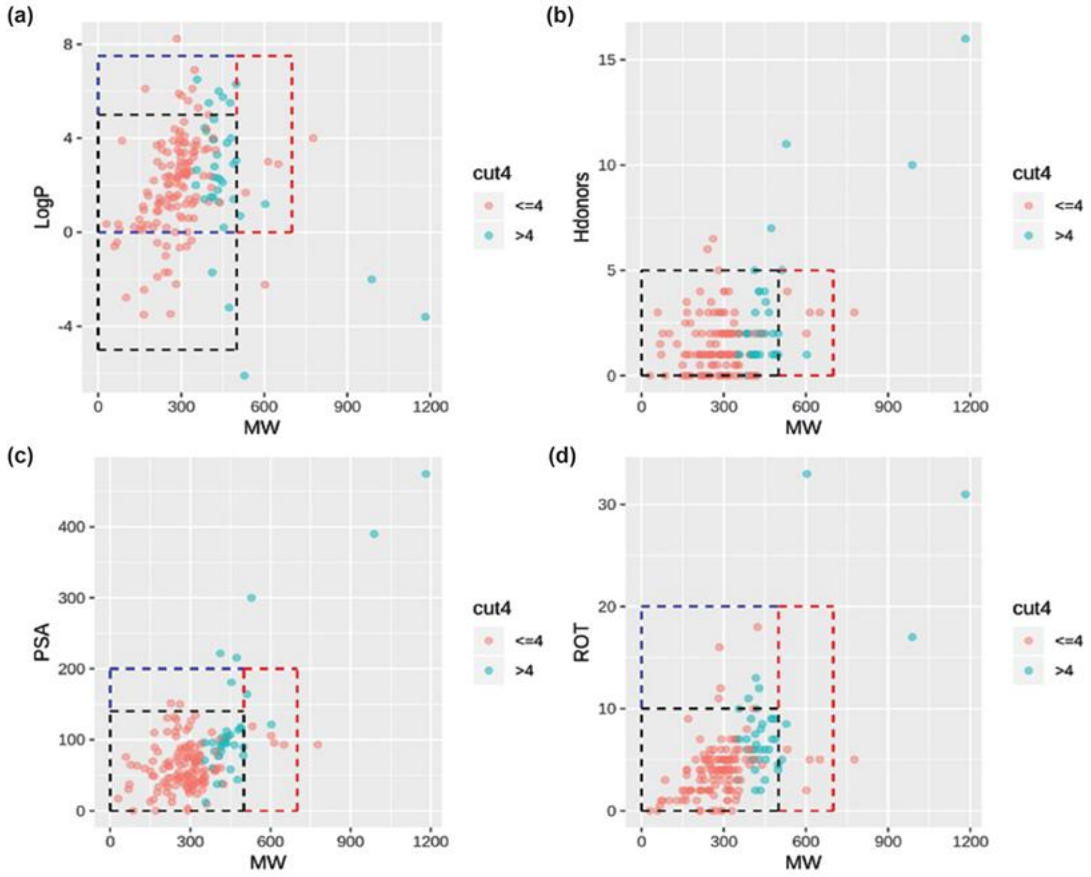


Figure 11.8 Target median data 2D maps: black=Ro5 (with extension) region, blue (to the right of the figure)=bRo5 region, and red (lighter colours to the left of the figure)=eRo5 region (points in the plots are medians for each target).

The RP and ANN rules to classify the high score target and the low-score molecular violators, based on both MPs and the PCA constructs, confirmed the value of logD's inclusion in the scoring function, validating Zafar *et al.*[11,12] and supporting the recently established cutpoints.[14,15] Overall, the RF models and ANNs showed different variable importance profiles. The ANN models performed better than RFs, and the use of logD rather than logP was confirmed. Earlier work[15] had also shown that the most important MPs that influence classification, determined by RF methods, were MW, NATOM, logD, and PSA. By contrast, the ANN target models indicated that PSA and logD were highly important, more so than the traditional parameter MW. Overall, our score 4 partitions using logD were better at classification as shown in all RF and ANN analyses.

Our simple scoring functions of counts of violations partitioned chemospace well, identifying both good and poor druggable molecules and targets. Moreover, ligands with 4 or more violations, were shown to be associated with specific disease targets, *e.g.*, Anti- Bacterial, Antineoplastic, Antihypertensive and Anti-allergic, within which most molecules had a non-oral delivery mode. The Anti-Bacterial target drugs (score 4=6) were shown to have high values for all MPs, consistent with the results of Giordanetto *et al.*,[65,66] who reported that compounds that fall into the bRo5 space (higher MW and PSA) include the Anti-bacterial target. Interestingly, the value of HBA, HBD, and PSA were also high for Anti-Bacterial target drugs (10, 3.5 and 180.82, respectively) with these descriptors representing important hydrogen bond or electrostatic features. We note that these descriptors also displayed high loadings in PCA space. Target drugs with a low median score were *e.g.*, Adrenergic, Dietary, Analgesics, Anaesthetics, Adjuvants, Anti-convulsants, Antimetabolites and Antidepressants, most of which were non-oral. Notably, the Dietary target drugs had low values for the molecular descriptors, with a lowest score 4 of 1.0, lowest median MW and also negative medians for both logP=−2.45 and logD=−2.81.[14,15]

Two decades have passed since the Ro5 piloted the concept of “drug-like” properties. A recent examination of oral drug parameters approved before and after the original Ro5 analysis has demonstrated that while some

parameters such as calculated logP and HBD have remained constant, cutoffs for parameters such as MW and HBA have increased substantially over the past 20 years, thus bringing into question whether MW is a "drug-like" property.[67] Understanding the internal relations in the chemical database remains a key feature for the exploration of the chemical space to further develop new substances with predefined properties. Visualisation of the target chemical space by mapping from multidimensional descriptor space into a reduced dimensional space is still a challenging task in chemo-informatics and computational medicinal chemistry. Deep learning (DL) algorithms and neural networks now offer ways forward. For example, a parametric Stochastic Neighbour Embedding t-SNE approach, which yields a neural-network-based function to map new portions of data, has been suggested recently to aid the interpretation of structurally conditioned biological properties of chemical compounds.[68] Likewise, the exploration of chemical space by core analysis,[69] feature optimisation in high dimensional chemical space by DL,[70] and chemical space exploration guided by DL[68] have been indicated as new approaches. We refer the reader also to a recent review on DL methods in drug discovery[16] and a comparative analysis of machine learning algorithms for the prediction of potential druggable drugs by DrugMiner.[34]

Acknowledgements

The author is most grateful for editorial and graphic design assistance from Dr. Sean A. Hudson (Hudson Software Development, Australia, www.hudson-software.droppages.com).

References

1. C. A. Lipinski, *Drug Discovery Today: Technol.*, 2014, **1**(4), 337.
2. C. Lipinski and A. Hopkins, *Nature*, 2004, **432**(7019), 855.
3. C. A. Lipinski, *Adv. Drug Delivery Rev.*, 2016, **101**, 34.
4. C. Lipinski, *Nat. Rev. Drug. Discovery*, 2012, **11**(12), 900–901.
5. O. Ursu, A. Rayan, A. Goldblum and T. I. Oprea, *WIREs Comput. Mol. Sci.*, 2011, **1**(5), 760.
6. C. A. Lipinski, F. Lombardo, B. W. Dominy and P. J. Feeney, *Adv. Drug Delivery Rev.*, 1997, **23**, 3.
7. C. A. Lipinski, N. K. Litterman, C. Southan, A. J. Williams, A. M. Clark and S. Ekins, *J. Med. Chem.*, 2014, **58**(5), 2068.
8. I. B. Campbell, S. J. F. Macdonald and P. A. Procopiou, *Drug Discovery Today*, 2018, **23**(2), 219.
9. M. F. Rafferty, *J. Med. Chem.*, 2016, **59**(24), 10859.
10. Y. Wang, J. Xing, Y. Xu, N. Zhou, J. Peng and Z. Xiong, *et al.Q. Rev. Biophys.*, 2015, **48**(4), 488.
11. S. Zafar, S. A. Cheema, E. J. Beh, I. L. Hudson, S. A. Hudson and A. D. Abell, MODSIM2013, in *20th International Congress on Modelling and Simulation, Modelling and Simulation Society of Australia and New Zealand*, ed. J. Piantadosi, R. S. Anderssen and J. Boland, December 2013, p. 1945.
12. S. Zafar, I. L. Hudson, E. J. Beh, S. A. Hudson and A. D. Abell, in *31st International Workshop on Simulation and Modeling*, Institute National des Sciences Appliquees, Rennes, France, July 4–8, 2016, p. 163.
13. S. K. Bhal, K. Kassam, I. G. Peirson and G. M. Pearl, *Mol. Pharmaceutics*, 2007, **4**(4), 556.
14. I. L. Hudson, S. Y. Leemaqz, D. Shafi and A. D. Abell, MODSIM2017, 22nd International Congress on Modelling and Simulation, in *Modelling and Simulation Society of Australia and New Zealand*, ed. G. Syme, D. Hatton MacDonald, B. Fulton and J. Piantadosi, 2017, p. 487.
15. I. L. Hudson, S. Leemaqz and A. D. Abell, MODSIM2019, 23rd International Congress on Modelling and Simulation, in *Modelling and Simulation Society of Australia and New Zealand*, ed. S. Elsawah, December 2019, p. 28.
16. I. L. Hudson. in *Methods in Molecular Biology*, ed. H. M. Cartwright, Springer International Publishing, in press, 2020.
17. B. C. Doak and J. Kihlberg, *Expert Opin. Drug Discovery*, 2016, **12**(2), 115.
18. J. P. Overington, B. Al-Lazikani and A. L. Hopkins, *Nat. Rev. Drug Discovery*, 2006, **5**(12), 993.
19. B. C. Doak, B. Over, F. Giordanetto and J. Kihlberg, *Chem. Biol.*, 2014, **21**(9), 1115.
20. B. C. Doak, J. Zheng, D. Dobritzsch and J. Kihlberg, *J. Med. Chem.*, 2015, **59**(6), 2312.
21. I. Gashaw, P. Ellinghaus, A. Sommer and K. Asadullah, *Drug Discovery Today*, 2011, **16**(23–24), 1037.

22. G. V. Paolini, R. H. Shapland, W. P. van Hoorn, J. S. Mason and A. L. Hopkins, *Nat. Biotechnol.*, 2006, **24**(7), 805.
23. F. Giordanetto and J. Kihlberg, *J. Med. Chem.*, 2013, **57**(2), 278–295.
24. C. A. S. Bergström and P. Larsson, *Int. J. Pharm.*, 2018, **540**(1–2), 185.
25. F. Farouk and R. Shamma, *Arch. Pharm.*, 2019, **352**(2), e1800213.
26. A. Yosipof, R. C. Guedes and A. T. García-Sosa, *Front. Chem.*, 2018, 6.
27. V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan and B. P. Feuston, *J. Chem. Inf. Comp. Sci.*, 2003, **43**(6), 1947.
28. A. Karatzoglou, D. Meyer and K. Hornik, *J. Stat. Softw.*, 2006, **15**(9), DOI: 10.18637/jss.v015.i09.
29. M. Rask-Andersen, S. Masuram and H. B. Schiöth, *Annu. Rev. Pharmacol. Toxicol.*, 2014, **54**(1), 9–26.
30. A. Lavecchia, *Drug Discovery Today*, 2015, **20**(3), 318–331.
31. G. Kandoi, M. L. Acencio and N. Lemke, *Front. Physiol.*, 2015, **6**, DOI: 10.3389/fphys.2015.00366.
32. G. Hessler and K.-H. Baringhaus, *Molecules*, 2018, **23**(10), 2520.
33. Q. Hu, M. Feng, L. Lai and J. Pei, *Front. Genet.*, 2018, **9**, DOI: 10.3389/fgene.2018.00585.
34. A. A. Jamali, R. Ferdousi, S. Razzaghi, J. Li, R. Safdari and E. Ebrahimie, *Drug Discovery Today*, 2016, **21**(5), 718–724.
35. J. Jeon, S. Nim, J. Teyra, A. Datti, J. L. Wrana and S. S. Sidhu, *et al.Genome Med.*, 2014, **6**(7), 57.
36. A. Mayr, G. Klambauer, T. Unterthiner, M. Steijaert, J. K. Wegner and H. Ceulemans, *et al.Chem. Sci.*, 2018, **9**(24), 5441–5451.
37. Y.-C. Lo, S. E. Rensi, W. Torng and R. B. Altman, *Drug Discovery Today*, 2018, **23**(8), 1538.
38. H. Chen, O. Engkvist, Y. Wang, M. Olivecronaand and T. Blaschke, *Drug Discovery Today*, 2018, **23**(6), 1241.
39. L. Wang, L. Chen, Z. Liu, M. Zheng, Q. Gu and J. Xu, *PLoS One*, 2014, **9**(5), e95221.
40. L. Guan, H. Yang, Y. Cai, L. Sun, P. Di and W. Li, *et al.MedChemComm*, 2019, **10**(1), 148.
41. B. Shaik, R. Gupta, B. Louis and V. K. Agrawal, *J. Pharm. Invest.*, 2015, **45**(5), 461–473.
42. O. Ursu, J. Holmes, J. Knockel, C. G. Bologa, J. J. Yang and S. L. Mathias, *et al.Nucleic Acids Res.*, 2017, **45**(D1), DOI: D932.10.1093/nar/gkw993.
43. P. Gkeka, S. Eleftheratos, A. Kolocouris and Z. Cournia, *J. Chem. Theory Comput.*, 2013, **9**(2), 1272.
44. I. L. Hudson, S. Y. Leemaqz, A. T. Neffe and A. D. Abell, in *Artificial Neural Network Modelling*, ed. S. Shanmuganathan and S. Samarasinghe, Springer International Publishing, AG Switzerland, 2016, p. 161.
45. I. L. Hudson, S. Y. Leemaqz, A. T. Neffe and A. D. Abell. in *Artificial Neural Network Modelling*, ed. S. Shanmuganathan and S. Samarasinghe, Springer International Publishing, AG Switzerland, 2016, pp. 161–212.
46. S. Lee and G. J. McLachlan, *Stat. Comput.*, 2012, **24**(2), 181.
47. I. L. Hudson, S. Shafi and A. D. Abell, Paper presented at the Sixth Annual ASEARC Conference, University of Wollongong, Wollongong, Australia, February 2014.
48. S. Mignani, J. Rodrigues, H. Tomas, R. Jalal, P. P. Singh and J.-P. Majoral, *et al.Drug Discovery Today*, 2018, **23**(3), 605.
49. D. F. Veber, S. R. Johnson, H.-Y. Cheng, B. R. Smith, K. W. Ward and K. D. Kopple, *J. Med. Chem.*, 2002, **45**(12), 2615.
50. T. I. Oprea, *J. Comput.-Aided Mol. Des.*, 2000, **14**(3), 251.
51. T. I. Oprea, J. Gottfries, V. Sherbukhin, P. Svensson and T. C. Kühler, *J. Mol. Graphics Modell.*, 2000, **18**(4–5), 512.
52. T. I. Oprea and J. Gottfries, *J. Comb. Chem.*, 2001, **3**(2), 157.
53. J. Wang and T. Hou, *Adv. Drug Delivery Rev*, 2015, **86**(11), 16.
54. S. Y. Leemaqz, I. L. Hudson and A. D. Abell. Paper presented at the 11th International Conference of the ERCIM WG on Computational and Methodological Statistics (CMStatistics 2018), University of Pisa, Italy, 14–16 December 2018.
55. C. Fraley, A. E. Raftery, B. Murphy and L. Scrucca, Mclust version 4 for R: Normal mixture modeling for model-based clustering, classification, and density estimation, University of Washington, https://www.stat.washington.edu/sites/default/files/files/reports/2012/tr597.pdf, Accessed 1 Dec 2019.
56. V. Law, C. Knox, Y. Djoumbou, T. Jewison, A. C. Guo and Y. Liu, *Nucleic Acids Res.*, 2014, **42**(D1), DOI: D1091.10.1093/nar/gkt1068.

57. D. I. Osolodkin, E. V. Radchenko, A. A. Orlov, A. E. Voronkov, V. A. Palyulin and N. S. Zefirov, *Expert Opin. Drug Discovery*, 2015, **10**(9), 959.
58. H. Abdi, L. J. Williams and D. Valentin, *Wiley Interdisciplinary Reviews: Computational Statistics*, 2013, **5**(2), 149.
59. I. Jolliffe, in *International Encyclopedia of Statistical Science*, ed. M. Lovric, Springer Publishing, Berlin, 2011, p. 1094.
60. M. C. Sachs, *Generate ROC curve charts for print and interactive use*, https://cran.r-project.org/web/packages/plotROC/vignettes/examples.html, Accessed 1 Dec 2019.
61. F. Husson, J. Josse, S. Le and J. Mazet, *Multivariate exploratory data analysis and data mining. Package 'FactoMineR'*, https://cran.r-project.org/web/packages/FactoMineR/FactoMineR.pdf, Accessed 1 Dec 2019.
62. M. Kuhn, *Classification and regression training. Package 'caret'*, https://cran.r-project.org/web/packages/caret/caret.pdf, Accessed 1 Dec 2019.
63. M. W. Beck, *Visualisation and Analysis Tools for Neural Networks. Package 'NeuralNetTools'*, https://cran.r-project.org/web/packages/NeuralNetTools/NeuralNetTools.pdf, Accessed 1 Dec 2019.
64. X. Robin, N. Turck, A. Hainard, N. Tiberti, F. Lisacek and J.-C. Sanchez, *et al.BMC Bioinf.*, 2011, **12**(1), DOI: 10.1186/1471-2105-12-77.
65. F. Giordanetto and J. Kihlberg, *J. Med. Chem.*, 2014, **57**(2), 278.
66. B. C. Doak, B. Over, F. Giordanetto and J. Kihlberg, *Chem. Biol.*, 2014, **21**(9), 1115.
67. M. D. Shultz, *J. Med. Chem.*, 2019, **62**(4), 1701.
68. D. S. Karlov, S. Sosnin, I. V. Tetko and M. V. Fedorov, *RSC Adv.*, 2019, **9**(9), 5151.
69. J. J. Naveja and J. L. Medina-Franco, *Front. Chem.*, 2019, **7**, 510.
70. K. R. Jinuraj, M. Rakhila, M. Dhanalakshmi, R. Sajeev, G. Akshata and K. Jayan, *et al.BMC Res. Notes*, 2018, **11**(1), 463.
71. A. K. Ghose, V. N. Viswanadhan and J. J. Wendoloski, *J. Comb. Chem.*, 1999, **1**(1), 55.
72. M. Congreve, R. Carr, C. Murray and H. Jhoti, *Drug Discovery Today*, 2003, **8**(19), 876.
73. F. W. Goldberg, J. G. Kettle, T. Kogej, M. W. Perry and N. P. Tomkinson, *Drug Discovery Today*, 2015, **20**(1), 11.
74. T. W. Johnson, K. R. Dress and M. Edwards, *Bioorg. Med. Chem. Lett*, 2009, **19**, 5560.
75. D. Lewis, P. Winner, J. Saper, S. Ness, E. Polverejan and S. Wang, *et al.Pediatrics*, 2009, **123**(3), 924.
76. T. T. Wager, X. Hou, P. R. Verhoest and A. Villalobos, *ACS Chem. Neurosci.*, 2016, **7**(6), 767–775.
77. M. Nayal and B. Honig, *Proteins: Struct., Funct., Bioinf.*, 2006, **63**(4), 892.
78. Z.-C. Li, W.-Q. Zhong, Z.-Q. Liu, M.-H. Huang, Y. Xie and Z. Dai, *et al.Anal. Chim. Acta*, 2015, **871**, 18.
79. P. J. Hajduk, J. R. Huth and S. W. Fesik, *J. Med. Chem.*, 2005, **48**(7), 2518.
80. M. Zhu, L. Gao, X. Li, Z. Liu, C. Xu and Y. Yan, *et al.J. Drug Targeting*, 2009, **17**(7), 524.
81. A. Volkamer, D. Kuhn, T. Grombacher, F. Rippmann and M. Rarey, *J. Chem. Inf. Model.*, 2012, **52**(2), 360.
82. P. R. Costa, M. L. Acencio and N. Lemke, *BMC Genomics*, 2010, **11**(Suppl 5), S9.
83. M. A. Yıldırım, K.-I. Goh, M. E. Cusick, A.-L. Barabási and M. Vidal, *Nat. Biotechnol.*, 2007, **25**(10), 1119.
84. Y. Ofran, G. W. Tang and R. B. Altman, *PLoS Comput. Biol.*, 2014, **10**(4), e1003589.
85. S. Prasanthi, S. D. Bhavani, T. S. Rani and S. B. Raju, in *Bioinformatics: Concepts, Methodologies, Tools, and Applications*, IGI Global, Hershey, PA, USA, 2013, p. 937.
86. I. L. Hudson, S. Shafi, S. Lee, S. A. Hudson and A. D. Abell, Paper presented at the *Aust Stats Conference*, ASC, Adelaide, Australia, July 2012.
87. P. Chatterjee, S. Basuand and M. Nasipuri, Paper presented at the Proceedings of the International Symposium on Biocomputing, Calicut, Kerala, India, 2010.

CHAPTER 12

Artificial Intelligence Applied to the Prediction of Organic Materials

STEVEN BENNETT(0000-0001-6285-910X),[a] ANDREW TARZIA(0000-0001-8797-8666),[a] MARTIJN A. ZWIJNENBURG(0000-0001-5291-2130)[b] AND KIM E. JELFS(0000-0001-7683-7630)*[a]

[a] Department of Chemistry, Imperial College London, Molecular Sciences Research Hub, White City Campus, Wood Lane, London W12 0BZ, UK
[b] Department of Chemistry, University College London, 20 Gordon Street, London WC1H 0AJ, UK
*Email: k.jelfs@imperial.ac.uk

12.1 Introduction

Functional organic materials are prevalent in many fields, including electronics,[1] porous materials,[2] polymers,[3,4] and ionic liquids.[5] We define the scope of organic materials here as materials formed from small or polymeric molecules constituted of only light elements. We do not include pharmaceutical drugs and biological materials as these are covered elsewhere, although we will note similarities in the problems faced and the approaches that can be used to tackle them. The discovery of new materials has previously been dominated by serendipity, but there is now a clear need to accelerate their discovery to help tackle the problems facing humanity, such as climate change and the depletion of the Earth's natural resources.

The focus of this chapter is the application of artificial intelligence (AI) techniques to accelerate the discovery of new organic materials by assisting in the prediction of their structure and properties and guiding their optimisation for target applications (see Figure 12.1).

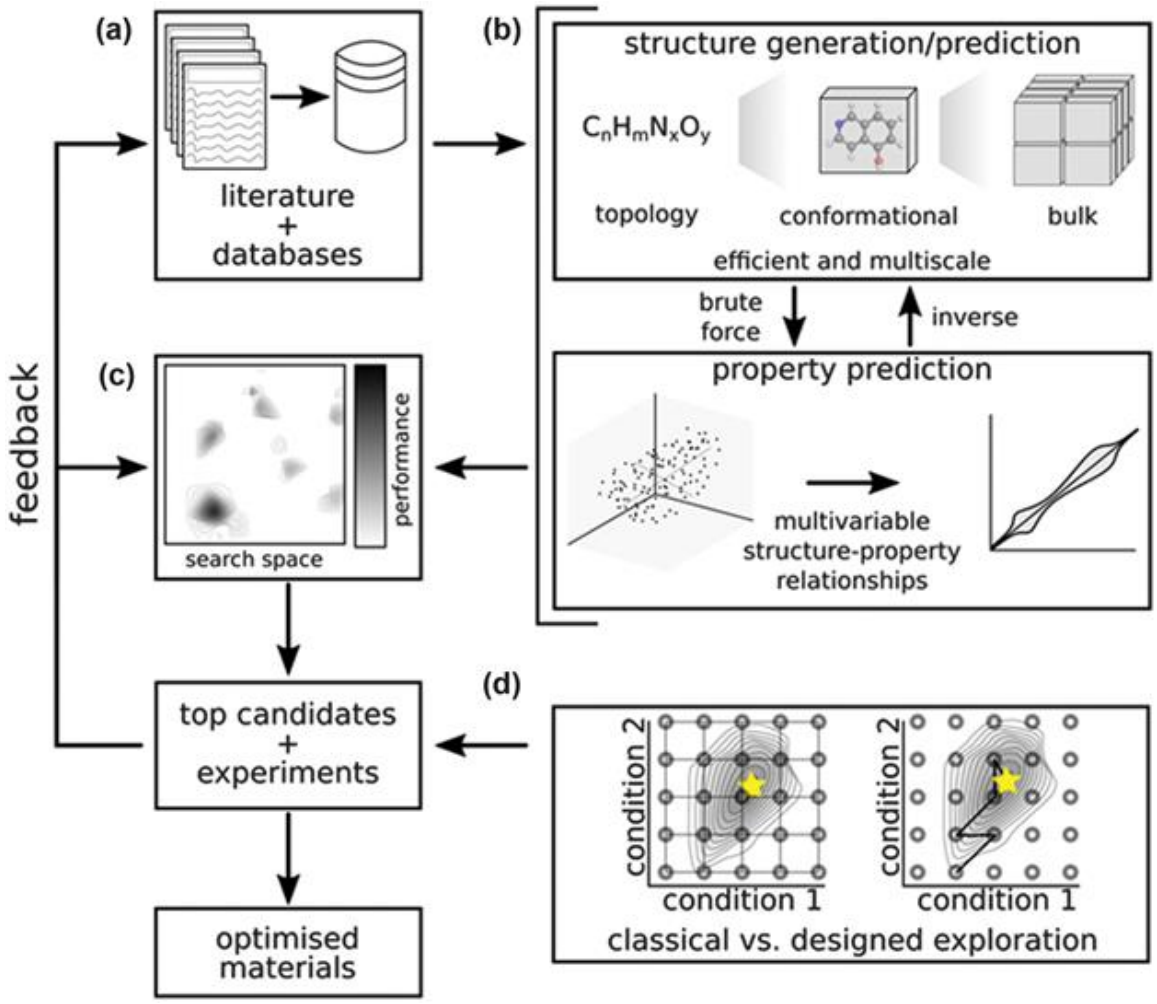


Figure 12.1 A schematic summary of the materials discovery workflow in the age of AI, which is driven by the data from (a) curated databases and the scientific literature. From the available data, (b) complex and multivariable structure–property relationships can be inferred using AI. Here, brute force structure generation and/or prediction may feed into property predictors or inverse design may be applied, which predicts top candidates for a target property prior to structure prediction. The (c) resultant search space can then be explored experimentally using AI-designed experiments (d) to efficiently explore phase space, leading to optimised materials, as well as improvements to material databases, search spaces, and the preceding AI models through feedback mechanisms.

Organic materials often offer advantages over their inorganic counterparts in their improved sustainability, increased processability, reduced toxicity of the final material, and often lower cost, all of which are crucial to the scale-up of material production. Furthermore, organic compounds are relatively easy to modify through a vast array of robust synthetic routes, allowing for the fine-tuning of material properties. For example, in the development of semiconductors and other electronics, there is currently a push to replace inorganic, silicon-based materials with organic materials.[6] From a modelling perspective, one major difference between organic materials and their inorganic counterparts is that they are normally a combination of separate molecules, be it small molecules or polymers, rather than a single entity. As such, one can in principle represent an organic material either as an isolated molecule, ignoring, perhaps beyond a mean-field approximation, the effects of the environment of the molecule, such as intermolecular packing in the case of solids, the solvent in the case of dissolved molecules, or even interfaces, or treat the whole condensed system. There are many clear examples in the literature for systems where a 'molecular representation' is employed. However, for this representation to be accurate and effective, the material's properties need to be dominated by that of the individual molecules. For example, for porous molecular cage materials, because the individual molecule's structure defines the solid-state pore topology, the potential for molecular separations can be determined from the single molecule.[7] Similarly, for conjugated polymers that form amorphous solids, a comparison between DFT calculations on isolated polymer chains and experimental photoelectron spectroscopy on polymeric solids[8] suggests that isolated polymer chains might be a good representation of the solid for predicting their optoelectronic properties. Conversely, for polymers of intrinsic microporosity (PIMs),[9] where intermolecular voids arise from the inherent poor packing of the polymer chains, a molecular representation will be clearly inadequate for predicting the emergent porosity, and a 'condensed-phase representation' is required. The choice of representation is relevant to both the generation of training data and the kind of descriptors used for machine learning. As will be discussed in more detail below, for some properties, computational limitations mean the only (efficient) way of calculating them is inherently molecular, while

completely different types of descriptors are used for molecules and condensed phases.

The search space for new organic materials does not only include the chemical space of the constituent molecules, which is estimated to consist of 10^{60} molecules,[10] but must also consider different ways of assembling individual molecules into materials, their resultant bulk properties, and the effect of experimental variables such as synthetic protocol.[11] With previous knowledge in a field, an experimental chemist can decrease this search space significantly. However, an inherent bias towards small changes within 'known' regions of phase space is a hindrance to a thorough exploration of phase space for the optimal material for an application. Computational methods and algorithms offer cheaper and more complete ways of searching phase space for the optimal materials.[12] A crucial aspect of exploring phase space using computational methods is the ability to predict the material properties of interest for each candidate accurately. Here, it is common to replace the complexity of multiple length and timescales with simplified models that are defined by existing structure–property relationships.

Computational optimisation has been applied in the field of drug discovery for many years because clear structure–property relationships exist.[13] In the case of early computer-aided drug discovery, only the molecular structure, which was used to calculate simple molecular properties (for example Lipinski's rule of five),[14] was necessary to advance the drug discovery process. Similarly, in the field of crystalline, inorganic materials, the existence of well-defined structures simplifies the determination of structure–property relationships, because in many cases the relatively well-defined electronic structure of the crystalline material contains the required information about the properties of interest.[11,15] However, because weak non-covalent interactions mostly define the bulk structure of many organic materials, the link between structure and organic material properties is not always clear. Therefore, the development and understanding of structure–property relationships is a complex, multiscale and multivariable problem. Given sufficient data, AI can be applied to understand multiscale structure–property relationships and remove the need for approximations and simplifications, which significantly advances the ability to predict material properties. Such a capability will drive the

implementation of inverse design procedures, which determine the optimal candidate materials and efficiently explore the material search space based on the optimisation of material function, without the need to test a myriad of possible candidates.[16,17]

It is possible to apply AI approaches to all aspects of a modern materials discovery pipeline, from candidate generation and data extraction to analysis and property prediction, offering the opportunity to drastically speed up the discovery process without a significant loss of accuracy. Additionally, AI presents a pathway toward the automation of materials discovery, at a significantly faster rate and lower cost than human-driven discovery. A significant barrier in the application of AI approaches is the quantity and quality of existing data, the lack of negative data,[18] as well as the choice of representation of chemical and physical systems.[17]

12.2 Extracting Data from the Literature

From the perspective of using AI approaches to predict new materials and their properties, it is crucial to have a large quantity of high-quality, machine-readable data for training. Within the entirety of the scientific literature, there is a massive amount of experimental and theoretical data in the form of unstructured text, figures or tables in journal articles, technical documents, and patents.[19,20] The manual collection and curation of this data into a machine-readable database becomes intractable on a large-scale. Fortunately, the scientific literature is often written in a formulaic way, which allows for the application of text-mining algorithms to automatically extract data. However, the ambiguity and variability in many conventions in different scientific domains limit the universality of established dictionary and rule-based approaches to text mining.[19–21] The current trend for large-scale information extraction is the use of machine learning (ML)-based (or hybrid) text-mining workflows because they are much more flexible than hard-coded methods.[19,22–24]

Recently, Cole and co-workers developed ChemDataExtractor (CDE), open-source software for the automatic extraction of chemical information.[25] CDE implements supervised and unsupervised ML models

throughout the entire natural language processing (NLP) and named entity recognition (NER) workflow (see Figure 12.2). NER is used to find named entities within the text, which can include chemical species, processes, synthetic conditions, material properties, and characterisation methods, and can be performed using ML[23] and hybrid ML, dictionary, and rule-based approaches.[19] A significant challenge of text mining is reliably creating a unique identifier for each entity, *i.e.* matching synonymous terms, which is called entity normalisation. Weston *et al.* performed tokenisation using CDE, NER using a neural network (NN) and entity normalisation using word embeddings (a machine-learned map of each word onto a dense vector space based on their co-occurrences in text) on the materials science literature toward a structured and accessible database of materials.[22] Similarly, Tshitoyan *et al.* learned a word embedding using text from scientific abstracts and used this to uncover relationships between materials and their potential properties.[24] As a result of its inbuilt flexibility, CDE is useful in many subdomains. In the field of organic materials, Cole and co-workers applied the CDE to the discovery of optimal organic dye pairs for co-sensitised solar cells.[26] They text mined 9431 dye candidates into a database that linked chemical structure with optical adsorption peak and molar extinction coefficient. From this initial database, a high-throughput screening workflow was applied to identify suitable candidate dye pairs and improved power conversion efficiency of the resultant top candidates was experimentally validated.

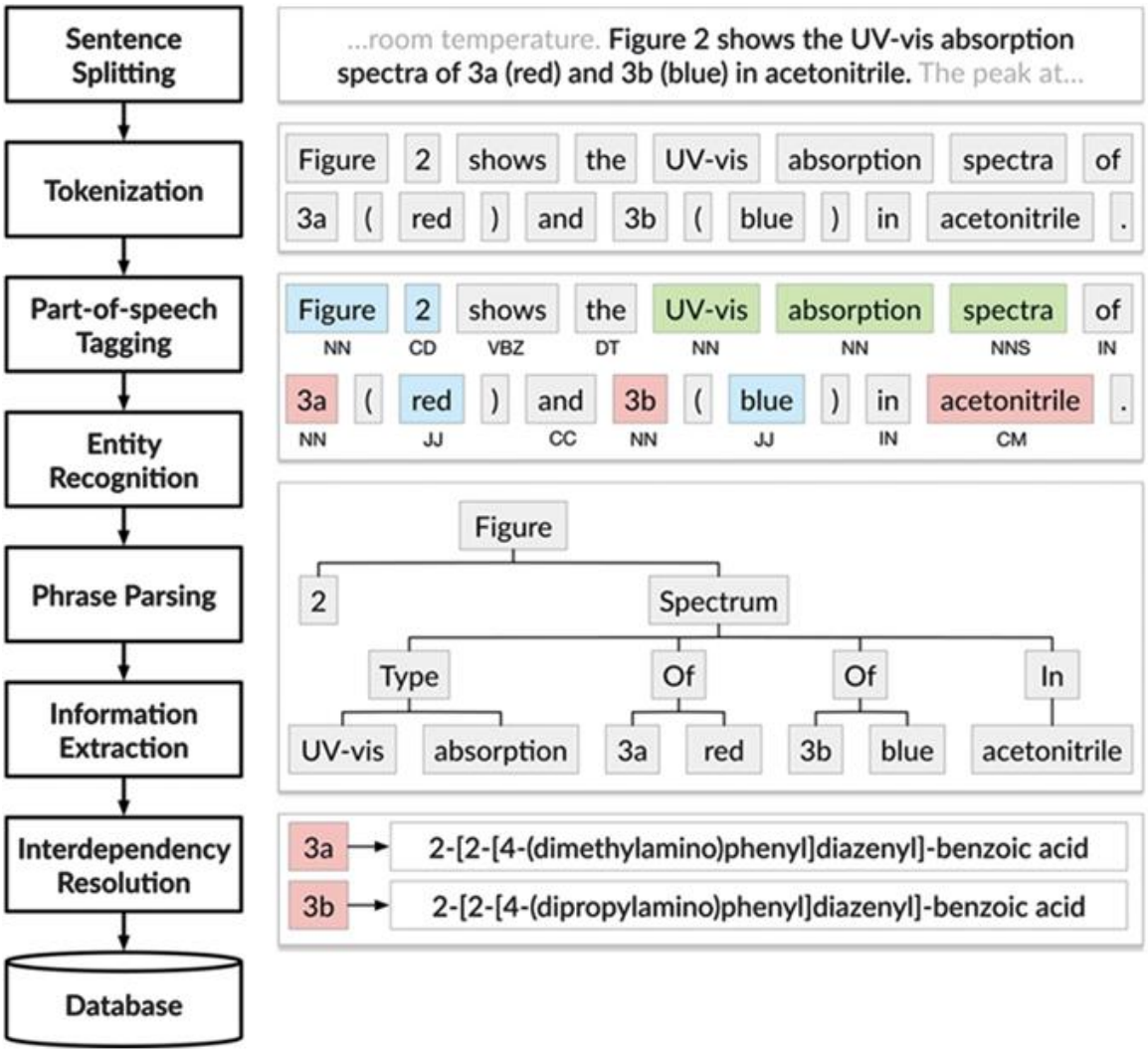


Figure 12.2 Natural language processing pipeline. Text is split into sentences and then into individual tokens. The part-of-speech tagger and entity recogniser outputs are combined to assign a single tag to each token, which is then parsed using a rule-based grammar to produce a tree structure. This tree structure is interpreted to extract individual chemical records for this sentence, which are then combined with records from throughout the document to resolve data interdependencies and produce unified records for depositing in a database. Tags shown: NN=noun, CD=cardinal number, VBZ=verb (third person singular present), DT=determiner, NNS=noun plural, IN=preposition, JJ=adjective, CC=coordinating conjunction, CM=chemical mention. Reproduced from ref. 25 with permission from American Chemical Society, Copyright 2016.

12.3 Building Block Libraries

Developing diverse precursor libraries is an important part of the materials screening process,[27] as it defines the chemical space available for examination. While molecular structure libraries such as ZINC,[28] GBD,[29] and PubChem[30] offer a large amount of diverse, synthesisable molecules, they are unfocused, making an exhaustive search of all precursor candidates infeasible, and many molecules in them are not designed with materials in mind. Enumeration of existing molecular libraries, through adding known useful functionalities or systematic processes, is a common method of extending building block libraries. However, hard-coded transformations often limit the chemical space spanned by precursor libraries,[27] potentially missing compounds that occupy favourable regions of property space. As part of the Harvard Clean Energy Project, a precursor library was created by enumerating compounds based on combining 26 molecular building blocks using pre-defined chemical transformations, to generate a library of over 1 000 000 molecules.[31] Additionally, Gómez-Bombarelli *et al.* designed a library of 1.6 million candidate materials for light-emitting diodes based on donor-bridge-acceptor molecules, systematically incorporating additional functional groups into the library according to chemical intuition.[32]

12.4 Predicting Properties

12.4.1 Training Data

Any application of ML requires training data for the model, the quantity and quality of which directly influence the types of algorithms that can be used and the likely accuracy of the trained model. Regression, *e.g.* Gaussian process and kernel ridge regression (KRR), and decision tree models can be trained with relatively small training datasets, but neural networks require much more data. Both algorithms offer advantages; however, deep neural networks typically outperform traditional ML techniques in property prediction.[33] The training data itself can be obtained from experiment or from theory and computation. In the context of property prediction, experimental data-sets are typically rather limited in size; for example, Kim *et al.* trained models to predict the glass temperature and density of

polymers using 451 and 173 experimental data points, respectively.[34] Hence experimental training data is often used in combination with regression and decision tree methods. Computational chemistry derived training datasets can be much larger, up to millions of data points, and can be obtained from a range of methods, including semi-empirical, density functional theory (DFT) and quantum chemistry calculations. DFT calculations are the norm, but modern semi-empirical methods, *e.g.* the xTB family of semi-empirical methods developed by the Grimme group,[35,36] can give results with a similar accuracy for a fraction of the computer time and hence allow for the generation of (much) larger training data-sets than feasible with DFT.[8,37] Conversely, if one wants to train an ML model to predict data with similar accuracy to quantum chemistry calculations, then one must train it with data from quantum chemistry calculations, which is possible, but time-consuming.[38–40]

Despite the ongoing progress in database creation for organic materials, there still remains an absence of data compared to other domains, such as in the fields of drug design or inorganic materials, in which automated calculation workflows have fuelled the growth of several inorganic material property databases, such as NOMAD[41] and the Materials Project database.[42] In the organic materials domain, there are several data repositories, focussing on specific material applications (see Table 12.1). Many datasets contain computational calculation results such as Khazana,[43] the Harvard Organic Photovoltaic Dataset,[44] the Open Spectral Dataset,[45] the dye-sensitised solar cell database[46] and the Organic Materials Database.[47] From these repositories, new material design principles have been established using both ML[48] and data-mining techniques.[49] Olsthoorn *et al.* used the smooth overlap of atomic positions (SOAP) kernel method of crystal structure featurisation to predict the band gap using KRR on data from the Organic Materials database.[50] The predictive model was then used to screen the Crystallographic Open Database, containing 30 713 organic crystal structures, and identified 15 candidates that were predicted to have a suitable band gap for solar cells.

Table 12.1 Examples of databases with potential for organic materials discovery, containing the properties stored and the number of

systems present within each database.

Name	Properties or Structures	Size
Khazana[69]	Atomisation energies Band gaps Dielectric constants	1073 molecules
Materials Project Nanoporous Database[153]	Largest cavity diameter Crystal density Isotherm data	85 000 networks
PoLyInfo[154]	Physical, optical, electrical, thermal, mechanical properties	44 280 molecules
Harvard Organic Photovoltaic Dataset[44]	HOMO/LUMO energies Band gap Power conversion efficiency Open circuit potential Short circuit current density	350 molecules
Dye-sensitised solar cell[46]	Open circuit potential Short circuit current density Fill factor Power conversion efficiency	4000 molecules
Organic Materials Database[47]	Crystal structures Electronic band structures	31 413 networks

Owing to the absence of large datasets, transfer learning has become an increasingly popular option for model training. In transfer learning, a deep neural network updates its initial weights from a large dataset from a similar domain to the target problem. A smaller, more accurate dataset is then used to fine-tune the weights of the model, learning subtle differences in the data from the new dataset. Paul *et al.* applied a similar methodology to predicting the HOMO of potential organic solar cell candidates.[51] The Harvard Clean Energy Project database was used to provide the properties of 2.3 million candidate donor molecules as training data for a convolutional neural network. Network weights were adapted using a combination of 243 experimental values and 344 values obtained at DFT accuracy, reducing the mean absolute error of the prediction by around half. However, as ML algorithms only interpolate from the provided dataset, the

model's accuracy is limited to that attained by the property calculation method used.

12.4.2 Descriptors

When identifying new structure–property relationships using AI, the chemical characteristics of the material must be captured in a numerical (often vector) representation. An ideal descriptor for a material would contain sufficient information for the model to identify chemical insight from the vector representation, without containing unnecessary information. Descriptors are usually engineered based on the type of algorithm being used and the property being targeted for prediction.[52] For example, electronic properties, originating from the three-dimensional conformational configuration of the molecule, are often poorly described using descriptors based on the underlying two-dimensional graph representation.[53] Descriptors are not limited to encoding structural features of the molecule; they can include experimental measures, such as solubility, dipole moments or polarizability (see Figure 12.3). Thousands of molecular descriptors can be calculated using the DRAGON software, which is frequently used in QSAR modelling.[54,55]

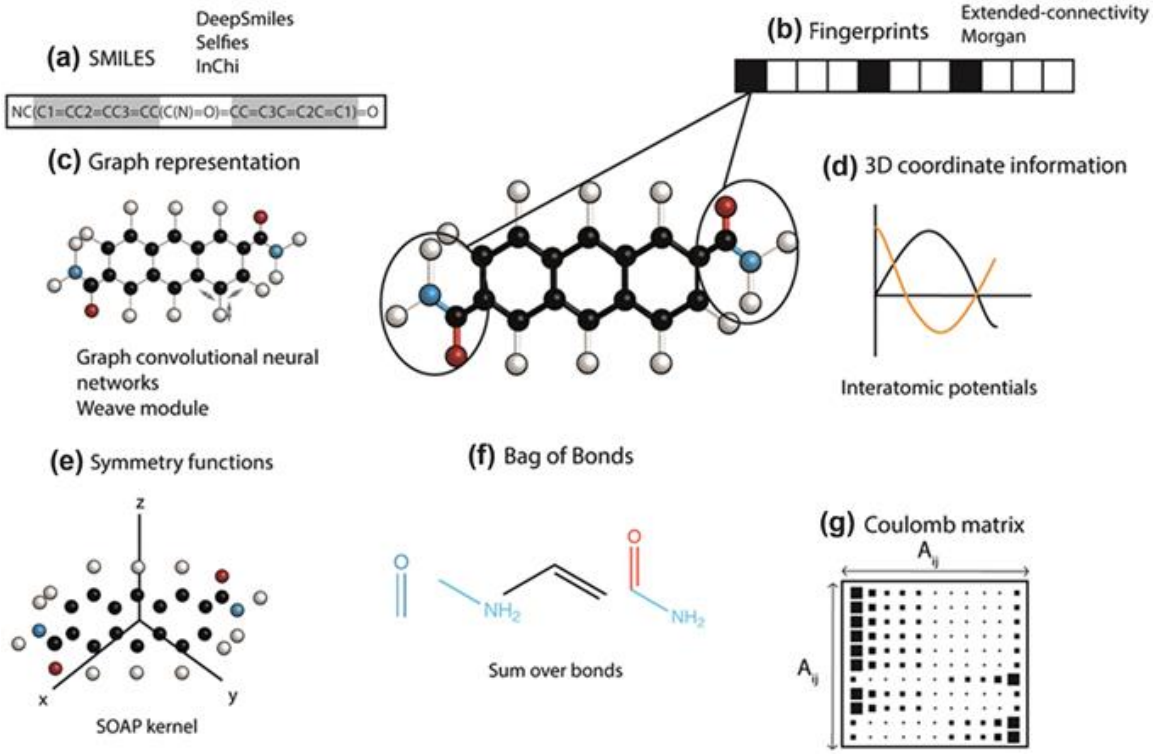


Figure 12.3 A summary of commonly used descriptors in machine learning for organic materials design. Single molecules can be represented using (a) SMILES string text encodings; (b) molecular fingerprints encoding structural fragments present in a molecule; and (c) graph representations, which are a machine-learned representation of atomic environments in relation to their neighbours. Full atomic environments can be described using descriptors based on (d) 3D geometry, such as modelling interatomic interactions using potential energies; (e) SOAP kernels or symmetry functions; (f) the bag-of-bonds representation; and (g) Coulomb matrices.

Many structural descriptors encode the molecular representation, relying on graph-based bonding representations of the molecule as an input vector. Bonding information can be captured in the form of adjacency matrices; however, bonding information alone is unlikely to contain sufficient chemical information to predict properties. Fingerprinting algorithms, such as Extended-Connectivity or Morgan fingerprints,[56] encode a hash of different substructures present in a molecule, facilitating property prediction based on fragments present in the single molecule. Descriptors based on the two-dimensional structure of a molecule do not contain information relating to the broader environment of that molecule in the solid state (*i.e.* in crystalline or amorphous structures). Therefore, there is significant interest

in developing new descriptors to predict properties that arise as a result of molecular packing. Descriptors used to model the solid-state often contain geometric information about atomic environments; however, it is a significant challenge to ensure that they are invariant with respect to rotation, translations or reordering of the atoms in the 3D structure. Coulomb matrices,[57] containing the nuclear charge and position of atoms, have proven to be effective at predicting crystal structure properties, despite suffering from issues with describing periodicity.[58] The Bag of Bonds descriptor, based on bags representing bond types present in the molecule, and their geometric distances, was used to predict total and atomisation energies.[59] Elton *et al.* found the descriptor calculated from the sum over bonds performed best when calculating density, heat of formation, shock velocity, and explosive energy, when using KRR.[60]

The SOAP kernel function has been used as a descriptor for properties of molecules in the solid state.[61] This kernel function is a measure of dissimilarity between atoms in an environment, enabling three-dimensional atomic arrangements to be distinguished. To calculate the SOAP descriptor, atomic environments are represented by Gaussian functions placed on atomic positions. Alternatively, symmetry functions are useful in converting the higher-dimensional Cartesian coordinate representation into a lower-dimensional space.[62] These representations were then used to train an NN; however, SOAP performed similarly well at a reduced computational cost.[63] These types of descriptors have been used to learn atomic potentials, with no underlying physical assumptions, enabling simulations of a significantly larger number of atoms than can be achieved using *ab initio* methods.[64]

More recently, deep learning (DL) algorithms have been used to generate novel representations of underlying chemical structures, relying solely on a graph representation of a molecule.[65] Using automatic feature extraction, DL algorithms are able to learn their own representations from the graph of the molecular structure, assuming no domain knowledge about the material. Convolutional neural networks (CNNs) can learn novel representations based on the graph of a molecule, leading to an increase in the predictive performance and interpretability of the trained ML model.[65] Graph CNNs are not limited to a single-molecule graph representation and have been

applied to the featurisation of crystal structures using the positions of atoms in the unit cell.[66,67] Wu *et al.* describe a benchmarking test, MoleculeNet, for both learned and fixed molecular descriptors based on their ability to predict QM energies and physical, biophysical and physiological properties of molecules from benchmarking datasets.[68] Learned graph representations were shown to perform better on nearly all datasets because the learned representations took into account spatial details of the atomic environment. However, this approach requires significantly more data than conventional ML methods. In materials design, learned representations could potentially establish novel insights that are not captured using traditional descriptors, but the absence of large datasets is a significant drawback to achieving greater predictive power using these methods.

12.4.3 Accelerated Property Prediction

AI approaches, in particular (supervised) ML offer the opportunity for vastly accelerated property prediction, being many orders of magnitude faster than, for example, DFT calculations, taking property prediction from hours or days to seconds. This sort of forward problem is fairly simple compared with inverse design. While the properties of some organic materials, in particular porous materials, may rely only on geometric features that are computationally cheap to calculate, or properties such as diffusion that can be simulated with molecular dynamics, the vast majority of organic materials have properties that depend on an accurate quantum mechanical description of the material. Performing *ab initio* calculations to calculate properties is computationally demanding and thus inherently limiting to the number of materials that can be studied. Furthermore, there is the potential for missed opportunities, where a material is only tested for a particular property in mind – for example, power conversion efficiency – when it could have excellent ferroelectric properties, but that property is not considered, and even if it was it would require a different set of specialist calculations.

Property prediction acceleration has already been demonstrated for many properties of isolated organic molecules, such as thermochemistry, electronic properties, atomisation energies, electron density, and solvation energies.[17,43,69,70] The next step of modelling the properties of solid-state

materials is obviously more challenging, but, later in this chapter, case studies across the range of organic materials will describe successes so far and the future potential in this area. As in other areas, key challenges are to have accessible databases of high-quality data in sufficient quantity for supervised ML; or, alternatively, to have algorithms that are effective with smaller quantities of data. Ideally, one might like experimental data on the properties of interest to more accurately reflect the full complexity of features that can influence performance, but thus far the majority of studies have focused on the rapid prediction of properties based on training data from simulations. A future goal would be to not just predict material performance, but to predict the device-level performance, taking into account a multi-component system with macroscopic influences on properties.

12.5 Structure Prediction

As currently there is often no route to do a forward prediction directly from molecular structure or precursor to bulk material properties, to make predictions it is first necessary to predict the solid-state structure of the material, whether that be an amorphous or crystalline structure. It is also a valuable endeavour to assist in the identification of synthesised materials and to explore and mine structure–property relationships. Structure prediction is a large field in its own right, with potential for AI to accelerate the process, as has already been demonstrated for several elements of it. Crystal structure prediction (CSP) for organic molecules has a long history, in large part due to the interest of pharmaceutical companies in understanding and predicting polymorphs of active pharmaceutical ingredients.[71,72] These approaches have also been applied to other organic materials, in particular molecular electronics.[73,74] The typical key steps of the prediction (as outlined in Figure 12.4), are to first construct an accurate energetic description of the system, for instance by parameterising a tailor-made forcefield, then to conduct a global optimisation search, which requires an accurate relative energy ranking of the hypothetical polymorphs, on the principle that low energy polymorphs are the most likely to be

experimentally observed. Further analysis of the packing motifs or properties of the polymorphs can then be conducted. Regular blind tests have demonstrated the increasing ability of the approaches to deal with more complex systems, such as those including molecular flexibility, co-crystals, and hydrates, with the ability to use *ab initio* calculations for energy ranking proving particularly effective.[75]

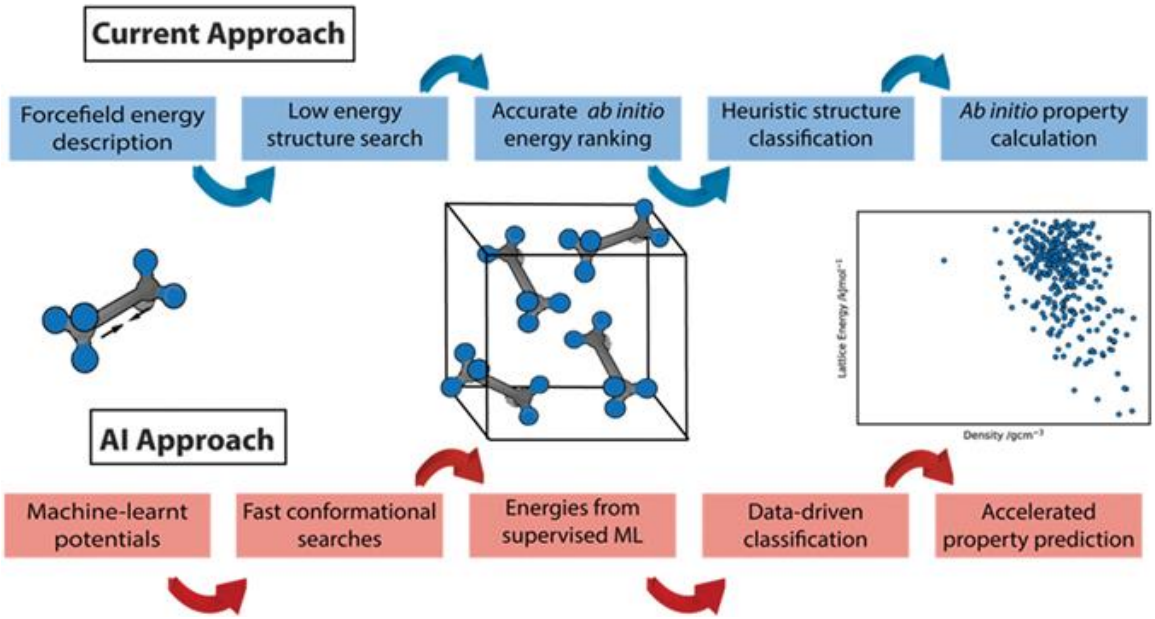


Figure 12.4 The key steps in crystal structure prediction (top) and where AI techniques can improve the process (bottom).

While the typical current approach for CSP does not involve AI, there are multiple opportunities here (see Figure 12.4), in particular with the development of new algorithms, as demonstrated in several studies looking at improving individual elements of the process. Machine-learnt potentials (MLP)[76–78] offer an alternative to transferable or tailor-made forcefields, involving a more generalised approach to their fitting and the potential for higher accuracy. A variety of approaches are used for the search for low energy polymorphs, including genetic algorithms,[79] random and quasi-random searches.[80] Although not yet applied for molecular CSP, neural networks are effective for enhanced sampling or learning of free energy landscapes.[81,82] As the relative energetic rankings of polymorphs are

extremely sensitive to small differences in intermolecular interactions, the accurate energetic ranking of polymorphs is a significant challenge. Although the inclusion of final energy rankings using DFT calculations with dispersion corrections is valuable, these calculations are currently prohibitively expensive on all but a few structures; hence forcefield calculations are used in the earlier stage of the structure search. Here, improvements in the calculation of accurate free energies using ML can massively accelerate the calculation of accurate energies.[77] McDonagh *et al.* have demonstrated that forcefield energies for CSP can be improved by using ML to correct the two-body terms with second-order perturbation theory, yielding higher accuracy at a five-fold reduced cost.[83] The challenge would be to extend this to more flexible molecules, rather than the rigid molecules in this study.

Musil *et al.* applied a Gaussian Process Regression (GPR) based upon the SOAP-REMatch kernel to assess the structural similarity of the crystal packings and then accurately predict the relative energies of packings of pentacene and two azapentacenes.[73] For the simplest structure, pentacene, only 50 data points were needed for accuracy below 1 kJ mol^{-1}, but for the azapentacenes, which had more complicated intermolecular interactions, approximately 1000 data points were needed for the equivalent level of accuracy. Next, Musil *et al.* used SOAP-REMatch to perform unsupervised learning to classify the thousands of crystal structures to assess the diversity of packing motifs, replacing the need for heuristic rules.[73] Yang *et al.* later showed that this packing exploration was possible for a single multi-landscape analysis of 28 different molecular organic semiconductors, although this did require a manual selection of the parameters for the kernel.[84] Finally, the benefit of accelerated property calculation through supervised ML is obviously of great use for property calculation across structure-energy-property landscapes of thousands of structures.

In addition to structure prediction to identify promising hypothetical materials for synthesis, structure prediction can also facilitate faster structure identification, which is a potential bottleneck in high throughput automation studies. If low energy hypothetical crystal structures have been calculated, then their powder diffraction patterns can be simulated, but this is often insufficient for conclusive structure identification without

significant further effort. The calculation of NMR shifts has proven to be an effective route to structure identification,[85] but this requires solid-state *ab initio* calculations, so it is certainly not feasible for routine use. However, recently Paruzzo *et al.* have demonstrated that ML can predict chemical shifts from DFT training data to DFT-level accuracy and thus identify molecular solids.[86] One could also envisage ML being used to accelerate analysis from a variety of other data, including imaging.[87]

Not all organic materials adopt crystalline packings in the solid-state and instead form a molecular or polymeric amorphous packing. Approaches to creating simulated models of these systems typically involve the generation of random packings, followed by some sort of annealing procedure. The most obvious application of ML here is in the generation of MLPs that can not only accelerate the molecular dynamics simulations involved in the annealing, but also allow larger, more realistic models to be simulated. This has already been demonstrated for inorganic materials, for example for amorphous silicon.[88] Although we are still a long way from being able to rapidly predict the assembly of a molecule from only its molecular structure, there is a lot of promise in the acceleration of parts of the process of structure prediction. The next challenge will be to make these approaches more generalisable and transferable between systems. Of course, along with better crystal representations and algorithms, more training data on molecules and their low energy polymorphs would be required to try to make a significant leap forward towards a more universal structure prediction of organic materials.

12.6 Exploration of Phase Space

Given the vast potential possibilities for molecular building blocks of organic materials, and the fact that only a limited number of these have ever been synthesised, a significant challenge is how to effectively sample that chemical space to find the most promising materials. All too often, computational studies have been limited by initial building block libraries that are extremely limited in size (of the order of 100), significantly restricting the scope of such studies. In the next section, we discuss how to

generate new molecules *de novo*, but for a given search space there are multiple possible options to try to effectively and efficiently sample the vast phase space of possible materials, or rather the building blocks they could be made from. A brute force approach can be used in cases of a small search space and a relatively cheap assessment method, or a random approach, often combined with intuitive rules included by a chemist, could be applied. Moving into AI approaches, genetic algorithms (GAs) are frequently used as efficient approaches to sample phase space and find promising candidates. Flavours of GAs can include, with a SMILES representation, graph-based representation (GB-GA) or Monte Carlo Tree Search (GA-MCTS).[89] More recently, ML approaches have also been applied, including variational autoencoders (VAE), deep neural networks (DNN), and recurrent neural networks (RNN). However, recent benchmarking of GA-based approaches compared to ML in the area of molecular design for drug discovery have suggested that GB-GA can outcompete RNNs that use SMILES representations, but RNNs trained on large datasets did outperform the GA in the compound quality.[90] This type of benchmarking has not yet been performed for organic materials design, but it should be borne in mind that an NN will not necessarily outperform other methods.

Of course, effective exploration of chemical space is a problem not just limited to computational studies, but is also of increasing importance for experimental studies for materials discovery or material optimisation, particularly with the increase of automation.[12,91,92] GAs, simulated annealing, and particle swarm optimisations have been used to guide discovery, find new systems, or find optimal conditions to enhance performance. Design of Experiments (DOE) is a statistical approach for multivariate analysis and is used to test and optimise several variables simultaneously, accelerating optimisation, time and resources. This is obviously valuable for optimisation of materials and devices where there is a high degree of complexity arising from many components and processing conditions. DOE combined with ML was found to be effective by Cao *et al.* for optimising power conversion efficiency (PCE) in an organic photovoltaic, where they used a support-vector machine learning approach to create phase maps based on sampled points.[93] RNNs will likely be

increasingly used to select the initial test sets, for automated analysis of outcomes on chemical robots, including classification of results, and the prediction and selection of the next tests to perform. The latter has been achieved for organic synthesis reactions, classifying “reactive” and “non-reactive” outcomes on a robot and predicting future reactivity.[94] This will be extended further to the more complex problems of organic materials in the future.

12.7 *De Novo* Molecular Design of Materials Building Blocks

There has been a significant interest in both ML and, most notably, DL algorithms for *de novo* molecular library design to assist in exploring larger regions of chemical space than that covered by known building blocks or small libraries. There is typically a trade-off between large chemical space exploration and creating new molecules that are chemically synthesisable. NNs possess the ability to learn novel representations of molecular input representations, transforming input features into machine-interpreted latent space, whilst DNNs can create even more complex representations as they contain a greater number of hidden layers. Many approaches rely on perturbing these learned representations, thereby sampling the machine-learnt chemical space (see Figure 12.5(a)).[16]

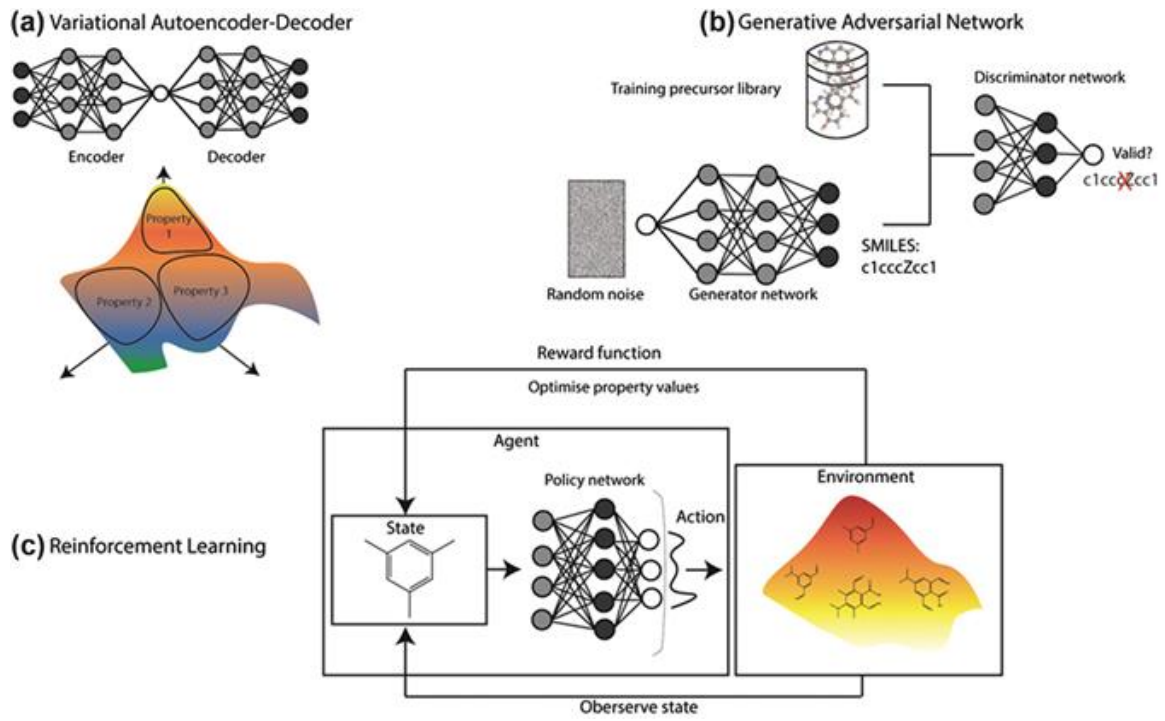


Figure 12.5 A summary of molecular *de novo* design techniques: (a) variational autoencoder-decoder, in which a neural network is trained to transform molecules into a latent representation; (b) generative adversarial networks, in which a generator neural network creates molecules, that are classified as real compounds using the discriminator network; and (c) reinforcement learning, where a neural network is trained to predict the best possible action to perform on a given state.

The ability of DNNs to transform human-interpretable molecular representations into phase space makes them suitable for molecular design. Gómez-Bombarelli *et al.* used a DNN as part of a VAE, to create a mapping from the graph-based SMILES representation of a molecule into latent space. Transformations were then applied to a molecule by perturbing the continuous vector representation of the molecule in latent space, which was converted back into SMILES using another DNN as the decoder.[16] Both an RNN and CNN were trialled as the encoder, with the CNN performing slightly better. An advantage of this approach is the ability to perform property predictions in latent space, potentially allowing for inverse design when sampling regions of this space.[27] Transfer learning has shown to be useful at generating molecules with desirable properties, by adapting a model that has been pre-trained on valid chemical SMILES from large

precursor datasets. This is followed by retraining the model to generate compounds with certain properties.[95] The ChemTS library uses a Monte Carlo tree search, combined with an RNN to dictate the probabilities of selecting each search path.[96] Sumita *et al.* used ChemTS to generate molecules with desirable excitation energies, combining the reward function of the molecular generator with time-dependent DFT calculations.[97] Excitation energies were calculated and fed into the generator model, resulting in the identification of five novel compounds with favourable oscillator strengths.

Reinforcement learning, which aims to learn the best course of action to take based on the prior experiences the model has seen, has been shown to bias the generative model towards creating molecules with favourable properties.[98,99] Popova *et al.* used the ReLeaSE framework to generate molecules with desired properties.[99] Consisting of both a generative and predictive NN, combined with reinforcement learning, the generative NN was able to propose molecules that maximised a single-optimisation reward function, which can be tuned to maximise a desired property. Despite the many types of generative algorithms that have been developed, many suffer from significant flaws. A large proportion of the GANs and RNNs often produce invalid SMILES strings, limiting their potential usage. While using a long short-term memory architecture in RNNs has been shown to diminish this issue,[27] algorithms that optimise towards a specific goal, often significantly reduce the diversity of the molecules, most notably in GANs.[100] The vast number of generative algorithms has resulted in many benchmarking metrics for generated molecules, including quantitative analyses of uniqueness, diversity, and novelty of molecules.[90] These standardised benchmarks have been primarily used to evaluate drug molecules space; however, these same measures could be used to evaluate generated molecules for materials use in the future.[101]

12.8 Predicting Synthetically Accessible Materials

While AI approaches have been shown to successfully identify materials with favourable properties, it is difficult to determine which of those materials are experimentally realisable. The synthetic accessibility (SA) of a material is an important part of the materials screening process, due to the time and costs associated with experimentally validating a potential new material. Several points need to be considered: (i) can the material's precursors be synthesised, (ii) can the material then be synthesised in the correct form using a modular approach in high yield and purity, and (iii) can the material be processed into the correct form to obtain the desired properties. When performing a high-throughput screening procedure, SA of the precursors can be incorporated into the methodology in one of two ways. Precursor libraries can be restricted to those that have previously been synthesised or do not require difficult synthetic procedures to create. Alternatively, the SA of a material can be defined as a numerical property and optimised during material optimisation. Significant challenges arise when calculating SA due to the vast number of factors that can affect it, and the assumptions inherent in its calculation. A molecule that requires several low yielding reaction steps, with several difficult separation processes involved and costly reagents, would be inherently difficult to synthesise. As new reactions are discovered, and chemical prices fluctuate, a material's SA can change with time. Early approaches aimed at computing the SA of an organic molecule focused on penalising molecules exhibiting more complex structural moieties.[102] However, medicinal chemists were shown to disagree when selecting a target molecule to easily synthesise, suggesting training data is not often reliable from expert chemists.[103]

ML has provided an alternative solution to predicting the SA of a molecule, by using a data-driven approach. Reaction data can provide an insight into the type of synthetic transformations that are possible, allowing a prediction to be made as to whether a molecule is synthesisable. Using an NN, reaction databases have been used to predict the SA of a molecule.[104] ML has also shown to perform well for retrosynthetic pathway prediction, allowing an understanding of the possible reaction pathways to create a molecule.[105,106] While the number of reaction steps is a factor in dictating the SA of a molecule, another important factor, cost, was shown to be linearly dependent on the number of reaction steps to synthesise an organic

photovoltaic molecule.[107] One challenging factor in this type of ML prediction is the absence of informative reaction data. Organic reaction databases are primarily composed of reactant, product pairs and do not contain yield, solvent or work-up information associated with the reaction. Furthermore, the absence of negative examples of failed reactions makes classification tasks difficult.

O'Boyle *et al.* used chemical intuition from previously synthesised molecules to create a library of precursors that were used to design polymers for organic photovoltaics.[108] Additionally, the symmetry of each monomer was considered when creating tetramers to increase the probability that each polymer could be synthesised at the expense of reducing the size of explored chemical space. Even if the covalent bonding in the organic material can be predicted using the techniques mentioned previously, it is still challenging to predict whether the molecule will form the desired three-dimensional conformation with desirable properties. For example, when considering porous organic cages, the topology and resulting porosity are sensitive to a diverse range of factors, including solvent effects, reaction conditions, and the structural characteristics of precursor molecules.[109] Therefore, each of these factors should be considered when determining if a porous organic cage is synthesisable.

12.9 Case Studies

12.9.1 Organic Electronics

In the field of organic electronics, the optoelectronic properties of small molecules, oligomers, and/or polymers are exploited to make solar cells, light emitting diodes, transistors, and thermoelectrics. The potential chemical space of relevant molecules is enormous; for example, even from a limited library of 500 monomers, one could construct up to 250 000 binary ordered co-polymers.[37] Therefore, virtual screening of molecules to find promising leads or to understand trends is impossible in the absence of machine learning models. Most, if not all, studies in this area make use of the single-molecule approximation in some way and focus either on training

and using ML models for predicting the properties of isolated molecules or on taking the properties of isolated molecules calculated using traditional quantum chemistry and using AI to predict the properties of the bulk or devices.

Application of AI in this area started essentially with the use of simple multivariable linear regression (though Sumpter and Noid already in 1994 showed that NNs could in principle be used to predict properties of polymers including electronic properties such as their dielectric constant, see Section 12.9.3 for more details).[110] Olivares-Amaya *et al.* used multivariable linear regression in combination with classical cheminformatics descriptors, such as molecular mass and logP, to predict the open circuit voltage (V_{oc}), short circuit current (J_{sc}), fill-factor (FF) and power conversion efficiency (PCE) of organic solar cells based on a combination of organic electron-donor polymers and the electron-acceptor fullerene PCBM,[111] while Jabeen *et al.* use a similar set-up to predict the refractive index of polymers.[112] Pyzer-Knapp *et al.* trained GPR models to correct the energy of the highest occupied molecular orbitals (HOMO) and lowest unoccupied molecular orbitals (LUMO) of organic electron-donors included in the HOPV15 data-set[44] as calculated with DFT, as well as the V_{oc}, J_{sc}, and PCE of solar cells based upon them, to reproduce the corresponding experimentally measured values.[113] Lopez *et al.* similarly used GPR models but focussed on small-molecule electron acceptors instead as an alternative to PCBM.[44] These GPR models take as input the property based on DFT, as well as the Tanimoto similarities between the Morgan fingerprints of the different molecules in the set. Kim *et al.* used GPR with a mixture of cheminformatics descriptors to predict properties, such as the band gap, refractive index, dielectric constant and glass temperature, for a mixture of conjugated polymers and non-conjugated polymers.[34] Mannodi-Kanakkithodi *et al.* also studied the dielectric properties of polymers but used KRR instead of GPR.[114] In contrast to most of these studies, Kim *et al.* trained their GPR to reproduce a mixture of experimental and DFT predicted properties.[34] Finally, Musil *et al.* used a combination of SOAP descriptors and GPR to accelerate the prediction of electronic transport properties of large numbers of polymorphs of a given organic semiconductor, exploiting the fact that different polymorphs are

very similar and that learning properties based on a sub-set of DFT calculations is relatively easy.[74]

GPR and KRR are mostly used to train models to experimental data, NNs, in contrast, are generally trained to computational data, *e.g.* from DFT. Pyzer-Knapp *et al.* as a proof of principle trained an NN using Morgan fingerprints on the DFT predicted HOMO and LUMO energies and PCE values of 200 000 random molecules from the Harvard Clean Energy Project database and used another 50 000 molecules as the test set.[115] Gómez-Bombarelli *et al.* trained an NN using Morgan fingerprints on the thermally assisted delayed fluorescence (TADF) rates, as calculated by TD-DFT, of 200 000 TADF candidate molecules as part of a workflow for discovering new leads for TADF.[32] Wilbraham *et al.* trained an NN using Morgan fingerprints to predict the ionisation potential, electron affinity and optical gap of 350 000 binary conjugated polymers using a training sub-set of only 50 000 polymers, 50% of which were used for the training of the NN and 50% for testing it.[37] Instead of using DFT derived-data, Wilbraham *et al.* generated their training data using the xTB family of semi-empirical, density functional tight-binding methods,[35,36] which often provides properties with similar accuracy to DFT but with computational cost reduced by at least three orders of magnitude.

All the NN studies above use standard Morgan fingerprints. In contrast, St. John *et al.* when training an NN for the optoelectronic properties of conjugated polymers used a message-passing NN for feature selection as input for their property prediction NN.[116] However, this does not necessarily seem to result in better predictions. They found that providing an NN with information about the 3D structure of conjugated polymers does not result in improved performance when training it to predict the optoelectronic properties of these polymers relative to only 2D structural information. This is in line with the observation by Wilbraham *et al.* that such properties generally vary only slightly between conformers of conjugated polymers.[8,37,117] While all of the above studies are for very large data-sets, the structures in them belong to a more limited structural family, for example, polymers based on aromatic monomers or TADF candidate molecules consisting of a donor and an acceptor separated by a spacer. Ramakrishnan *et al.* demonstrated that it is also possible to train an

NN that can reproduce the optical spectra of a much wider variety of organic molecules.[38] They trained NNs to recover the approximate coupled-cluster low-energy singlet excitation spectra of a very diverse set of 21 800 molecules with up to 8 CONF atoms, including both aromatic and aliphatic molecules. However, to do so, Ramakrishnan *et al.* used descriptors that contain 3D structural information (Coulomb matrix, bag-of-bonds) as input.

A completely different example of ML in the area of organic electronics is the recent study by Bai *et al.*, where gradient boosted trees were used to study the correlation between a number of predicted and measured properties of polymers and their activity as hydrogen evolution photocatalysts.[118,119] Use of these gradient boosted trees allowed the authors to estimate how much of the variation in the experimentally measured hydrogen evolution rates could be explained by these material properties, something that would be difficult using classical linear regression, as there is no good understanding of a model that links them. Finally, Sumita *et al.*[97] and Benjamin Sanchez-Lengeling *et al.*[120] used, as discussed above, molecular generation tools, ChemTS (a combination of GA-MCTS and RNN) and ORGANIC (GAN) respectively, to generate molecules with desired optical properties and non-fullerene acceptors that maximise the PCE of organic solar cells. Yuan *et al.* also used transfer learning combined with an RNN to generate donor–acceptor oligomers with targeted electronic properties.[101]

12.9.2 Property Prediction and Phase Space Exploration of Porous Organic Materials

In the last two decades, the field of porous materials has been revolutionised by multiple classes of materials, including metal–organic frameworks (MOFs),[121] covalent organic frameworks (COFs),[122] porous organic polymers,[123] and porous molecular materials (PMMs).[124] Through simple modifications of building blocks or synthetic conditions, the diverse chemical and structure space of porous materials can be explored toward fine-tuned materials for a target application. Currently, the most prominent field of porous materials is in the study of MOFs because of their (mostly) crystalline and extended nature (leading to well-defined structures), their structural and chemical diversity, and their interesting hybrid properties (a

result of the presence of inorganic and organic building blocks). The existence of large experimental (∼70 000 structures)[125,126] and hypothetical (∼150 000 structures)[127,128] databases of MOF structures simplifies the development of structure–property relationships for the performance of MOFs in many applications. Large databases of purely organic extended, porous materials (*e.g.* COFs),[129] and porous polymer networks (PPNs)[130] do exist. However, their structures are mostly hypothetical or idealised, because COFs and PPNs are commonly semi-crystalline or amorphous materials. Taking advantage of the existence of these structure databases, Thornton *et al.* used an NN to guide their exploration of the nanoporous materials genome (∼800 000 porous materials) toward hydrogen storage.[131] To avoid simulating the adsorption properties of all porous materials, they used an NN to learn which materials to simulate next after only simulating a small subset (1000 materials) of the entire database. After simulating only 3000 materials in total and training their NN three times (one time per 1000 simulations to guide the selection of the next 1000 materials to simulate), they determined the optimal range of material void fraction for hydrogen working capacity, which shows a strong trade-off effect.

Simplified descriptors of the pore networks of extended porous materials (including the solvent-accessible surface area, accessible pore volume, and pore size distributions) are cheap to calculate and are useful as parts of fingerprints in ML representations of extended porous materials.[131–133] Bucior *et al.* used supervised ML to predict hydrogen storage in MOFs from the discretised potential energy inside the pore network.[134] Similarly, Lee *et al.* used topological data analysis to quantify the structure of pores, which allows for the accurate comparison of two porous materials in a barcode-like fashion.[135] Furthermore, ML has assisted in speeding up the calculation of more difficult to access properties such as catalyst performance in carbon dioxide reduction,[136] MOF band-gap (using transfer learning)[137] and mechanical properties.[138,139] At present, there are no uses of AI in the design and property prediction of organic, extended porous materials.

PMMs, including porous organic cages (POCs), are a subclass of porous materials that have porosity derived from their internal cavity (intrinsic

porosity) and the packing of individual molecules in the solid-state (extrinsic porosity), which is defined by nonbonded interactions.[140] Similar to the field of organic electronics, the properties of some PMMs can be inferred from the properties of the single constituent molecules.[7] For example, Evans *et al.* discovered that single-molecule properties could reasonably predict whether a molecule would form a porous crystal using support-vector machine learning.[141] Sturluson *et al.* learned a latent space representation of POCs using unsupervised singular-value decomposition of the 3D shape and size of a POCs intrinsic pore.[142] While a proof-of-concept, they were able to show that the nearest neighbours in the latent space had similar intrinsic pores, and that exploration through latent space was possible, which could lead to the efficient exploration of POC phase space. Currently, the prediction of the molecular structure of a new cage from this latent space is not possible, but this work represents an exciting approach to exploring the POC search space. Using a database of ∼60 000 hypothetical and known POCs, Turcani *et al.* used ML to predict whether a cage would remain shape persistent, and if so, also predict its pore size.[143] Importantly, the input to their model is the chemical structure of the building blocks and proposed topology of the cage. Therefore, their ML algorithm could predict cage properties before constructing a model of the cage, which could drastically speed up the exploration of possible POC materials. However, their ML model required training on the target subclass of the cage (*e.g.* cages of a specific topology formed *via* imine condensation) to accurately predict the properties of new cages within that subclass.

Evolutionary algorithms (EAs) are efficient methods to explore chemical space. In particular, EAs are well equipped for the exploration of modular porous materials, where: (i) individual building blocks of a given material can be modified using simple methods, toward a new material; (ii) simple descriptors predict material properties, and (iii) material optimisation is a function of multiple variables. EAs have been applied to explore the chemical and structural space of MOFs to find candidate materials for adsorption applications.[144,145] Berardo *et al.* recently implemented an EA for the targeted discovery of POCs.[146] They implement a fitness function that focusses on the single-molecule properties of the cage (*i.e.* pore and

window size and window asymmetry) and highlight the ability to present a diverse set of high-ranking candidates in a series of case studies. During the cage discovery process, they monitor the dominant cage topologies and molecular properties in the population, which act as guiding design principles for future experimental work.

12.9.3 Other Functional Organic Materials

Organic polymers can be used for a multitude of other applications, including insulating materials, structural applications, and even information storage.[147] Interest in organic polymers for alternative uses has grown as they are cheaper to manufacture, more flexible, and have increased chemical resistance over most of their inorganic counterparts. The ability of NNs to generate complex, non-linear relationships between observables and output make them particularly useful for predicting polymer properties, where structure–property relationships are often complex or unknown. Early work involving polymer informatics by Sumpter and Noid used quantitative descriptors based on graph-theory to predict nine polymer properties simultaneously, including the molecular volume, heat capacity, glass temperature, and thermal conductivity.[110] Despite a small dataset of only 357 polymers, the model was able to achieve a maximum error of only 8%, using cross-validation to reduce overfitting. Generating new quantitative structure–property relationships is an effective strategy for predicting the tensile strength at break, providing an insight into the strength of the bulk polymer material. Cravero *et al.* showed that simplistic descriptors, based on the molecular weight of the polymer, used as input lead to a better predictive model than using automatic feature extraction of the underlying molecular structure.[148] Menon *et al.* were able to overcome the problem of small polymer datasets, using hierarchical machine learning (HML), to predict the stress-at-break, strain-at-break and tan δ using 18 polyurethanes as ground-truth data.[149] In HML, an intermediate layer containing physiochemical polymer properties was used to feed into the next layer of the ML architecture. This was shown to perform better than a direct property prediction with polymer composition data using a random forest model. More recently, Wu *et al.* employed ML to predict polymers with high thermal conductivity.[150] A Bayesian molecular design process

was employed to generate new candidate polymers, from regions of chemical space known to exhibit favourable properties. Owing to the small experimental thermal conductivity dataset, containing 28 polymers, transfer learning was used, and three candidates were selected and synthesised.

Packing density affects a significant number of properties, including mechanical, optical, and ionic conductivity. Using 197 descriptors from the Dragon 7 library, Atif Faiz Afzal *et al*. predicted the final packing density of amorphous structures.[151] Ground-truth training data was obtained from MD of simulated packed boxes of molecules and used as predictive values for the DNN. While not limited for predicting optoelectronic or mechanical material properties, ML has been applied to predict the redox potential of carbonyl-containing molecules. By calibrating the semi-empirical PM7 using GPR, Jinich *et al.* generated a model that was more accurate than using the group contribution method to estimate thermodynamic properties.[152]

12.10 Conclusions

Artificial intelligence is useful for a wide range of tasks in the field of organic materials; from accelerating the prediction of properties, *via* the *de novo* design of molecules and hence predicting molecular solids with desired properties, to assessing the likely synthesisability of such molecules. As in all applications of artificial intelligence, one major hurdle in many, if not all, of these tasks, is the availability of sufficient data to train models with. However, for molecular materials where a molecular representation is sensible, training data might be more easily generated or found in the literature than for other materials. The application of artificial intelligence in the area is thus clearly very promising and has surpassed the hype stage, suggesting that there are good reasons to believe that this is just the beginning of a bright future for artificial intelligence in organic materials discovery.

Acknowledgements

We acknowledge the Royal Society for a University Research Fellowship (K.E.J.) and an Enhancement Award to fund A.T. and the ERC through grant agreement number 758370 (ERC-StG-PE5-CoMMaD). S.B. acknowledges Leverhulme Trust for funding *via* the Leverhulme Research Centre for Functional Materials Design. M.A.Z. acknowledges the UK Engineering and Physical Sciences Research Council (EP/N004884/1).

References

1. K. Alberi, M. B. Nardelli, A. Zakutayev, L. Mitas, S. Curtarolo, A. Jain, M. Fornari, N. Marzari, I. Takeuchi, M. L. Green, M. Kanatzidis, M. F. Toney, S. Butenko, B. Meredig, S. Lany, U. Kattner, A. Davydov, E. S. Toberer, V. Stevanovic, A. Walsh, N. G. Park, A. Aspuru-Guzik, D. P. Tabor, J. Nelson, J. Murphy, A. Setlur, J. Gregoire, H. Li, R. Xiao, A. Ludwig, L. W. Martin, A. M. Rappe, S. H. Wei and J. Perkins, *J. Phys. D: Appl. Phys.*, 2019, **52**, 013001.
2. S. Rashidi, J. A. Esfahani and N. Karimi, *Renewable Sustainable Energy Rev.*, 2018, **91**, 229–247.
3. N. Enjamuri, S. Sarkar, B. M. Reddy and J. Mondal, *Chem. Rec.*, 2019, **19**, 1782–1792.
4. H. Bildirir, V. G. Gregoriou, A. Avgeropoulos, U. Scherf and C. L. Chochos, *Mater. Horizons*, 2017, **4**, 546–556.
5. K. Marsh, J. Boxall and R. Lichtenthaler, *Fluid Phase Equilib.*, 2004, **219**, 93–98.
6. T. B. Schon, A. J. Tilley, C. R. Bridges, M. B. Miltenburg and D. S. Seferos, *Adv. Funct. Mater.*, 2016, **26**, 6896–6903.
7. M. Miklitz, S. Jiang, R. Clowes, M. E. Briggs, A. I. Cooper and K. E. Jelfs, *J. Phys. Chem. C*, 2017, **121**, 15211–15222.
8. L. Wilbraham, E. Berardo, L. Turcani, K. E. Jelfs and M. A. Zwijnenburg, *J. Chem. Inf. Model.*, 2018, **58**, 2450–2459.
9. N. B. McKeown, *ISRN Mater. Sci.*, 2012, **2012**, 1–16.
10. P. G. Polishchuk, T. I. Madzhidov and A. Varnek, *J. Comput. Aided. Mol. Des.*, 2013, **27**, 675–679.
11. A. L. Ferguson, *J. Phys.: Condens. Matter*, 2017, **30**, 043002.

12. P. S. Gromski, A. B. Henson, J. M. Granda and L. Cronin, *Nat. Rev. Chem.*, 2019, **3**, 119–128.
13. G. Sliwoski, S. Kothiwale, J. Meiler and E. W. Lowe, *Pharmacol. Rev.*, 2014, **66**, 334–395.
14. C. A. Lipinski, F. Lombardo, B. W. Dominy and P. J. Feeney, *Adv. Drug Delivery Rev.*, 2001, **46**, 3–26.
15. A. Jain, Y. Shin and K. A. Persson, *Nat. Rev. Mater.*, 2016, **1**, 15004.
16. R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2018, **4**, 268–276.
17. B. Sanchez-Lengeling and A. Aspuru-Guzik, *Science*, 2018, **361**, 360–365.
18. P. Raccuglia, K. C. Elbert, P. D. F. Adler, C. Falk, M. B. Wenny, A. Mollo, M. Zeller, S. A. Friedler, J. Schrier and A. J. Norquist, *Nature*, 2016, **533**, 73–76.
19. M. Krallinger, O. Rabal, A. Lourenço, J. Oyarzabal and A. Valencia, *Chem. Rev.*, 2017, **117**, 7673–7761.
20. K. T. Butler, D. W. Davies, H. M. Cartwright, O. Isayev and A. Walsh, *Nature*, 2018, **559**, 547–555.
21. D. J. Audus and J. J. de Pablo, *ACS Macro Lett.*, 2017, **6**, 1078–1082.
22. L. Weston, V. Tshitoyan, J. Dagdelen, O. Kononova, A. Trewartha, K. A. Persson, G. Ceder and A. Jain, *J. Chem. Inf. Model.*, 2019, **59**, 3692–3702.
23. I. Korvigo, M. Holmatov, A. Zaikovskii and M. Skoblov, *J. Cheminf.*, 2018, **10**, 28.
24. V. Tshitoyan, J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K. A. Persson, G. Ceder and A. Jain, *Nature*, 2019, **571**, 95–98.
25. M. C. Swain and J. M. Cole, *J. Chem. Inf. Model.*, 2016, **56**, 1894–1904.
26. C. B. Cooper, E. J. Beard, Á. Vázquez-Mayagoitia, L. Stan, G. B. G. Stenning, D. W. Nye, J. A. Vigil, T. Tomar, J. Jia, G. B. Bodedla, S. Chen, L. Gallego, S. Franco, A. Carella, K. R. J. Thomas, S. Xue, X. Zhu and J. M. Cole, *Adv. Energy Mater.*, 2019, **9**, 1802820.

27. M. H. S. Segler, T. Kogej, C. Tyrchan and M. P. Waller, *ACS Cent. Sci.*, 2018, **4**, 120–131.
28. J. J. Irwin and B. K. Shoichet, *J. Chem. Inf. Model.*, 2005, **45**, 177.
29. J. Arús-Pous, T. Blaschke, S. Ulander, J. L. Reymond, H. Chen and O. Engkvist, *J. Cheminf.*, 2019, **11**, 20.
30. S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang and E. E. Bolton, *Nucleic Acids Res.*, 2019, **47**, 1102–1109.
31. J. Hachmann, R. Olivares-Amaya, S. Atahan-Evrenk, C. Amador-Bedolla, R. S. Sánchez-Carrera, A. Gold-Parker, L. Vogt, A. M. Brockway and A. Aspuru-Guzik, *J. Phys. Chem. Lett.*, 2011, **2**, 2241–2251.
32. R. Gómez-Bombarelli, J. Aguilera-Iparraguirre, T. D. Hirzel, D. Duvenaud, D. Maclaurin, M. A. Blood-Forsythe, H. S. Chae, M. Einzinger, D. G. Ha, T. Wu, G. Markopoulos, S. Jeon, H. Kang, H. Miyazaki, M. Numata, S. Kim, W. Huang, S. I. Hong, M. Baldo, R. P. Adams and A. Aspuru-Guzik, *Nat. Mater.*, 2016, **15**, 1120–1127.
33. E. B. Lenselink, N. ten Dijke, B. Bongers, G. Papadatos, H. W. T. van Vlijmen, W. Kowalczyk, A. P. IJzerman and G. J. P. van Westen, *J. Cheminf.*, 2017, **9**, 45.
34. C. Kim, A. Chandrasekaran, T. D. Huan, D. Das and R. Ramprasad, *J. Phys. Chem. C*, 2018, **122**, 17575–17585.
35. S. Grimme, C. Bannwarth and P. Shushkov, *J. Chem. Theory Comput.*, 2017, **13**, 1989–2009.
36. V. Ásgeirsson, C. A. Bauer and S. Grimme, *Chem. Sci.*, 2017, **8**, 4879–4895.
37. L. Wilbraham, R. S. Sprick, K. E. Jelfs and M. A. Zwijnenburg, *Chem. Sci.*, 2019, **10**, 4973–4984.
38. R. Ramakrishnan, M. Hartmann, E. Tapavicza and O. A. von Lilienfeld, *J. Chem. Phys.*, 2015, **143**, 084111.
39. J. S. Smith, B. T. Nebgen, R. Zubatyuk, N. Lubbers, C. Devereux, K. Barros, S. Tretiak, O. Isayev and A. E. Roitberg, *Nat. Commun.*, 2019, **10**, 2903.
40. S. Heinen, M. Schwilk, G. F. von Rudorff and O. A. von Lilienfeld, *Mach. Learn. Sci. Technol.*, 2020, **1**, 025002.
41. C. Draxl and M. Scheffler, *MRS Bull.*, 2018, **43**, 676–682.

42. A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder and K. A. Persson, *APL Mater.*, 2013, **1**, 011002.
43. T. D. Huan, A. Mannodi-Kanakkithodi, C. Kim, V. Sharma, G. Pilania and R. Ramprasad, *Sci. Data*, 2016, **3**, 160012.
44. S. A. Lopez, E. O. Pyzer-Knapp, G. N. Simm, T. Lutzow, K. Li, L. R. Seress, J. Hachmann and A. Aspuru-Guzik, *Sci. Data*, 2016, **3**, 160086.
45. S. J. Chalk, *J. Cheminf.*, 2016, **8**, 55.
46. V. Venkatraman, R. Raju, S. P. Oikonomopoulos and B. K. Alsberg, *J. Cheminf.*, 2018, **10**, 18.
47. S. S. Borysov, R. M. Geilhufe and A. V. Balatsky, *PLoS One*, 2017, **12**, e0171501.
48. S. A. Lopez, B. Sanchez-Lengeling, J. de Goes Soares and A. Aspuru-Guzik, *Joule*, 2017, **1**, 857–870.
49. A. Mannodi-Kanakkithodi, T. D. Huan and R. Ramprasad, *Chem. Mater.*, 2017, **29**, 9001–9010.
50. B. Olsthoorn, R. M. Geilhufe, S. S. Borysov and A. V. Balatsky, *Adv. Quantum Technol.*, 2019, **2**, 1900023.
51. A. Paul, Di. Jha, R. Al-Bahrani, W. Liao, A. Choudhary and A. Agrawal, *Int. Jt. Conf. Neural Networks, Proc., IEEE*, 2019, 1–8.
52. Y.-C. Lo, S. E. Rensi, W. Torng and R. B. Altman, *Drug Discov. Today*, 2018, **23**, 1538–1546.
53. F. Pereira and J. Aires-de-Sousa, *J. Cheminf.*, 2018, **10**, 43.
54. A. Mauri, V. Consonni, M. Pavan and R. Todeschini, *MATCH*, 2006, **56**, 237–248.
55. P. Gramatica, P. Pilutti and E. Papa, *J. Chem. Inf. Comput. Sci.*, 2004, **44**, 1794–1802.
56. D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.
57. G. Montavon, K. Hansen, S. Fazli, M. Rupp, F. Biegler, A. Ziehe, A. Tkatchenko, O. A. Von Lilienfeld and K. R. Müller, *Adv. Neural Inf. Process Syst.*, 2012, **1**, 440–448.
58. K. T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. R. Müller and E. K. U. Gross, *Phys. Rev. B*, 2014, **89**, 205118.
59. K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. von Lilienfeld, K.-R. Müller and A. Tkatchenko, *J. Phys. Chem. Lett.*,

2015, **6**, 2326–2331.
60. D. C. Elton, Z. Boukouvalas, M. S. Butrico, M. D. Fuge and P. W. Chung, *Sci. Rep.*, 2018, **8**, 9059.
61. S. De, A. P. Bartók, G. Csányi and M. Ceriotti, *Phys. Chem. Chem. Phys.*, 2016, **18**, 13754–13769.
62. J. Behler and M. Parrinello, *Phys. Rev. Lett.*, 2007, **98**, 146401.
63. G. Imbalzano, A. Anelli, D. Giofré, S. Klees, J. Behler and M. Ceriotti, *J. Chem. Phys.*, 2018, **148**, 241730.
64. J. Behler, *J. Chem. Phys.*, 2016, **145**, 170901.
65. D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik and R. P. Adams, *Adv. Neural Inf. Process Syst.*, 2015, 2224–2232.
66. T. Xie and J. C. Grossman, *Phys. Rev. Lett.*, 2018, **120**, 145301.
67. C. Chen, W. Ye, Y. Zuo, C. Zheng and S. P. Ong, *Chem. Mater.*, 2019, **31**, 3564–3572.
68. Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, *Chem. Sci.*, 2018, **9**, 513–530.
69. D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue and T. Lookman, *Nat. Commun.*, 2016, **7**, 11241.
70. A. Chandrasekaran, D. Kamal, R. Batra, C. Kim, L. Chen and R. Ramprasad, *npj Comput. Mater.*, 2019, **5**, 22.
71. S. L. Price, *Chem. Soc. Rev.*, 2014, **43**, 2098–2111.
72. G. M. Day, *Crystallogr. Rev.*, 2011, **17**, 3–52.
73. F. Musil, S. De, J. Yang, J. E. Campbell, G. M. Day and M. Ceriotti, *Chem. Sci.*, 2018, **9**, 1289–1300.
74. B. Rice, L. M. Leblanc, A. Otero-De-La-Roza, M. J. Fuchter, E. R. Johnson, J. Nelson and K. E. Jelfs, *Nanoscale*, 2018, **10**, 1865–1876.
75. A. M. Reilly, R. I. Cooper, C. S. Adjiman, S. Bhattacharya, A. D. Boese, J. G. Brandenburg, P. J. Bygrave, R. Bylsma, J. E. Campbell, R. Car, D. H. Case, R. Chadha, J. C. Cole, K. Cosburn, H. M. Cuppen, F. Curtis, G. M. Day, R. A. DiStasio, A. Dzyabchenko, B. P. Van Eijck, D. M. Elking, J. A. Van Den Ende, J. C. Facelli, M. B. Ferraro, L. Fusti-Molnar, C. A. Gatsiou, T. S. Gee, R. De Gelder, L. M. Ghiringhelli, H. Goto, S. Grimme, R. Guo, D. W. M. Hofmann, J. Hoja, R. K. Hylton, L. Iuzzolino, W. Jankiewicz, D. T. De Jong, J. Kendrick, N. J. J. De Klerk, H. Y. Ko, L. N. Kuleshova, X. Li, S.

Lohani, F. J. J. Leusen, A. M. Lund, J. Lv, Y. Ma, N. Marom, A. E. Masunov, P. McCabe, D. P. McMahon, H. Meekes, M. P. Metz, A. J. Misquitta, S. Mohamed, B. Monserrat, R. J. Needs, M. A. Neumann, J. Nyman, S. Obata, H. Oberhofer, A. R. Oganov, A. M. Orendt, G. I. Pagola, C. C. Pantelides, C. J. Pickard, R. Podeszwa, L. S. Price, S. L. Price, A. Pulido, M. G. Read, K. Reuter, E. Schneider, C. Schober, G. P. Shields, P. Singh, I. J. Sugden, K. Szalewicz, C. R. Taylor, A. Tkatchenko, M. E. Tuckerman, F. Vacarro, M. Vasileiadis, A. Vazquez-Mayagoitia, L. Vogt, Y. Wang, R. E. Watson, G. A. De Wijs, J. Yang, Q. Zhu and C. R. Groom, *Acta Crystallogr., Sect. B: Struct. Sci., Cryst. Eng. Mater.*, 2016, **72**, 439–459.

76. K. V. J. Jose, N. Artrith and J. Behler, *J. Chem. Phys.*, 2012, **136**, 194111.
77. J. S. Smith, O. Isayev and A. E. Roitberg, *Chem. Sci.*, 2017, **8**, 3192–3203.
78. K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller and A. Tkatchenko, *Nat. Commun.*, 2017, **8**, 13890.
79. F. Curtis, X. Li, T. Rose, Á. Vázquez-Mayagoitia, S. Bhattacharya, L. M. Ghiringhelli and N. Marom, *J. Chem. Theory Comput.*, 2018, **14**, 2246–2264.
80. D. H. Case, J. E. Campbell, P. J. Bygrave and G. M. Day, *J. Chem. Theory Comput.*, 2016, **12**, 910–924.
81. A. Z. Guo, E. Sevgen, H. Sidky, J. K. Whitmer, J. A. Hubbell and J. J. de Pablo, *J. Chem. Phys.*, 2018, **148**, 134108.
82. H. Sidky and J. K. Whitmer, *J. Chem. Phys.*, 2018, **148**, 104111.
83. D. McDonagh, C. K. Skylaris and G. M. Day, *J. Chem. Theory Comput.*, 2019, **15**, 2743–2758.
84. J. Yang, S. De, J. E. Campbell, S. Li, M. Ceriotti and G. M. Day, *Chem. Mater.*, 2018, **30**, 4361–4371.
85. E. Salager, G. M. Day, R. S. Stein, C. J. Pickard, B. Elena and L. Emsley, *J. Am. Chem. Soc.*, 2010, **132**, 2564–2566.
86. F. M. Paruzzo, A. Hofstetter, F. Musil, S. De, M. Ceriotti and L. Emsley, *Nat. Commun.*, 2018, **9**, 4501.
87. H. S. Stein, D. Guevarra, P. F. Newhouse, E. Soedarmadji and J. M. Gregoire, *Chem. Sci.*, 2019, **10**, 47–55.

88. N. Bernstein, B. Bhattarai, G. Csányi, D. A. Drabold, S. R. Elliott and V. L. Deringer, *Angew. Chem., Int. Ed.*, 2019, **131**, 7131–7135.
89. J. H. Jensen, *Chem. Sci.*, 2019, **10**, 3567–3572.
90. N. Brown, M. Fiscato, M. H. S. Segler and A. C. Vaucher, *J. Chem. Inf. Model.*, 2019, **59**, 1096–1108.
91. T. C. Le and D. A. Winkler, *Chem. Rev.*, 2016, **116**, 6107–6132.
92. F. Häse, L. M. Roch and A. Aspuru-Guzik, *Trends Chem.*, 2019, **1**, 282–291.
93. B. Cao, L. A. Adutwum, A. O. Oliynyk, E. J. Luber, B. C. Olsen, A. Mar and J. M. Buriak, *ACS Nano*, 2018, **12**, 7434–7444.
94. J. M. Granda, L. Donina, V. Dragone, D. L. Long and L. Cronin, *Nature*, 2018, **559**, 377–381.
95. A. Gupta, A. T. Müller, B. J. H. Huisman, J. A. Fuchs, P. Schneider and G. Schneider, *Mol. Inform.*, 2018, **37**, 1700111.
96. X. Yang, J. Zhang, K. Yoshizoe, K. Terayama and K. Tsuda, *Sci. Technol. Adv. Mater.*, 2017, **18**, 972–976.
97. M. Sumita, X. Yang, S. Ishihara, R. Tamura and K. Tsuda, *ACS Cent. Sci.*, 2018, **4**, 1126–1133.
98. M. Olivecrona, T. Blaschke, O. Engkvist and H. Chen, *J. Cheminf.*, 2017, **9**, 48.
99. M. Popova, O. Isayev and A. Tropsha, *Sci. Adv.*, 2018, **4**, eaap7885.
100. M. Benhenda, 2017, *arxiv:1708.08227*.
101. Q. Yuan, A. Santana-Bonilla, M. Zwijnenburg and K. Jelfs, *Nanoscale*, 2020, **12**, 6744–6758.
102. R. Barone and M. Chanon, *J. Chem. Inf. Comput. Sci.*, 2001, **41**, 269–272.
103. P. S. Kutchukian, N. Y. Vasilyeva, J. Xu, M. K. Lindvall, M. P. Dillon, M. Glick, J. D. Coley and N. Brooijmans, *PLoS One*, 2012, **7**, 48476.
104. C. W. Coley, L. Rogers, W. H. Green and K. F. Jensen, *J. Chem. Inf. Model.*, 2018, **58**, 252–261.
105. J. S. Schreck, C. W. Coley and K. J. M. Bishop, *ACS Cent. Sci.*, 2019, **5**, 970–981.
106. M. H. S. Segler and M. P. Waller, *Chem. – Eur. J.*, 2017, **23**, 5966–5971.
107. T. P. Osedach, T. L. Andrew and V. Bulović, *Energy Environ. Sci.*, 2013, **6**, 711.

108. N. M. O'Boyle, C. M. Campbell and G. R. Hutchison, *J. Phys. Chem. C*, 2011, **115**, 16200–16210.
109. V. Santolini, G. A. Tribello and K. E. Jelfs, *Chem. Commun.*, 2015, **51**, 15542–15545.
110. B. G. Sumpter and D. W. Noid, *Macromol. Theory Simul.*, 1994, **3**, 363–378.
111. R. Olivares-Amaya, C. Amador-Bedolla, J. Hachmann, S. Atahan-Evrenk, R. S. Sánchez-Carrera, L. Vogt and A. Aspuru-Guzik, *Energy Environ. Sci.*, 2011, **4**, 4849–4861.
112. F. Jabeen, M. Chen, B. Rasulev, M. Ossowski and P. Boudjouk, *Comput. Mater. Sci.*, 2017, **137**, 215–224.
113. E. O. Pyzer-Knapp, G. N. Simm and A. Aspuru Guzik, *Mater. Horizons*, 2016, **3**, 226–233.
114. A. Mannodi-Kanakkithodi, G. Pilania, T. D. Huan, T. Lookman and R. Ramprasad, *Sci. Rep.*, 2016, **6**, 20952.
115. E. O. Pyzer-Knapp, K. Li and A. Aspuru-Guzik, *Adv. Funct. Mater.*, 2015, **25**, 6495–6502.
116. P. C. St. John, C. Phillips, T. W. Kemper, A. N. Wilson, Y. Guan, M. F. Crowley, M. R. Nimlos and R. E. Larsen, *J. Chem. Phys.*, 2019, **150**, 234111.
117. I. Heath-Apostolopoulos, L. Wilbraham and M. A. Zwijnenburg, *Faraday Discuss.*, 2019, **215**, 98–110.
118. Y. Bai, L. Wilbraham, B. J. Slater, M. A. Zwijnenburg, R. S. Sprick and A. I. Cooper, *J. Am. Chem. Soc.*, 2019, **141**, 9063–9071.
119. T.-S. Lin, C. W. Coley, H. Mochigase, H. K. Beech, W. Wang, Z. Wang, E. Woods, S. L. Craig, J. A. Johnson, J. A. Kalow, K. F. Jensen and B. D. Olsen, *ACS Cent. Sci.*, 2019, **5**, 1523–1531.
120. B. Sanchez-Lengeling, C. Outeiral, G. L. Guimaraes and A. Aspuru-Guzik, 2017, ChemRxiv:5309668.
121. H. Furukawa, K. E. Cordova, M. O'Keeffe and O. M. Yaghi, *Science*, 2013, **341**, 1230444.
122. X. Feng, X. Ding and D. Jiang, *Chem. Soc. Rev.*, 2012, **41**, 6010.
123. P. Kaur, J. T. Hupp and S. T. Nguyen, *ACS Catal.*, 2011, **1**, 819–835.
124. S. Das, P. Heasman, T. Ben and S. Qiu, *Chem. Rev.*, 2017, **117**, 1515–1563.

125. Y. G. Chung, J. Camp, M. Haranczyk, B. J. Sikora, W. Bury, V. Krungleviciute, T. Yildirim, O. K. Farha, D. S. Sholl and R. Q. Snurr, *Chem. Mater.*, 2014, **26**, 6185–6192.
126. P. Z. Moghadam, A. Li, S. B. Wiggin, A. Tao, A. G. P. Maloney, P. A. Wood, S. C. Ward and D. Fairen-Jimenez, *Chem. Mater.*, 2017, **29**, 2618–2625.
127. Y. J. Colón, D. A. Gómez-Gualdrón and R. Q. Snurr, *Cryst. Growth Des.*, 2017, **17**, 5801–5810.
128. C. E. Wilmer, M. Leaf, C. Y. Lee, O. K. Farha, B. G. Hauser, J. T. Hupp and R. Q. Snurr, *Nat. Chem.*, 2012, **4**, 83–89.
129. R. L. Martin, C. M. Simon, B. Medasani, D. K. Britt, B. Smit and M. Haranczyk, *J. Phys. Chem. C*, 2014, **118**, 23790–23802.
130. R. L. Martin, C. M. Simon, B. Smit and M. Haranczyk, *J. Am. Chem. Soc.*, 2014, **136**, 5006–5022.
131. A. W. Thornton, C. M. Simon, J. Kim, O. Kwon, K. S. Deeg, K. Konstas, S. J. Pas, M. R. Hill, D. A. Winkler, M. Haranczyk and B. Smit, *Chem. Mater.*, 2017, **29**, 2844–2854.
132. C. M. Simon, R. Mercado, S. K. Schnell, B. Smit and M. Haranczyk, *Chem. Mater.*, 2015, **27**, 4459–4475.
133. M. Fernandez, T. K. Woo, C. E. Wilmer and R. Q. Snurr, *J. Phys. Chem. C*, 2013, **117**, 7681–7689.
134. B. J. Bucior, N. S. Bobbitt, T. Islamoglu, S. Goswami, A. Gopalan, T. Yildirim, O. K. Farha, N. Bagheri and R. Q. Snurr, *Mol. Syst. Des. Eng.*, 2019, **4**, 162–174.
135. Y. Lee, S. D. Barthel, P. Dłotko, S. M. Moosavi, K. Hess and B. Smit, *Nat. Commun.*, 2017, **8**, 15396.
136. A. W. Thornton, D. A. Winkler, M. S. Liu, M. Haranczyk and D. F. Kennedy, *RSC Adv.*, 2015, **5**, 44361–44370.
137. Y. He, E. D. Cubuk, M. D. Allendorf and E. J. Reed, *J. Phys. Chem. Lett.*, 2018, **9**, 4562–4569.
138. J. D. Evans and F.-X. Coudert, *Chem. Mater.*, 2017, **29**, 7833–7839.
139. P. Z. Moghadam, S. M. J. Rogge, A. Li, C.-M. Chow, J. Wieme, N. Moharrami, M. Aragones-Anglada, G. Conduit, D. A. Gomez-Gualdron, V. Van Speybroeck and D. Fairen-Jimenez, *Matter*, 2019, **1**, 219–234.
140. J. R. Holst, A. Trewin and A. I. Cooper, *Nat. Chem.*, 2010, **2**, 915–920.

141. J. D. Evans, D. M. Huang, M. Haranczyk, A. W. Thornton, C. J. Sumby and C. J. Doonan, *CrystEngComm*, 2016, **18**, 4133–4141.
142. A. Sturluson, M. T. Huynh, A. H. P. York and C. M. Simon, *ACS Cent. Sci.*, 2018, **4**, 1663–1676.
143. L. Turcani, R. L. Greenaway and K. E. Jelfs, *Chem. Mater.*, 2019, **31**, 714–727.
144. Y. G. Chung, D. A. Gómez-Gualdrón, P. Li, K. T. Leperi, P. Deria, H. Zhang, N. A. Vermeulen, J. F. Stoddart, F. You, J. T. Hupp, O. K. Farha and R. Q. Snurr, *Sci. Adv.*, 2016, **2**, e1600909.
145. Y. Bao, R. L. Martin, C. M. Simon, M. Haranczyk, B. Smit and M. W. Deem, *J. Phys. Chem. C*, 2015, **119**, 186–195.
146. E. Berardo, L. Turcani, M. Miklitz and K. E. Jelfs, *Chem. Sci.*, 2018, **9**, 8513–8527.
147. B. J. Cafferty, A. S. Ten, M. J. Fink, S. Morey, D. J. Preston, M. Mrksich and G. M. Whitesides, *ACS Cent. Sci.*, 2019, **5**, 911–916.
148. F. Cravero, M. J. Martínez, G. E. Vazquez, M. F. Díaz and I. Ponzoni, *J. Integr. Bioinform.*, 2016, **13**, 286.
149. A. Menon, J. A. Thompson-Colón and N. R. Washburn, *Front. Mater.*, 2019, **6**, 87.
150. S. Wu, Y. Kondo, M. Kakimoto, B. Yang, H. Yamada, I. Kuwajima, G. Lambard, K. Hongo, Y. Xu, J. Shiomi, C. Schick, J. Morikawa and R. Yoshida, *npj Comput. Mater.*, 2019, **5**, 66.
151. M. A. F. Afzal, A. Sonpal, M. Haghighatlari, A. J. Schultz and J. Hachmann, *Chem. Sci.*, 2019, **10**, 8374–8383.
152. A. Jinich, B. Sanchez-Lengeling, H. Ren, R. Harman and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2019, **5**, 1199–1210.
153. C. M. Simon, J. Kim, D. A. Gomez-Gualdron, J. S. Camp, Y. G. Chung, R. L. Martin, R. Mercado, M. W. Deem, D. Gunter, M. Haranczyk, D. S. Sholl, R. Q. Snurr and B. Smit, *Energy Environ. Sci.*, 2015, **8**, 1190–1199.
154. S. Otsuka, I. Kuwajima, J. Hosoya, Y. Xu and M. Yamazaki, in *2011 International Conference on Emerging Intelligent Data and Web Technologies*, IEEE, 2011, pp. 22–29.

CHAPTER 13

A New Era of Inorganic Materials Discovery Powered by Data Science

YA ZHUO, ARIA MANSOURI TEHRANI AND JAKOAH BRGOCH*

Department of Chemistry, University of Houston, Houston, TX 77204, USA
*Email: jbrgoch@central.uh.edu

13.1 Introduction

Empiricism has been the central force driving science forward, allowing us to gain and generate extensive knowledge and understanding about our surroundings. As our comprehension increased, numerous scientific theories in the form of numerical expressions were eventually developed that could describe or replicate various ideas or 'real-world' observations, such as the establishment of Newton's Laws. However, scientific complexity has increased seemingly exponentially, and simple mathematical expressions are not always capable of solving many of the modern scientific problems. Fortunately, the advent of computers, and the ideas propelling computational modeling and ultimately data science have transformed how researchers approach scientific questions. This evolution has been nicely categorized in four paradigms, as shown in Figure 13.1.[1]

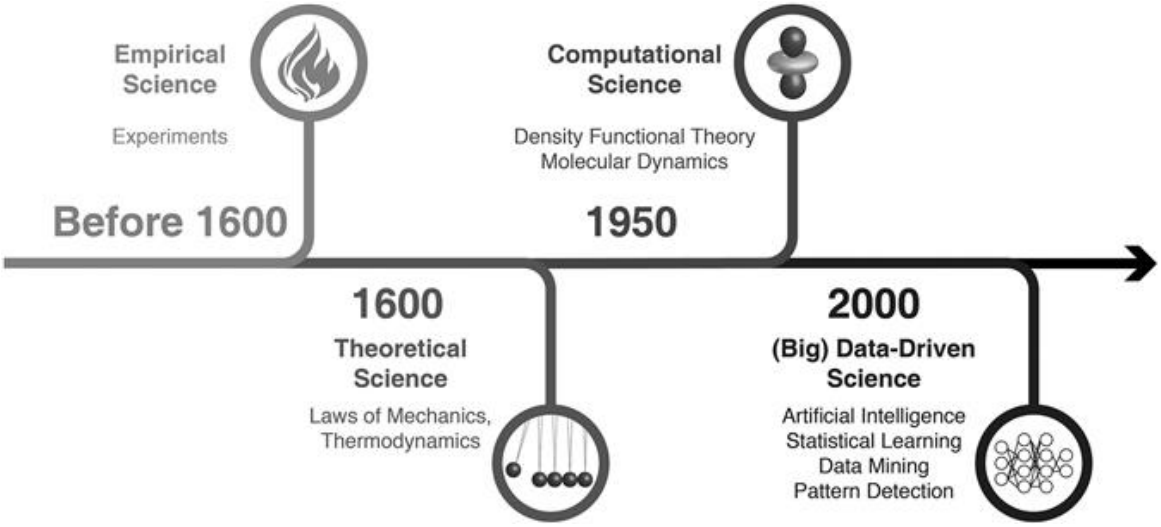


Figure 13.1 The four paradigms of science: empirical, theoretical, computational, and data-driven.

Chemistry began like all branches of science with qualitative, empirical observations and the development of chemical intuition followed by the construction of quantitative theories to explain these observations, like laws of thermodynamics. The eventual creation and implementation of advanced experimental approaches stemming from this second paradigm marked a significant revolution, allowing a better understanding of the observations. Inorganic, solid-state chemistry specifically was one of the subfields extensively strengthened by the introduction of improved synthesis techniques and advanced characterization techniques. The creation of methods to solve complex crystal structures or perform physical property measurements and data analysis over the past couple of centuries has led to the generation of significant amounts of data, most of which are now compiled in structured databases. For example, the Inorganic Crystal Structure Database (ICSD),[2] the Pauling File (now Pearson's Crystal Data; PCD),[3] the Materials Platform for Data Science (MPDS),[4] the NIMS Materials Database (MatNavi),[5] and the American Society for Metals datasets like the ASM Phase Diagram Database,[6] among numerous others, have endeavored to provide well-curated data sets that contain information on known (inorganic) crystal structures and measured physical properties. A majority of these data have been submitted directly by the research groups that generated the results, which is an excellent example of successful scientific ‘crowd-sourcing’. Some of the databases have also been

expanded through the laborious task of manually extracting data from the peer-reviewed literature.

The transition to the third, computational paradigm has been propelled by the outstanding accomplishments of computational materials chemists and spurred by their access to these sizeable crystallographic data sets. Numerous computational approaches have been developed over time, ranging from semi-empirical methods to *ab initio* methods. Density functional theory (DFT), as a subset of *ab initio* calculations, is arguably the most widely used among the inorganic materials community. Although it was first proposed in the 1960s, the recognition of its importance came only in the 1980s as the result of the increase in computational capacity and continuous theoretical refinement and expansion of the methodologies.[7] Indeed, DFT has made tremendous strides in determining the electronic structures of inorganic solids[8,9] even though limitations still exist.[10,11] For example, these calculations frequently require high-performance computing clusters, which are not always accessible or available. Also, DFT struggles to calculate disordered crystal structures and highly correlated systems, although DFT+U can improve the outcome.[12] DFT is usually able to accommodate only a limited number of atoms, typically a few hundred at most currently, which prevents researchers from modeling longer length scale interactions, such as microstructures, at the *ab initio* level. Nevertheless, the expansion of computational power has allowed DFT to obtain the electronic structure for thousands of inorganic solids as well as be implemented in an automated, high-throughput format where considerable computational resources are employed to ascertain the electronic structure of a vast number of compounds. Supported by the launch of the Materials Genome Initiative (MGI)[13] in 2011, the outputs from these high-throughput computational projects have been compiled in massive on-line databases providing accessible, robust, and trustworthy computational (DFT) results. Some of the most well-known examples are the Open Quantum Materials Database (OQMD),[14] Materials Project (MP),[15] and Automatic-Flow for Materials Discovery (AFLOW).[16] These databases are quite extensive, containing basic information like electronic structures, calculated elastic tensors, thermodynamic properties, and even predicted phase diagrams.

The MGI was arguably the catalyst not only for the creation of these large high-throughput computational databases but also for encouraging the transition to the fourth paradigm where these enormously useful repositories are employed directly for research. The accessibility to such high-quality data has

compelled investigators to think about the best pathway to utilize this ‘big-data’, which has been bolstered by subfields such as data analytics or informatics that are focused on understanding the principles behind data and the practice of information processing and the engineering of information systems. From these initial ideas, data science and machine learning have emerged as the principal methodology to analyze data sets, identify chemical trends, and suggest new opportunities to advance materials chemistry. The goal is to accelerate materials discovery by recognizing previously unknown correlations in a compound's composition–structure–property-processing relationship, as well as uncovering new qualitative and quantitative rules for materials development. Moreover, advanced experimental and computational techniques with their ever-growing rates of data acquisition and storage capabilities required new procedures for quickly assessing and analyzing the large collected data sets. Machine learning techniques and big data approaches have also provided an avenue to improve these methods by employing algorithms that can model the relationships between materials properties and related factors, even in high-dimensional data. Data science can also help decipher the intrinsic characteristics of materials and then reveal new hypotheses regarding the conditions that lead to successful products, which are much more challenging and critical for materials discovery.[17]

Given the potential provided by data science and its recent prevalence in inorganic materials chemistry, this chapter reviews some of the recent successful strategies where machine learning was essential for materials discovery and design. It also analyzes the challenges that the community is facing and those that should be overcome in the near future. It is anticipated that this chapter can introduce a new perspective toward data-driven materials chemistry research. In the rest of this chapter, we first introduce the formal description of machine learning as it pertains to inorganic materials chemistry, including discussing the commonly used steps in machine learning, and then survey the current research status with respect to the applications of machine learning with a focus on inorganic materials discovery. In particular, we highlight notable examples of synthetic process optimization, crystal structure prediction, and physical property prediction achieved using a variety of machine learning strategies. Finally, the most pressing challenges and research opportunities in this field are examined.

13.2 Workflow for Materials Discovery *via* Machine Learning

Artificial intelligence in computer science is the intelligence demonstrated by a machine aimed at mimicking ‘cognitive’ functions that humans associate with the human mind, such as ‘learning’ and ‘problem-solving’. Attempts have been made to approach the problem of acquiring knowledge by a machine through various symbolic methods, as well as methods based on connection principles, such as neural networks and perceptrons. The term ‘machine learning’, coined in 1959 by Arthur Samuel, grew out of the quest for artificial intelligence as a scientific endeavor. Its emphasis is more on logical and knowledge-based methods than on symbolic approaches. The goal of machine learning is to tackle solvable problems of a practical nature by developing methodologies and models derived from statistics and probability theories. Benefitting from the development of various learning algorithms over the years, machine learning has shown outstanding success in clustering, classification, regression, and other related tasks. Examples of applying machine learning specifically to inorganic materials chemistry can be traced back to the mid-1990s when approaches like neural networks were first employed to predict the melting point of binary intermetallic compounds and the mechanical behavior of fiber/matrix interfaces in ceramic-matrix composites.[18–20] The increasing availability of digitized information in the form of the materials-specific databases, and the ability to distribute data in a digital format has allowed machine learning to finally flourish. As a result, a substantial amount of research addressing various topics such as property prediction, new materials discovery, and materials design, has been published.

Regardless of the specific problem under study, a machine-learning problem can be broken into the following steps: data collection, data representation, data reduction, model construction, and model evaluation (see Figure 13.2). In the following subsections, a detailed explanation of each step as they relate to machine learning in inorganic materials chemistry is presented.

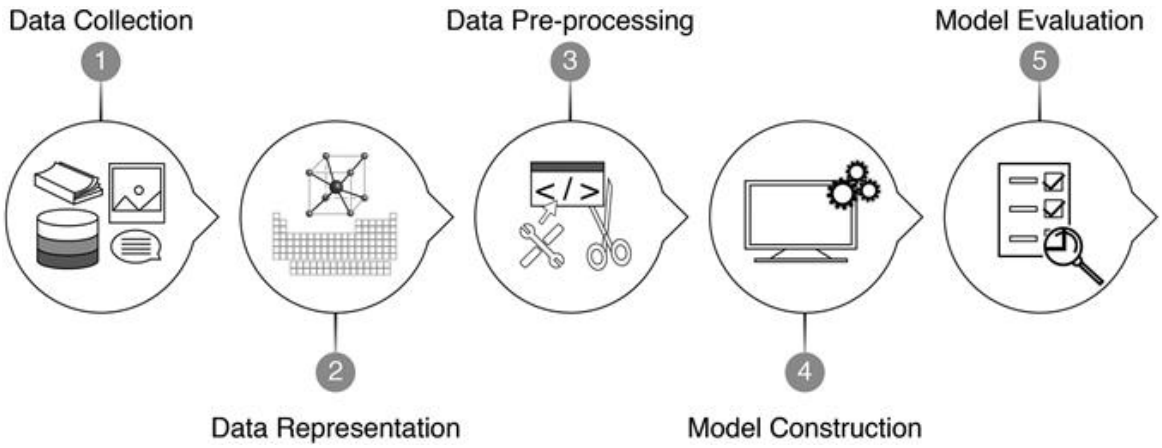


Figure 13.2 The general machine learning workflow.

13.2.1 Data Collection

The foremost prerequisite for any machine-learning task is the availability of a significant number of data in a machine-readable format that can be used as a 'training set'. Access to the chemical information is arguably the most pressing need that scientists must overcome to grow this research area. Fortunately, the large crystal structure databases like ICSD and PCD, the physical property databases like MPDS and MatNavi, and electronic structure databases like AFLOW, OQMD, or the Materials Project, have provided access to viable training data. Users can even efficiently extract information from some of these large data sets through the open Materials Application Programming Interface (API) and the open-source Python Materials Genomics (Pymatgen) materials analysis package.[21,22] Additionally, considerable scientific knowledge is present in figures, tables, and textual paragraphs. For example, laboratory notebooks contain extensive information that can be valuable as an input data set.[23] Peer-reviewed publications also contain troves of information; however, these data are challenging to analyze using traditional statistical analysis or machine learning methods. Several studies have used supervised natural language processing (NLP) to improve access to the information contained within research articles by text mining the scientific literature.[24–29] In one specific example, researchers encoded materials science knowledge present in the published literature as information-dense word embeddings (vector representations of words), which can capture the underlying structure of the periodic table and structure–property relationships in materials without human

labeling or supervision.[30] Extracting and gathering all of the necessary training data in a single location is the essential first step to any machine learning project.

13.2.2 Data Representation

Once a set of training data is available, the next step is to create numerical representations of this information that can be interpreted by computers, which are known as features, descriptors, predictors, or attributes. In the context of inorganic materials, features are mainly based on property-related compositions and crystal structures. The former captures a diverse range of physical and chemical properties of the compounds. These are developed based on the fractions of elements present in a chemical formula or through a series of mathematical expressions describing a variety of elemental properties. For example, Ward *et al.* used this kind of feature set to accurately predict a diverse set of properties, including the formation energy and bandgap energy of crystalline materials as well as the glass-forming ability in metal alloy systems, with tremendous success.[31] Even so, only using compositional features prevents the descriptions of compounds that form polymorph crystal structures. As an example, graphite and diamond would produce identical compositional features in machine learning; yet, their mechanical properties are distinctly different. Systems that have invariance in their compositional features, but a large discrepancy in the corresponding properties, are something that current algorithms simply cannot appreciate. One solution is to incorporate structural features or augment the compositional features to distinguish between graphite and diamond. In terms of structure descriptors, different from molecular systems, which can easily lend themselves to descriptions as graphs, the conventional description of inorganic crystal structures employs primitive translation vectors and fractional coordinates of the atoms so that a crystal can be represented in an infinite number of equivalent ways by choice of different coordinate systems. This type of representation is not unique and thus not a suitable set of features for the learning process. Representations based on partial radial distribution function (PRDF),[32] Voronoi tessellations,[33] or property-labeled materials fragments[34] are among the promising approaches to tackle the problem of structure. Currently, multiple materials informatics packages have been developed and are widely available, such as Pymatgen,[21] Matminer,[35] and MatErials Graph Network (MEGNet)[36] to generate features with minimal effort.

It is essential that the representations uniquely characterize each material in a data set and relate to the essential physics and chemistry that influence the property of interest.[37,38] Unfortunately, there is not a unified feature set that is effective across inorganic materials chemistry. Caution must, therefore, be exercised when selecting features. For example, the density of a crystalline compound is expected to relate to the density of the constituent elements. Including the average density of the constituent elements as a descriptor, a machine-learning algorithm could recognize the correlation between this value and the density of the crystal and then build a predictive model. Nevertheless, the mean density of the constituent elements neither uniquely describes a composition nor perfectly represents the density of the compounds. Therefore, additional descriptors must be included to create an appropriate feature set for this problem. However, the choice of descriptors is rarely simple; for instance, one possible feature to include in this problem might be crystal structure information that reflects the atomic packing scheme, but it is never clear. To overcome this conundrum, most researchers will create an excessively large feature set that includes all possible features and use data reduction techniques to determine the specific features related to the desired prediction. It is also imperative to examine the final set of features to ensure the reduced set represents experimentally or chemically reasonable information. Take the prediction of Li ion conductivity as an example.[39] Although ionic conductivity is expected to be influenced by several intrinsic and extrinsic factors, only ion diffusivity, average volume, transition temperature, and experimental temperature are of experimental interest. Therefore, it is necessary to ensure that these experimentally relevant factors remain in the learning process and are not removed during feature selection.

Finally, a typical good feature set should be universal, although this is not strictly necessary when performing analysis within a well-constrained chemical space. Features also have to be able to be reversed back into a description of the material in cases such as inverse design, in which the desired target property is used as inputs to predict the composition and structure.[40]

13.2.3 Data Pre-processing

Data pre-processing usually requires a majority of a data scientist's time and is one of the most critical steps in machine learning. Analyzing data that has not been carefully pre-processed can produce misleading results. For instance, if a dataset includes a significant portion of irrelevant, redundant information, or

noisy and unreliable information, then the knowledge gained by the algorithm during the training step may be questionable. This problem is often more significant in materials chemistry as the datasets are not, in most of the cases, 'big data' and usually range between a couple of hundreds to thousands. Therefore, within such relatively small datasets, each datum can be infulential to the performance of the model. Data pre-processing can usually influence the outcomes of the model or the ability to interpret the results. Proper pre-processing is, therefore, critical to the accuracy of the machine-learning model. Typically, pre-processing tasks include data cleaning, transformation, and reduction, as illustrated in Figure 13.3. In the following, we will expand on the discussions and meanwhile present the most well-known algorithms for some steps of data pre-processing as they pertain to inorganic materials chemistry.

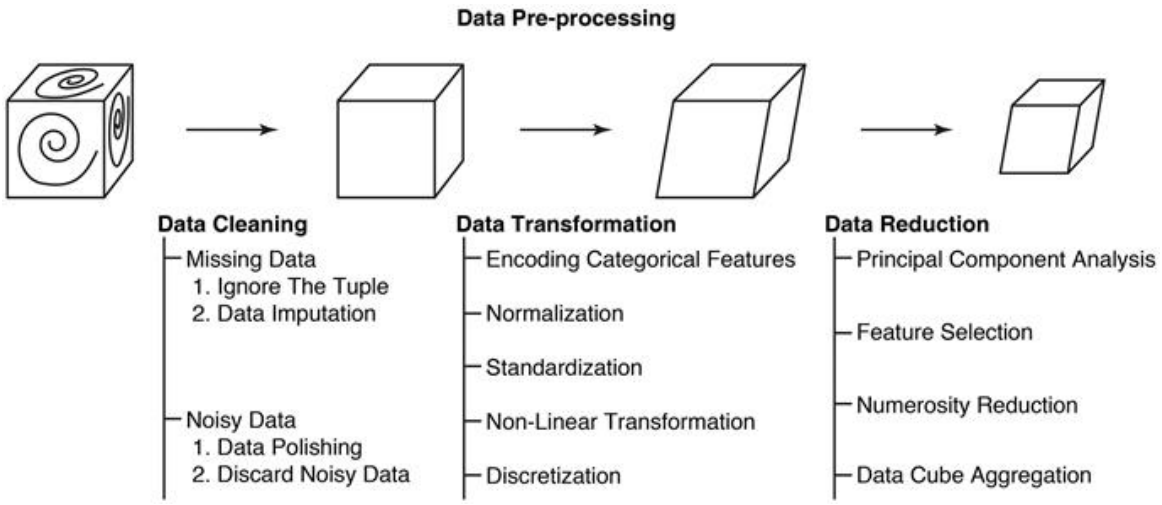


Figure 13.3 Data pre-processing tasks and some commonly used approaches.

13.2.3.1 Data Cleaning

Most techniques in machine learning rely on data sets that are complete and noise-free. However, real data are usually incomplete, noisy, and inconsistent, which can make analysis problematic. These imperfections in scientific data are often the result of complex issues like instrument limitations or technical problems at the time of data collection, as well as simple problems like the duplication of data or human error. Therefore, a cleaning step is essential to convert the original information into a trustworthy data set.

The occurrence of missing values is pervasive in the acquisition processes. A missing value is a datum that has not been stored or gathered due to non-existence, a flawed sampling process, cost restrictions, or limitations in the acquisition process. Incomplete data are usually an unavoidable problem, but they tend to cause severe difficulties for researchers because handling missing values, particularly in materials science, is generally difficult. Inappropriate treatment of the missing values can lead to inadequate knowledge extraction, strong biases, and wrong conclusions.

Many approaches are available to tackle the problem imposed by missing values. The simplest is to discard the instances that contain a missing value. However, this approach is beneficial only when a data set is quite large, and multiple values are missing within a tuple. Otherwise, it may produce a bias in the learning process as a result of producing an even smaller data set. The seminal works on data imputation stem from statistics and rely on probability functions. For instance, it is possible to use maximum likelihood procedures, where approximate probabilistic models are employed to fill the missing values. It is also possible to predict the missing value by developing a regression or classification model based on a complete data set for a given feature.

Another problem associated with the data set can be noise generated from faulty data collection, or data entry errors, among other sources. Data mining algorithms assume that any data set is a sample of an underlying distribution with no disturbances. In specific machine-learning problems, noise can have a significant impact on the input features. The situation is made worse when the noise affects the output targets or both inputs and outputs, as this means that the bias introduced will be more significant. To remove noise in data, two approaches are commonly applied. The first is to use data polishing methods to correct the noise, especially if it affects the labeling of an instance. Even partially correcting noise is beneficial to the model; however, this requires knowledge of the data set and is usually limited to correcting small amounts of noise.[41] The second approach is to use noise filters, like variable-by-variable data cleaning, to detect and remove noisy instances in a data set. An advantage of using noise filters is that they are straightforward to implement and require minimal knowledge about the data set. Noise identification and elimination are still active areas of research and development.[42,43]

13.2.3.2 Data Transformation

Transforming the cleaned data in an appropriate form suitable for the learning process is the next step. Mandatory transformations must be done for data compatibility. For example, a classifier was constructed to distinguish metals from insulators. Because machine-learning models are based on mathematical equations, it can sometimes be problematic if categorical strings such as ‘metal’ or ‘insulator’ are fed directly into the computer. Instead, these are generally converted into machine-readable numerical values before modeling, *e.g.*, encoding metals and insulators as the integers ‘0’ and ‘1’, respectively. Often features are also not available as numerical values but are instead categorical; it is mandatory to convert these features to numerical representations.

Optional quality transformations may also improve model performance. The continuous features in a data set are often at varying scales; for instance, the melting point covers a broad range from a few kelvin to thousands of kelvin. A standardizer could be used to translate this information to a scale where 0 represents the average melting point for the data set, and −1/+1 represents one standard deviation from the average. In general, learning algorithms benefit immensely from standardizing the data. Robust scaling or transforming is more appropriate if outliers are present in the data set. Many popular machine-learning estimators may behave poorly if the individual features do not follow a standard distribution, which is usually considered to be a Gaussian distribution with zero mean and unit variance. For example, elements used in the objective function of a learning algorithm, such as the l1 and l2 regularizers of linear models or the RBF (radial basis function) kernel of Support Vector Machines, assume that all features are centered at zero, and have variance in the same order. If a feature has a variance that is orders of magnitude larger than others are, it might dominate the objective function and make the estimator unable to adequately learn from other features. Practically, there is an almost unlimited number of ways that a data set can be transformed. To provide one example, transformation based on a logarithm function is a commonly used method of dealing with an exponential data series.

Discretization is another action sometimes needed in the pre-processing step. For example, in the process of inducing decision trees and deriving decision rules, the importance of one set of features tends to be overestimated if it contains too many continuous values.[44] Discretization, which partitions continuous numerical features into discrete values, is a useful process to address this problem. Discrete values are closer to a knowledge-level representation than continuous values, making data more understandable and easier to use,

thereby improving predictive accuracy. Numerous discretization methods have been proposed and tested. Feature binarization converts numerical features to binary features with the assumption that data follows a Bernoulli distribution. A bucketizer discretizes a set of continuous features into bins. Generally, most discretization methods can be grouped into *top-down* (splitting) and *bottom-up* (merging) methods.[45] Top-down methods start with an empty list of cut-points and recursively split data into smaller intervals. As an example, Maas *et al.* examined candidate cut-points against an error function and explored a search space of boundary points to minimize the number of wrong predictions in a classification problem.[46] Conversely, bottom-up methods start with the complete list of all continuous values as cut-points and iteratively merge neighboring intervals. One example is Khiops, where data can be discretized based on the global optimization of chi-squared.[47] Although many methods have been developed, the selection of the interval borders and the correct arity for the discretization of a numerical value range remain open questions.

13.2.3.3 Data Reduction

Data mining is a technique that is utilized to handle vast amounts of data. Working with a large volume of data makes the analysis more difficult, leading to reduced accuracy, low efficiency, and increased cost. Performing data reduction is valuable because it can improve model accuracy by reducing the number of misleading data, save storage space, and lower computing needs. The critical challenge in proper data reduction is to produce a condensed representation that is smaller in volume yet retains the integrity of the original data set. Additionally, data reduction can help simplify a model. The complexity of the model increases with the increasing number of features potentially causing overfitting, because as a machine-learning algorithm is trained on a large number of features, it gets increasingly dependent on the data. Avoiding overfitting is one of the primary motivations for performing data reduction, with the added benefit that the resulting models tend to be more straightforward.

A few methods are often used to reduce the volume of data. Principal component analysis, or PCA, is a statistical method that reduces the numbers of features by lumping highly correlated features together regardless of the importance of the meaning of these features. With each iteration, orthogonal transformation converts the features into principal components, which are a set of linearly uncorrelated values. Feature selection is another useful procedure to

reduce the data volume by identifying and removing irrelevant and redundant features. Straightforward feature selection methods are low variance filter, which aims to remove less informative features that are characterized by having a low variance, as well as a high correlation filter, which discards features that have a correlation coefficient higher than a threshold, meaning other features carry similar information. Algorithms associated with a stopping criterion are also employed to generate a subset of features and to evaluate the performance of a proposed subset. A stopping criterion can be: (i) including (or excluding) any feature does not produce a better subset; and (ii) an optimal subset is obtained according to the evaluation function. Without a proper stopping criterion, the selection process may run exhaustively or forever through the space of subsets. The genetic algorithm (GA) is a well-known approach for feature selection. It generates a feature subset that is only composed of the 'winning' features in the race between being included in a subset or excluded from a subset.[48,49] Research has also shown the possibility of using a particular learning algorithm as a pre-processor to discover useful feature subsets. Recursive feature elimination (RFE) is a heuristic method to remove irrelevant features. First, an external estimator is trained on the original features, and weights are assigned to each feature. Then, comparison of the weights indicates the least important features and recursively prunes these features from subsequent sets until a desired number of features is reached.

Dimensional reduction techniques have become popular for producing condensed feature representations to speed up model construction. Many of these ideas have also been implemented by materials chemists to establish process–structure–property relationships and for exploratory data analysis to understand trends in multivariate space.[50] For example, Dey *et al.* used PCA to identify the features that cause outliers in the task of predicting the bandgap energy for chalcopyrite compounds.[51] Agrawal *et al.* employed a ranking-based feature selection to get insights about the relative predictive potential of the input features for the problem of fatigue strength prediction of steels.[52]

13.2.4 Model Construction

The data are ready for modeling once appropriate data pre-processing has been completed. A model bridges the finger-printed features to the target property through mathematical functions. A range of algorithms are available, among them, some use actual functional forms that relate input to output, *e.g.*, support vector machine, whereas others do not, *e.g.*, decision tree. Moreover, the

amount of input data strongly dictates the choice of learning algorithms. For example, tens to thousands of data may be adequately analyzed using algorithms such as kernel ridge regression or Gaussian process, but a much larger data set (>hundreds of thousands or millions) may be necessary for artificial neural networks or deep learning. In some cases, a single algorithm may work poorly, but the use of several estimators built within a single learning environment may improve model generalizability and robustness; this is called an ensemble method and includes methods such as bagging and boosting. In addition, intelligent optimization algorithms, such as genetic algorithms (GA) and particle swarm optimization (PSO) are sometimes employed to optimize the model hyper-parameters.[53,54] Regardless of approach or methodology, one of the most significant distinctions is that learning can be supervised and unsupervised depending on whether or not the outputs are labeled.

13.2.4.1 Supervised Learning

A significant amount of practical machine-learning problems in materials science rely on supervised learning. It requires that the ground truth, or the knowledge of what the output values for the samples should be, is prior available. The goal of supervised learning is to produce a mapping function from the input features (x) to the output targets (y) so that the model can predict an output y when given a new input x. Supervised learning models predict output values within a discrete set or a continuous set. The former is seen in classification problems, *e.g.*, categorizing a material as a metal or an insulator, whereas the latter is the output from a regression model, *e.g.*, predicting a material's bandgap energy. There is a variety of supervised learning algorithms that can be applied depending on specific tasks. The selection of an appropriate algorithm is critical in model construction because it dramatically influences the prediction accuracy and model generalizability. Each approach has its unique features, and no one algorithm is suitable for all problems. In the following, some commonly used algorithms in materials chemistry are outlined.

K Nearest Neighbor (KNN) is an instance-based algorithm used in both classification and regression.[55] It hinges on the assumption that similar things exist in close proximity and that the distances between points in a descriptor hyperspace can be calculated. The prediction is then determined based on the values of the K ‘nearest neighbors’ in the data, where K is an integer value specified by the user. In classification, the prediction is the most frequent label

within the region of K nearest points, whereas in regression, the prediction is determined to be the average of the K nearest points.

Decision tree (DT) is a non-parametric algorithm that creates a tree-like flowchart to support each decision step.[56] It can handle both classification and regression tasks, although it is less appropriate for regression tasks. The model is trained by recursively partitioning the source set into subsets based on attribute value tests until members of the subset at one node all have the same value of the target, or splitting no longer adds value to the predictions. The tree depth determines the complexity of the decision rules and fitting. It is worth noting that an overly complex tree cannot generalize the data well. Mechanisms such as pruning, setting the minimum number of samples required at a node, or setting the maximum depth of the tree are, to some extend, necessary to avoid this problem.

Kernel methods are a class of algorithms for pattern analysis, whose best-known members are support vector machine (SVM), kernel ridge regression (KRR), and Gaussian process (GP).[57] Kernel methods work as a similarity function over pairs of data points, *i.e.*, it transforms two input data points into an output that quantifies how similar they are. Any kernel method solution comprises two components: a kernel function that performs the feature mapping and a learning algorithm used to discover the patterns in high dimensional feature space. Some popular kernel functions include linear, polynomial, sigmoid, while the radial basis function (RBF) is probably among the most used, as it has localized and finite response along the entire x-axis. Kernel methods are modular in the sense that any kernel will work with any kernel-based algorithm.

Artificial neural networks (ANN) and deep neural networks (DNN) are computing systems that are intended to mimic the way that humans learn.[58,59] They consist of artificial neurons arranged in input and output layers, as well as (in most cases) hidden layers. In a straightforward computation, hidden layers transform input signals into something that the output layer can use. Neurons and connections between neurons typically have weights, indicating the strength of the signal. Learning is the process of adjusting the weights to reproduce the training data as accurately as possible.

13.2.4.2 Unsupervised Learning

Unlike supervised learning, unsupervised learning does not have labeled outputs, so its goal is to infer the inherent structure present within a set of data.

The most common task within inorganic materials chemistry is clustering. Instead of directly predicting new values, clustering divides data into groups based on a similarity metric in a way that uncovers patterns and categories. Several algorithms are available today for automatically clustering data, such as *k*-means clustering, Ward hierarchical clustering, Gaussian mixtures, and density-based spatial clustering of applications with noise (DBSCAN). Each algorithm partitions data based on a geometric metric; for example, the *k*-means method aims to minimize the squared distances between data and cluster centers while Gaussian mixtures group data based on Mahalanobis distances to centers. Sometimes, additional parameters depending on the algorithm, *e.g.*, the number of clusters or distance threshold, need to be defined before performing a clustering analysis. Iterative algorithms may also require different trials to find an optimal grouping. The results of a clustering analysis can be used downstream. The advantage of using this approach is that materials in different clusters might bury some non-obvious physical characteristics, and a separate model thus can be constructed for each cluster. By using this approach, Meredig and Wolverton identified electronic structure features of dopant oxides and predicted zirconia dopant stability.[60]

Unsupervised learning is also becoming popular in experimental data analysis, where the collected data set is used as the fingerprint. Shen *et al.* developed robust convolutional neural network approaches to reconstruct crystal orientations from electron backscatter diffraction patterns.[61] It is also possible to use this approach to quickly determine the presence of different crystallographic phases in a sample analyzed using high-throughput X-ray diffraction, allowing the automated creation of phase diagrams and identification of new phases.[62–65] Merging machine learning approaches with these types of studies will be critical to advance the field of materials chemistry and aid the discovery of new compounds and crystal structures.

13.2.5 Model Evaluation

In principle, errors of a model arise from three sources: model bias, model variance, and irreducible errors, with the total error being the sum of these. Bias comes from erroneous assumptions taken by the learner. High bias can lead to missing underlying relations between features and target outputs. Variance is the result of sensitivity to small fluctuations in the training set. High variance models have limitations on the flexibility to learn the true signal from the data. Irreducible errors exist in every model, even in well-trained ones, and they stem

from the inherent variability in data due to systematic errors in laboratory instruments, calculation uncertainties, or simply from outliers or missing data. A poorly performing model usually indicates high bias or high variance.

A robust machine-learning model should have adequate prediction power on not only existing data but also unseen data. A widely used approach for evaluating the quality of a model involves partitioning the original data set into a training set for model construction and a test set for model evaluation. The data in the test set should be randomly selected and completely unseen from the model. It is advisable, especially for small data sets, to use multiple (>3) randomly selected training and test sets to ensure the robustness of the model. To avoid the introduction of additional deviations, sampling methods can be used to maintain the consistency of the data distribution to the highest possible extent. The model performance can be approximated by evaluating the extent to which the output data in the test set is accurately predicted. Cross-validation (CV), as a second approach for model evaluation, partitions the original data set into mutually exclusive subsets. For example, *k*-fold CV divides the data set into *k* subsets of the same size, and the union set of *k*-1 subsets is taken to train the model, while the remaining one is used as the test set. This process is repeated for each subset. A particular case is when *k* equals the number of original data, called leave-one-out cross-validation (LOOCV). LOOCV is a powerful CV approach to ensure a model's validity; however, it is not necessarily suitable for large data sets because it can take a long time for the cross-validating process, and it may also artificially underestimate the error for highly correlated data sets. Recently, leave-one-cluster-out cross-validation (LOCOCV) was proposed as a new measure of model performance geared toward materials discovery.[66] Instead of a standard LOOCV, LOCOCV first generates clusters and then removes the entire cluster in the cross-validation step. This approach offers a more realistic evaluation by estimating the ability of the model to extrapolate by removing entire groups providing information on the capability to extrapolate to entire classes of unseen materials.

13.3 Applications of Machine Learning in Inorganic Materials

The use of data-driven tools has the potential to result in significant materials advances, whether through the enumeration and analysis of experimental and

computational data or the codification of chemical intuition. The purpose of building predictive models is not to replace experiment and computation, but rather to focus the effort in these research areas. In this section, we highlight a selection of research projects applying many of the machine-learning techniques that have been discussed here to advance the field as well as reduce the barriers to understand the compositions–structure–property-processing relationships in inorganic materials.

13.3.1 Machine Learning in Synthesis Optimization

Determining the optimal reaction condition or processing procedures to prepare an inorganic material has historically relied extensively on heuristics, theoretical analysis, and the accumulation of experience. As a result, the targeted production of new materials has always been a slow step in materials realization. A more productive approach might be to establish a comprehensive and accessible link between the specific material and the requisite preparation conditions to achieve the compound. Given the enormous complexities in the synthesis processes, machine-learning methods could be one of the most beneficial approaches to unify the different factors in the synthetic route and suggest the optimal conditions. One example where this has proved successful is in the study of Al–Zn–Mg–Cu series alloys. Fang *et al.* used least square support vector machine (LSSVM) models to optimize the aging temperature and aging time for targeting the product with high hardness and high electrical conductivity, followed by hybridizing nondominated sorting-based multi-objective genetic algorithm (MOGA) and LSSVM to make tradeoffs between hardness and electrical conductivity.[67] Ling *et al.* employed ensemble machine-learning algorithms to map functions between mechanical properties, such as tensile yield strength and shear strength, and processing steps, such as heat treatment and pressure treatment, in aluminum alloys and nickel-based superalloys.[68] Neural networks are also showing particular promise for predicting materials' synthesis routes. Kim *et al.* developed a framework that compiled inorganic materials synthesis recipes across over half a million scholar publications through natural language processing and used these recipes to train machine-learning models to obtain insights into the critical parameters, such as calcination temperature, needed to synthesize specific, technologically relevant materials at a high level of automation.[69] In another study, failed or unsuccessful hydrothermal syntheses collected from laboratory notebooks, termed 'dark reactions', were analyzed using machine-learning models to

determine the optimal reaction conditions for the crystallization of templated vanadium selenites.[23] Their model exceeded human intuitive strategies, and successfully predicted synthesis conditions for new organically templated inorganic product formation with a success rate of 89%.

Given the fact that numerous competing factors simultaneously govern the preparation of a material, it is sometimes complicated to directly model the complex kinetic processes with mathematical expressions by conventional methods. Fuzzy neural networks (FNN) offer an alternative to handle imprecision and uncertainties presented in synthesis procedures. It integrates the excellent learning capability of neural networks with fuzzy inference for extracting the initial rules of a fuzzy system. For example, Han *et al.* trained an FNN prediction model as well as a multiple linear regression (MLR) model for comparison to establish the relationship between processing parameters during hot working and heat treatment processing and the resulting mechanical properties of Ti–10V–2Fe–3Al titanium alloy.[70] The optimum processing parameters leading to the desired mechanical property can be accurately selected based on the fuzzy model. The FNN method achieved better prediction precision and showed much better agreement with experimental results than the MLR method.

In addition, merging intelligent models with simulation techniques turns out to be efficient in some complex systems. For example, neuro-fuzzy and physically based models were jointly used to predict the flow stress and microstructural evolution during the thermomechanical processing of aluminum alloys.[71,72] The physically based components in the hybrid model facilitated extrapolation beyond process conditions, leading to better agreement with experimental data.

13.3.2 Machine Learning in Structure Prediction

Predicting the crystal structure preference of inorganic materials has always been of great interest. The crystal structure of any composition dictates the electronic structure and hugely influences its properties. Consequently, understanding the composition-crystal structure–property relationship in solids comprises a significant endeavor of inorganic chemists, emphasizing the importance of this subject. However, there is no practical method to predict the crystal structure of any given composition before synthesis. Often comprehensive characterization techniques such as single crystal structure diffraction are also required once a new compound has been prepared to solve

the structure. To achieve the goal of predicting a structure *a priori*, correlations must be identified based on innate elemental properties, while many other factors such as temperature and pressure are influential. A simple yet fundamentally imperative correlation was identified by Pauling in which the coordination number of crystals was predicted based on the radius ratio of anions and cations.[73] Many other empirical correlations followed, laying the foundations to decode the intricate complexity of composition crystal structure relationships of inorganic materials. More recently, the emergence of *ab initio* computational methods such as DFT to numerically estimate the ground-state energy of extended compounds based on atomic coordination made a considerable impact on the community with regard to a prior crystal structure prediction. Clearly, it is not possible to compare the DFT ground-state energy of all possible crystal structure-types, even ignoring the possibility of a new-crystal structure-type, to indicate the most energetically favorable structure. To approach this problem, methods based on evolutionary algorithms have been developed, such as Crystal Structure Analysis by Particle Swarm Optimization (CALYPSO)[74] and Universal Structure Predictor: Evolutionary Xtallography (USPEX)[75] to potentially reach the ground state structure significantly faster. However, it is difficult to ensure that the calculated structure is at a global minimum rather than a local minimum of the energy landscape. Further, these methodologies are still extremely computationally expensive, especially for *k*-nary (k>3) compounds, which in many cases negates the purpose of the prior prediction.

Researchers have also implemented an alternative approach that combines data mining with first-principle calculations. A probability heuristic framework is constructed to suggest candidate structures of compositions to predict alloy formation of metallic binaries.[76] Regardless, DFT calculations have their limitations; the calculations are performed at zero kelvin while many compounds go through phase transformation at higher temperatures, and the calculations cannot account for statistical site-sharing of atomic positions in a straightforward manner, while correlation effects should be dealt with carefully in some compounds, *e.g.*, transition metal oxides or compounds containing f electrons.

Similar to the original correlation diagrams and analysis that Pauling and others created to unravel the correlations between crystal structure and composition,[73,77–80] data-driven techniques are being implemented for the same purpose. Machine-learning algorithms are capable of constructing complex

correlations, potentially in multi-dimensional space unattainable to humans, in big data sets. Therefore, it is a natural transition to implement these tools to predict crystal structures. Thus far researchers have utilized various machine learning methods to predict crystal structure types. For example, Oliynyk *et al.* used a random forest algorithm to model the possibility of AB_2C compositions forming the Heusler structure.[81] Cluster resolution feature selection (CR-FS) in conjunction with support vector classification (SVC) was adopted to predict the crystal structure of AB and ABC compounds.[82,83] Moreover, Balachandran *et al.* implemented a multi-faceted framework by combining group theory, machine learning, and DFT calculations to identify new noncentrosymmetric materials.[84] Additionally, researches have utilized machine learning techniques to assist in solving X-ray diffractions and the automatic classification of diffractograms.[85–87] These examples highlight the success of predicting whether a generic composition will adopt a specific structure type.

To construct a more general machine-learning model for structure prediction, a random forest classifier has been implemented, which is capable of predicting the structure-type for any composition.[88] The main issue in classifying structure-types is that crystal structures are, in fact, distributed in an extremely unbalanced fashion. For almost 25 000 unique compositions extracted from PCD there are around 10 700 entry prototypes with most of them corresponding to only a few (<10) structures. Not only there is an extreme imbalance concerning the structure-type distribution but also there are many instances of unusual or exotic structure-types that are extremely rare, which may disturb the learning process.[89] Therefore, one might not expect the algorithm to identify very exotic crystal structure-types correctly. To partially circumvent this issue, a cut-off, ranging between 10–150 instances, is determined. Any structure-type with fewer instances is therefore classified as ‘other’. The algorithm with a cut-off of 100 shows precision and recall of around 85% and 68%, respectively, indicating it is capable of identifying the structure-type of any composition with reasonable accuracy. Despite its usefulness, one should note that this model cannot distinguish if a combination of elements will crystalize or not, and the prediction of new instances could be, therefore, purely hypothetical. In another approach, a deep neural network was used based on atomic fingerprints as inputs. The researchers have focused on predicting the likelihood of various elements occupying atomic sites through a combinatorial approach. The reason for this implementation is that there are no examples of crystal structures that do not form, which must be included as negative results in the learning process.

Moreover, to prevent the issue of elemental imbalance distribution of training data an oversampling method is utilized by increasing the number of the elements with low repetitions.[90]

Finally, a crystal structure prediction engine was developed using Bayesian optimization algorithm.[91] Based on a combination of exploration and exploitation, the number of trials to reach the ground-state energy is significantly reduced, as shown for the cases of NaCl and Y_2Co_{17}. This methodology starts with a random generation of crystal structure that is relaxed using DFT calculations. These energies, in conjunction with relevant created descriptors, form the first set of training data. A Bayesian optimization algorithm then selects another set of candidates, which are optimized with descriptors being updated before the next set of selections until the global minimum is reached. Machine learning methods have also been used to develop interatomic potentials for inorganic material. One research project has constructed a dubbed Gaussian approximation potential-based random structure searching tool that leads to crystal structure prediction of inorganic solids. This methodology was applied to elemental phosphorus, correctly predicting its polymorphs.[92] These approaches demonstrate exceptional potential in future crystal structure prediction methods by significantly reducing the time required for methods such as first-principle evolutionary structure searches, ultimately enabling the future investigation of larger and more complex systems.

13.3.3 Machine Learning in Material Property Prediction

The application of machine learning in predicting inorganic materials' physical properties involves determining the relationships between the properties, either computationally modeled or experimentally measured, and the feature set. Numerous research projects have already been carried out specifically for property prediction. For example, Isayev *et al.* combined data from *ab initio* calculations with quantitative materials structure–property relationship (QMSPR) models to predict several electronic and thermomechanical properties.[34] They first developed a calculation tool called property-labeled materials fragments (PLMF) to represent materials. Features generated with PLMF capture both structure geometry and local atomic properties. Using the gradient boosting decision tree (GBDT) method, a candidate material is classified as a metal or an insulator, followed by the prediction of bandgap value if the material is predicted to be an insulator. Regardless of the material's

metal/insulator classification, another six GBDT models are then constructed to predict thermomechanical properties, including bulk modulus, shear modulus, Debye temperature, heat capacity at constant pressure, heat capacity at constant volume and thermal expansion coefficient. Hyper-parameters of all models are optimized with grid searches, and the model's generalizability is assessed with five-fold cross-validation.

Calculations done in a high-throughput manner have been critical for maximizing data availability, which makes the data collection process much more efficient. However, models that learn from computational data may have some shortcomings. Take bandgap as an example; high-throughput DFT calculations can obtain a large number of bandgap values at the PBE-level of exchange and correlation. However, these calculations show a systematic underestimation (∼40%) of bandgap compared with the experimental values.[10,11] It is also often limited to ordered crystal structures and loses its reliability for highly correlated systems.[93] Therefore, any machine-learning model constructed using these DFT data are expected to underestimate the bandgap. At the same time, the predicted bandgap of site-mixing containing compounds or highly correlated compounds is less accurate than those ordered compounds. Alternatively, machine learning on a training set containing only experimentally measured bandgaps can be used to address many of those limitations.[94] The resulting model does not suffer from the DFT systematic error and achieves excellent agreement between the experimentally measured bandgaps and predicted bandgaps. The success of this experiment-based model is partially attributed to the relatively abundant availability of experimental data (∼4000 bandgaps) and, more importantly, the adequate selection of features. These data enable the model to capture the relative variation in bandgaps as a function of the features, eventually culminating in the prediction of bandgaps for new compounds. Unfortunately, in most cases, gathering a large amount of experimental data can be a difficult task for reasons such as the lack of reports or technical limitations in data extraction.

Conventional machine learning processes are not ideal for small data sets because the shortage in data can prevent the model fully capturing the inherent relationships between properties and features, resulting in a weak model, significant uncertainties in predictions, and thus selections of nonoptimal materials in a screening process. Fortunately, an adaptive procedure, called adaptive design or active learning, shows excellent potential for scientific problems where data are scarce. This strategy works as an iterative loop with

feedback from experiments. It balances the trade-off between exploitation, where the goal is to search for materials likely to have the best property, and exploration, where the goal is to improve the quality of the model by searching the space that contains fewer sampling points and greater uncertainty. Adaptive design has been successfully applied in areas such as computer science,[95] engineering optimization,[96] cancer genomics,[97] and recently materials chemistry.[98,99] For example, Xue *et al.* employed this adaptive design framework in the discovery of low thermal hysteresis NiTi-based shape memory alloys by using a limited number of 22 training examples.[99] An RBF embedded SVR regressor was first modeled to predict the thermal hysteresis with the uncertainties obtained from a bootstrap method, and then a knowledge gradient (KG) based selector used these predictions and their uncertainties to suggest the next material to investigate. By iterating the learning process nine times, 36 predicted compositions were synthesized and tested from a potential space of ∼800 000 compositions with 14 out of these 36 new compounds having a lower thermal hysteresis than any of the 22 compounds in the initial data set.

Predicting a material's properties with informatics techniques has also become a considerable research focus. Besides modeling materials' physical properties, such as formation energies and melting points, machine learning can also be employed to predict functional properties, such as ionic conductivities for battery materials or critical temperatures for superconductors.[39,100–102] In the example of predicting superconducting critical temperatures (T_c), >30 candidate materials were identified.[102] The learning process happened in two stages: a classifier first categorized a material into either the high-T_c class (T_c>10K) or the low-T_c class (T_c<10K); then a regressor predicted the values of T_c for compounds that were categorized to the high-T_c class. However, explicitly predicting the functionalities is not a straightforward action in problems where many internal or external factors greatly influence the target functional properties. For example, in the search for new intrinsically hard and superhard materials, explicitly predicting the hardness is hindered by the limited number of reports. Moreover, hardness is a load-dependent property, and the load is not always reported. Hardness is also related to microstructure, which is also often unknown. As an alternative, Mansouri Tehrani *et al.* employed bulk and shear modulus as proxies for hardness, which are two intrinsic properties easily extracted from high-throughput computational databases.[103] After

building a model using SVM based on a combination of compositional and structural features, the elastic moduli of more than one hundred thousand compounds were readily predicted for the screening for high bulk and shear compounds, which indicates the high hardness. The selected top candidates, $ReWC_{0.8}$ and $Mo_{0.9}W_{1.1}BC$, were revealed to be extremely hard, reaching the superhard threshold at low indentation load after the experimental examination of these systems. Similarly, a material's Debye temperature can act as a proxy for photoluminescence quantum yield of inorganic phosphors.[104] Screening for compounds of high predicted Debye temperature and adequate bandgap has lead to the discovery of a highly efficient blue-emitting phosphor.

13.4 Challenges and Opportunities

Machine learning has become pervasive throughout chemistry and a majority of its subfields, including inorganic materials. It impacts the way that we do experiments, analyze data, and it is even altering the way that we teach. There are undoubtedly plentiful opportunities to use data-driven methods to advance materials research and development (R&D) and manufacturing; however, numerous challenges also remain. Materials discovery is based on understanding and interpreting multivariate correlations in terms of the fundamental physics, chemistry, and engineering. In the following, some general difficulties limiting the application of machine learning in materials chemistry are identified, while some promising strategies to address each problem are discussed.

13.4.1 Learning from Small Data Sets

Perhaps, the most frequently asked question by new researchers that are eager to use data science is whether their problem lends itself to such methods. No matter the specific problem, a prerequisite for machine learning is the existence of reliable data or the ability to generate at least a subset of the critical information in a uniform and controlled manner. In materials chemistry, however, we are often limited to hundreds or thousands, sometimes even fewer, high-quality data points. One of the biggest hindrances is a combination of a relatively small number of well-characterized and reliable materials, with the absence of effective aggregation of this high-quality data. There are several existing data infrastructures that are quite general, and they usually provide

basic knowledge about compounds, *e.g.*, structure or fundamental physical and chemical properties, but these databases can be expensive to access. Some of these databases also do not allow bulk data downloads. For example, the Inorganic Crystal Structure Database (ICSD) is an authoritative collection of crystallographic data containing tens of thousands of compounds, but it is not straightforward to download the data in a bulk manner.

Expanding data sets through data mining techniques is one route to address small data sets, but for some scientific problems, there may simply not be more data available. A promising solution in these cases is through the modification of the learning process. Adaptive design (also mentioned in Section 13.3.3) is one method that has shown fantastic capability, where knowledge is quickly accumulated iteratively.[105] Meta-learning is also useful for handling limited data sets because knowledge in this approach is gained within and across problems.[106] New developments in computer science such as Neural Turing machines (NTM),[107] transfer learning,[108] and imitation learning[109] are enabling the realization of this process. For example, a Bayesian program learning (BPL) framework has recently been demonstrated to outperform recent deep learning approaches and even achieve human-level accuracy on one-shot learning tasks with limited data.[110] The outstanding performance of these models provides exciting potential if applied to materials chemistry where data are usually scarce, exorbitantly expensive and slow to obtain.

13.4.2 Generating Efficient Data Representation

Generally, a more suitable representation of the input data leads to a better model, whereby the relationship between the input and output data can be more accurately learned. Essential research efforts are being directed toward the development of efficient features for encoding complex materials data. However, either compositional features or structural features emphasize the importance of leveraging domain knowledge, resulting in a significant impediment to the use of machine learning in problems where the physics driving behavior is yet to be understood. One solution would be to learn the properties of materials from their atomic identities. Jha *et al.* designed a deep neural network model using only the elemental compositions as features to learn the chemical interactions and similarities between different elements automatically.[111] This methodology purely relies on representation learning without using any domain knowledge about materials.

13.4.3 Enhancing Model Interpretability

The knowledge extracted by machine-learning algorithms is usually difficult to interpret even though the model has predictive power. Several reasons could cause for this shortcoming. First, the knowledge is not presented in a way that scientists are familiar with; for example, when employing the SVM model in a classification task, the optimal hyperplanes that are created to separate each class are usually in multidimensional space and too complex to visualize, which prevents additional analysis. The second reason is that scientists sometimes are uncertain of the information produced by the algorithm and cannot link the input to the output in any rational, direct way, which makes machine learning truly a ‘black box’. The question is how to turn a ‘black box’ into a ‘white box’. To date, research on enhancing machine learning interpretability mainly focuses on developing methods that run alongside the existing black box models to generate statistical explanations in a human-understandable manner.[112] These methods range from ranking input features importance to constructing entirely new models that closely approximate the original black box model while using a much simpler, easier-to-understand algorithm such as linear regression. A better understanding of the machine-learning model could potentially accelerate the discovery of scientific laws and principles.[113]

13.4.4 Increasing Model Accessibility

Most currently available data informatics programs require users’ programming skills and knowledge about data science to analyze and manage data. Without this background, researchers may quickly become mired in jargon, arcane parameters, or unfamiliar syntax. As a result, data-driven materials chemistry can only be accessible to those who have appropriate experience in computer programming and data science. It is vital to democratize materials informatics. That way the community can benefit from the widespread access of large-scale materials data analysis. To achieve such a goal, software needs to be developed that is intuitive and easy to use for materials experts, even those who have never been trained in computer science and data science. Such tools have now been emerging for pervasive consumption by the materials community; examples include AFLOW-ML and MEGNet for machine-learning predictions of materials properties,[36,114] Pourbaix diagrams generator on the Materials Project,[115] Citrine Informatics's thermoelectric materials recommendation engine,[116] and many others. The continued development and deployment of

models with friendly graphical user interfaces will provide exceptional computational tools to researchers with a range of skill levels at institutions around the world.

13.5 Concluding Remarks

As a branch of data science and one of the most powerful tools of data analysis, machine learning has gained unprecedented attention in all scientific disciplines. The number of reported applications is growing at an extraordinary rate, and materials chemistry is not different. There has been a steady shift from experimental and computational focused research toward data-driven studies with close collaboration between computational predictions and experimental validation. The objects under study, particularly in the field of inorganic materials chemistry, are extensive, including inorganic oxide materials, electrolyte materials, metallic materials, functional materials, and numerous others. The function of machine-learning algorithms in developing these materials ranges from optimizing synthesis conditions to predicting crystal structures and physical properties. The advantage of using advance machine learning strategies lies in their ability to make full use of the extensive data available, even failed data, and to speed up the materials research process. Machine learning methods have exhibited a great ability to solve problems such as regression, classification, and clustering. Recently, these approaches have also shown useful performance in materials design and experimental data analysis. Applications in material chemistry will continue to expand and will likely lead to even more significant developments. The application of one particular model is usually limited to one kind of material and one property; therefore, fabricating different models together could develop a unified framework to target the most promising compounds with the balance between several factors. This may also help research produce novel multifunctional materials. The success of applying data science no matter how promising, however, will rely on improved access to open-source and data sharing among researchers as well as the development of more advanced computer science techniques with a particular ability to deal with smaller data sets. There is no doubt that machine learning is revolutionizing chemistry, and we are in a new era for inorganic materials discovery.

Acknowledgements

The authors are grateful for the generous financial support provided by the National Science Foundation through grant No. DMR SSMC 18-47701, DMR CER 19-11311, ENG CMMI 15-62142, the Welch Foundation (grant No. E-1981), a Seed Funding for Advanced Computing (SeFAC) award from the University of Houston Center for Advanced Computing and Data Systems (CACDS). The authors also thank the Hewlett Packard Enterprise Data Science Institute (HPE-DSI) as well as the Research Computing Data Core (RCDC) at the University of Houston for their continued support.

References

1. T. Hey, S. Tansley and K. M. Tolle, *The Fourth Paradigm: Data-Intensive Scientific Discovery*, 2009.
2. A. Belsky, M. Hellenbrandt, V. L. Karen and P. Luksch, *Acta Crystallogr., Sect. B: Struct. Sci.*, 2002, **58**, 364.
3. O. Villars and K. Cenzual, *Pearson's Crystal Data: Crystal Structure Database for Inorganic Compounds*, ASM International, Ohio, USA, 2007.
4. Materials Platform for Data Science (MPDS), https://mpds.io/#start.
5. NIMS Materials Database (MatNavi), https://mits.nims.go.jp/index_en.html.
6. ASM Phase Diagram Database, https://www.asminternational.org/phase-diagrams.
7. G. R. Schleder, A. C. M. Padilha, C. M. Acosta, M. Costa and A. Fazzio, *J. Phys. Mater.*, 2019, **2**, 032001.
8. S. Curtarolo, G. L. W. Hart, M. B. Nardelli, N. Mingo, S. Sanvito and O. Levy, *Nat. Mater.*, 2013, **12**, 191.
9. W. Setyawan, R. M. Gaume, S. Lam, R. S. Feigelson and S. Curtarolo, *ACS Comb. Sci.*, 2011, **13**, 382.
10. A. Seidl, A. Görling, P. Vogl, J. A. Majewski and M. Levy, *Phys. Rev. B*, 1996, **53**, 3764.
11. J. P. Perdew, *Int. J. Quantum Chem.*, 2009, **28**, 497.
12. B. Himmetoglu, A. Floris, S. de Gironcoli and M. Cococcioni, *Int. J. Quantum Chem.*, 2014, **114**, 14.

13. *Materials Genome Initiative for Global Competitiveness*, 2011.
14. S. Kirklin, J. E. Saal, B. Meredig, A. Thompson, J. W. Doak, M. Aykol, S. Rühl and C. Wolverton, *Npj Comput. Mater.*, 2015, **1**, 15010.
15. A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder and K. A. Persson, *APL Mater.*, 2013, **1**, 011002.
16. S. Curtarolo, W. Setyawan, G. L. W. Hart, M. Jahnatek, R. V. Chepulskii, R. H. Taylor, S. Wang, J. Xue, K. Yang, O. Levy, M. J. Mehl, H. T. Stokes, D. O. Demchenko and D. Morgan, *Comput. Mater. Sci.*, 2012, **58**, 218.
17. I. H. Witten, E. Frank, M. A. Hall and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, Cambridge, 2016.
18. L. Chonghe, G. Jin, Q. Pei, C. Ruiliang and C. Nianyi, *J. Phys. Chem. Solids*, 1996, **57**, 1797.
19. H. S. Rao and A. Mukherjee, *Comput. Mater. Sci.*, 1996, **5**, 307.
20. Y. Reich and N. Travitzky, *Mater. Des.*, 1995, **16**, 251.
21. S. P. Ong, W. D. Richards, A. Jain, G. Hautier, M. Kocher, S. Cholia, D. Gunter, V. L. Chevrier, K. A. Persson and G. Ceder, *Comput. Mater. Sci.*, 2013, **68**, 314.
22. S. P. Ong, S. Cholia, A. Jain, M. Brafman, D. Gunter, G. Ceder and K. A. Persson, *Comput. Mater. Sci.*, 2015, **97**, 209.
23. P. Raccuglia, K. C. Elbert, P. D. F. Adler, C. Falk, M. B. Wenny, A. Mollo, M. Zeller, S. A. Friedler, J. Schrier and A. J. Norquist, *Nature*, 2016, **533**, 73.
24. C. Friedman, P. Kra, H. Yu, M. Krauthammer and A. Rzhetsky, *ISMB*, 2001, **17**, S74.
25. S. Eltyeb and N. Salim, *J. Cheminf.*, 2014, **6**, 17.
26. M. C. Swain and J. M. Cole, *J. Chem. Inf. Model.*, 2016, **56**, 1894.
27. R. Leaman, C.-H. Wei and Z. Lu, *J. Cheminf.*, 2015, **7**, S3.
28. M. Krallinger, O. Rabal, A. Lourenço, J. Oyarzabal and A. Valencia, *Chem. Rev.*, 2017, **117**, 7673.
29. S. Spangler, J. N. Myers, I. Stanoi, L. Kato, A. Lelescu, J. J. Labrie, N. Parikh, A. M. Lisewski, L. Donehower, Y. Chen, O. Lichtarge, A. D. Wilkins, B. J. Bachman, M. Nagarajan, T. Dayaram, P. Haas, S. Regenbogen, C. R. Pickering and A. Comer, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and*

Data Mining, ACM Press, New York, New York, USA, 2014, pp. 1877–1886.
30. V. Tshitoyan, J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K. A. Persson, G. Ceder and A. Jain, *Nature*, 2019, **571**, 95.
31. L. Ward, A. Agrawal, A. Choudhary and C. Wolverton, *Npj Comput. Mater.*, 2016, **2**, 16028.
32. K. T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. R. Müller and E. K. U. Gross, *Phys. Rev. B*, 2014, **89**, 205118.
33. L. Ward, R. Liu, A. Krishna, V. I. Hegde, A. Agrawal, A. Choudhary and C. Wolverton, *Phys. Rev. B*, 2017, **96**, 024104.
34. O. Isayev, C. Oses, C. Toher, E. Gossett, S. Curtarolo and A. Tropsha, *Nat. Commun.*, 2017, **8**, 15679.
35. L. Ward, A. Dunn, A. Faghaninia, N. E. R. Zimmermann, S. Bajaj, Q. Wang, J. Montoya, J. Chen, K. Bystrom, M. Dylla, K. Chard, M. Asta, K. A. Persson, G. J. Snyder, I. Foster and A. Jain, *Comput. Mater. Sci.*, 2018, **152**, 60.
36. C. Chen, W. Ye, Y. Zuo, C. Zheng and S. P. Ong, *Chem. Mater.*, 2019, **31**, 3564.
37. F. Faber, A. Lindmaa, O. A. von Lilienfeld and R. Armiento, *Int. J. Quantum Chem.*, 2015, **115**, 1094.
38. L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl and M. Scheffler, *Phys. Rev. Lett.*, 2015, **114**, 105503.
39. K. Fujimura, A. Seko, Y. Koyama, A. Kuwabara, I. Kishida, K. Shitara, C. A. J. Fisher, H. Moriwake and I. Tanaka, *Adv. Energy Mater.*, 2013, **3**, 980.
40. A. Zunger, *Nat. Rev. Chem.*, 2018, **2**, 0121.
41. X. Zhu and X. Wu, *Artif. Intell. Rev.*, 2004, **22**, 177.
42. M. A. Hernández and S. J. Stolfo, *Data Min. Knowl. Discovery*, 1998, **2**, 9.
43. F. Naumann and M. Herschel, *An Introduction to Duplicate Detection*, Morgan & Claypool, 2010.
44. S. B. Kotsiantis, D. Kanellopoulos and P. E. Pintelas, *Int. J. Comput. Sci*, 2006, **1**, 111.
45. H. Liu, F. Hussain, C. L. Tan and M. Dash, *Data Min. Knowl. Disc.*, 2002, **6**, 393.
46. W. Maass, in *Proceedings of the Seventh Annual Conference on Computational Learning Theory*, ACM Press, New York, NY, USA, 1994, pp. 67–75.
47. M. Boulle, *Mach. Learn.*, 2004, **55**, 53.

48. J. Yang and V. Honavar, in *Feature Extraction, Construction and Selection*, Springer US, Boston, MA, 1998, pp. 117–136.
49. H. Vafaie and K. De Jong, in *Proceedings Fourth International Conference on Tools with Artificial Intelligence*, IEEE Computer Society Press, 1992, pp. 200–203.
50. K. Rajan, C. Suh and P. F. Mendez, *Stat. Anal. Data Min.*, 2009, **1**, 361.
51. P. Dey, J. Bible, S. Datta, S. Broderick, J. Jasinski, M. Sunkara, M. Menon and K. Rajan, *Comput. Mater. Sci.*, 2014, **83**, 185.
52. A. Agrawal, P. D. Deshpande, A. Cecen, G. P. Basavarsu, A. N. Choudhary and S. R. Kalidindi, *Integr. Mater. Manuf. Innov.*, 2014, **3**, 90.
53. S. F. Fang, M. P. Wang, W. H. Qi and F. Zheng, *Comput. Mater. Sci.*, 2008, **44**, 647.
54. J.-F. Pei, C.-Z. Cai, Y.-M. Zhu and B. Yan, *Macromol. Theory Simulations*, 2013, **22**, 52.
55. D. W. Aha, D. Kibler and M. K. Albert, *Instance-based Learning Algorithms*, Kluwer Academic Publishers, Boston, 1991.
56. J. R. Quinlan, *Int. J. Man. Mach. Stud.*, 1987, **27**, 221.
57. J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, Cambridge, 2004.
58. C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, 1995.
59. J. Schmidhuber, *Neural Networks*, 2015, **61**, 85.
60. B. Meredig and C. Wolverton, *Chem. Mater.*, 2014, **26**, 1985.
61. Y.-F. Shen, R. Pokharel, T. J. Nizolek, A. Kumar and T. Lookman, *Acta Mater.*, 2019, **170**, 118.
62. J. K. Bunn, J. Hu and J. R. Hattrick-Simpers, *JOM*, 2016, **68**, 2116.
63. A. G. Kusne, T. Gao, A. Mehta, L. Ke, M. C. Nguyen, K. M. Ho, V. Antropov, C. Z. Wang, M. J. Kramer, C. Long and I. Takeuchi, *Sci. Rep.*, 2014, **4**, 6367.
64. A. G. Kusne, D. Keller, A. Anderson, A. Zaban and I. Takeuchi, *Nanotechnology*, 2015, **26**, 444002.
65. J. R. Hattrick-Simpers, J. M. Gregoire and A. G. Kusne, *APL Mater.*, 2016, **4**, 053211.
66. B. Meredig, E. Antono, C. Church, M. Hutchinson, J. Ling, S. Paradiso, B. Blaiszik, I. Foster, B. Gibbons, J. Hattrick-Simpers, A. Mehta and L. Ward, *Mol. Syst. Des. Eng.*, 2018, **3**, 819.
67. S. F. Fang, M. P. Wang and M. Song, *Mater. Des.*, 2009, **30**, 2460.

68. J. Ling, E. Antono, S. Bajaj, S. Paradiso, M. Hutchinson, B. Meredig and B. Gibbons, in *ASME Turbo Expo 2018: Turbomachinery Technical Conference and Exposition*, American Society of Mechanical Engineers, 2018.
69. E. Kim, K. Huang, A. Saunders, A. McCallum, G. Ceder and E. Olivetti, *Chem. Mater.*, 2017, **29**, 9436.
70. Y. F. Han, W. D. Zeng, Y. Shu, Y. G. Zhou and H. Q. Yu, *Comput. Mater. Sci.*, 2011, **50**, 1009.
71. Q. Zhu, M. F. Abbod, J. Talamantes-Silva, C. M. Sellars, D. A. Linkens and J. H. Beynon, *Acta Mater.*, 2003, **51**, 5051.
72. M. F. Abbod, D. A. Linkens, Q. Zhu and M. Mahfouf, *Mater. Sci. Eng., A*, 2002, **333**, 397.
73. L. Pauling, *The Nature of the Chemical Bond*, Cornell University Press, Ithaca, New York, 3rd edn, 1960.
74. Y. Wang, J. Lv, L. Zhu and Y. Ma, *Comput. Phys. Commun.*, 2012, **183**, 2063.
75. C. W. Glass, A. R. Oganov and N. Hansen, *Comput. Phys. Commun.*, 2006, **175**, 713.
76. C. C. Fischer, K. J. Tibbetts, D. Morgan and G. Ceder, *Nat. Mater.*, 2006, **5**, 641.
77. V. M. Goldschmidt, *Trans. Faraday Soc.*, 1929, **25**, 253.
78. P. Villars, K. Brandenburg, M. Berndt, S. LeClair, A. Jackson, Y.-H. Pao, B. Igelnik, M. Oxley, B. Bakshi, P. Chen and S. Iwata, *J. Alloys Compd.*, 2001, **317–318**, 26.
79. J. C. Phillips, *Helv. Phys. Acta*, 1985, **58**, 209.
80. A. Zunger, *Phys. Rev. B*, 1980, **22**, 5839.
81. A. O. Oliynyk, E. Antono, T. D. Sparks, L. Ghadbeigi, M. W. Gaultois, B. Meredig and A. Mar, *Chem. Mater.*, 2016, **28**, 7324.
82. A. O. Oliynyk, L. A. Adutwum, J. J. Harynuk and A. Mar, *Chem. Mater.*, 2016, **28**, 6672.
83. A. O. Oliynyk, L. A. Adutwum, B. W. Rudyk, H. Pisavadia, S. Lotfi, V. Hlukhyy, J. J. Harynuk, A. Mar and J. Brgoch, *J. Am. Chem. Soc.*, 2017, **139**, 17870.
84. P. V. Balachandran, J. Young, T. Lookman and J. M. Rondinelli, *Nat. Commun.*, 2017, **8**, 14282.
85. W. B. Park, J. Chung, J. Jung, K. Sohn, S. P. Singh, M. Pyo, N. Shin and K.-S. Sohn, *IUCrJ*, 2017, **4**, 486.

86. A. Ziletti, D. Kumar, M. Scheffler and L. M. Ghiringhelli, *Nat. Commun.*, 2(**9**, 2775.
87. G. Viswanathan, A. O. Oliynyk, E. Antono, J. Ling, B. Meredig and J. Brgoch, *Inorg. Chem.*, 2019, **58**, 9004.
88. J. Graser, S. K. Kauwe and T. D. Sparks, *Chem. Mater.*, 2018, **30**, 3601.
89. A. O. Oliynyk, M. W. Gaultois, M. Hermus, A. J. Morris, A. Mar and J. Brgoch, *Inorg. Chem.*, 2018, **57**, 7966.
90. K. Ryan, J. Lengyel and M. Shatruk, *J. Am. Chem. Soc.*, 2018, **140**, 10158.
91. T. Yamashita, N. Sato, H. Kino, T. Miyake, K. Tsuda and T. Oguchi, *Phys. Rev. Mater.*, 2018, **2**, 013803.
92. V. L. Deringer, D. M. Proserpio, G. Csányi and C. J. Pickard, *Faraday Discuss.*, 2018, **211**, 45.
93. B. Himmetoglu, A. Floris, S. de Gironcoli and M. Cococcioni, *Int. J. Quantum Chem.*, 2014, **114**, 14.
94. Y. Zhuo, A. Mansouri Tehrani and J. Brgoch, *J. Phys. Chem. Lett.*, 2018, **9**, 1668.
95. Z. Ghahramani, *Nature*, 2015, **521**, 452.
96. D. R. Jones, M. Schonlau and W. J. Welch, *J. Glob. Optim.*, 1998, **13**, 455.
97. E. R. Dougherty, A. Zollanvari and U. M. Braga-Neto, *Curr. Genomics*, 2011, **12**, 333.
98. P. V. Balachandran, D. Xue, J. Theiler, J. Hogden and T. Lookman, *Sci. Rep.*, 2016, **6**, 19660.
99. D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue and T. Lookman, *Nat. Commun.*, 2016, **7**, 11241.
100. F. Faber, A. Lindmaa, O. A. von Lilienfeld and R. Armiento, *Int. J. Quantum Chem.*, 2015, **115**, 1094.
101. A. Seko, H. Hayashi, K. Nakayama, A. Takahashi and I. Tanaka, *Phys. Rev. B*, 2017, **95**, 144110.
102. V. Stanev, C. Oses, A. G. Kusne, E. Rodriguez, J. Paglione, S. Curtarolo and I. Takeuchi, *npj Comput. Mater.*, 2018, **4**, 29.
103. A. Mansouri Tehrani, A. O. Oliynyk, M. Parry, Z. Rizvi, S. Couper, F. Lin, L. Miyagi, T. D. Sparks and J. Brgoch, *J. Am. Chem. Soc.*, 2018, **140**, 9844.
104. Y. Zhuo, A. Mansouri Tehrani, A. O. Oliynyk, A. C. Duke and J. Brgoch, *Nat. Commun.*, 2018, **9**, 4377.
105. B. Settles, *Active Learning Literature Survey*, University of Wisconsin-Madison Department of Computer Sciences, 2009.

106. C. Lemke, M. Budka and B. Gabrys, *Artif. Intell. Rev.*, 2015, **44**, 117.
107. A. Graves, G. Wayne and I. Danihelka, *arXiv Prepr.*, 2014, arXiv:1410.5401.
108. S. J. Pan and Q. Yang, *IEEE Trans. Knowl. Data Eng.*, 2010, **22**, 1345.
109. Y. Duan, M. Andrychowicz, B. Stadie, O. J. Ho, J. Schneider, I. Sutskever, P. Abbeel and W. Zaremba, *Adv. Neural Inf. Process. Syst.*, 2017, 1087.
110. B. M. Lake, R. Salakhutdinov and J. B. Tenenbaum, *Science*, 2015, **350**, 1332.
111. D. Jha, L. Ward, A. Paul, W. Liao, A. Choudhary, C. Wolverton and A. Agrawal, *Sci. Rep.*, 2018, **8**, 17593.
112. R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti and D. Pedreschi, *ACM Comput. Surv.*, 2018, **51**, 93.
113. K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev and A. Walsh, *Nature*, 2018, **559**, 547.
114. E. Gossett, C. Toher, C. Oses, O. Isayev, F. Legrain, F. Rose, E. Zurek, J. Carrete, N. Mingo, A. Tropsha and S. Curtarolo, *Comput. Mater. Sci.*, 2018, **152**, 134.
115. K. A. Persson, B. Waldwick, P. Lazic and G. Ceder, *Phys. Rev. B*, 2012, **85**, 235438.
116. M. W. Gaultois, A. O. Oliynyk, A. Mar, T. D. Sparks, G. J. Mulholland and B. Meredig, *APL Mater.*, 2016, **4**, 053213.

CHAPTER 14

Machine Learning Applications in Chemical Engineering

Y. YAN, T. N. BORHANI AND P. T. CLOUGH*

Energy and Power Theme, Cranfield University, Cranfield, Bedfordshire, MK43 0AL, UK
*Email: p.t.clough@cranfield.ac.uk

14.1 Introduction

Chemical engineering can be broadly defined as the process of converting a raw material/energy into a useful product; no matter how complex the process, everything is designed, optimised, and controlled to ensure economic, safe, and environmentally considerate production. Chemical engineers are commonly associated with the petrochemicals industry, where pipes, tanks, reactors, and pumps are all readily visible, but this is just one application of chemical engineering and there is a great diversity of roles that chemical engineers play. Through the so-called fourth industrial revolution (Industry 4.0 – Internet of Things with integrated autonomous data processing), chemical engineers are being called upon to utilise the vast quantities of data generated by their processes to make the process plant 'smarter' in order to gain a financial advantage; the application of artificial intelligence (AI) is one way of achieving such a task.[1]

AI is being exploited in a greater number of ways across all industries to find useful insights and understanding of data in order to drive up efficiency and process performance and is, therefore, understandably finding

applications in chemical engineering. Recently, companies including BP,[2] ExxonMobil,[3] HSE,[4] and GE[5] have all invested significant sums into AI for chemical engineering applications in order to improve process safety and reliability, identify new connections and work flows, accelerate project lifecycles, and enable rapid use of the latest process data. Within all this investment, we should remember that AI is still only ‘narrow AI’, meaning it is very good at doing one particular task; it is very time-consuming to make representative models that build connections between unseen data or understand the overall plant/process integrations and dependencies. Furthermore, AI systems that have been publicised as being applied to chemical engineering facilities and processes quite often take on the role of data analytics and data insight, finding patterns in data that would take significantly longer for a human to find, rather than operating or optimising the plant's operation.[6] This is not to say that advanced data analytics is less important; in fact, most companies would relish the opportunity to identify that a pump or compressor is close to failure so that it can be replaced, rather than waiting for it to fail, shutting down the plant and then performing a cold start. However, AI in chemical engineering is rarely applied online to give real-time data analytics and process performance insights, which is where the future of AI in chemical engineering could, and should, be heading.

While the current influx of faster, more powerful computing has brought a surge of AI-themed excitement to all industrial sectors, including chemical engineering, AI is actually not that new. In the early days of machine learning (ML) computing (1980s), expert systems (ESs) and later genetic algorithms (GAs) were utilised in the chemical industry to aid with catalyst development and predicting thermophysical fluid properties.[7,8] Over time, artificial neural networks (ANNs) have replaced other forms of AI because of their ability to adapt well to alternative problems, their ability to solve non-linear problems, and the wide availability of adequate training data. Machine learning is an all-encompassing term, so throughout we shall endeavour to keep the reader informed of the varieties and their differences, limitations, and benefits as we broach them.

In this chapter, we explore ML algorithms that have been applied to three areas in chemical engineering: modelling and simulation, process control, and estimation and prediction.

14.2 Modelling and Simulation

ML and AI have wide applications in modelling and simulation of chemical and process engineering.[1,9] Here, we consider important unit operations involved in chemical and process engineering; the application of ML and AI for each of them is reviewed and discussed using selected published studies. It should be noted that in addition to modelling and simulation studies, AI has substantial application in optimisation problems related to chemical and process engineering that can be found in relevant books.[10] The general format of any AI model is illustrated in Figure 14.1.

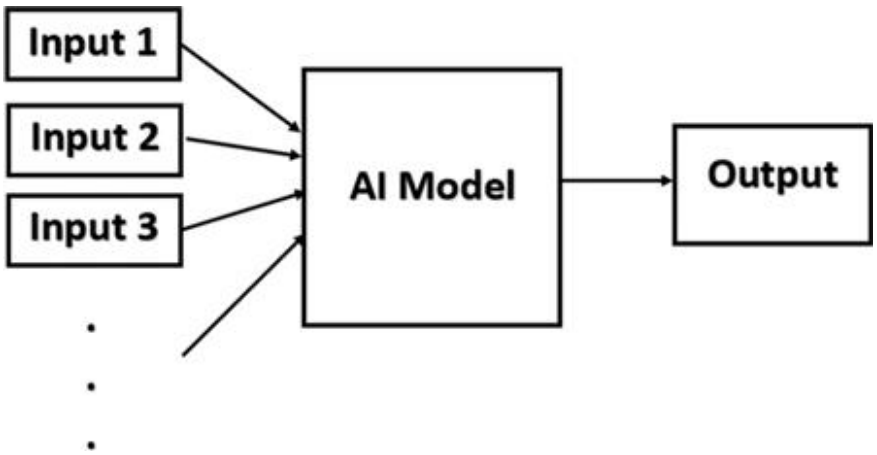


Figure 14.1 General format of AI models.

14.2.1 Application of AI for Modelling and Simulation of Separation Units

14.2.1.1 Distillation Columns

Modelling and simulation of distillation columns require accurate process models that contain governing equations and many relations to predict different parameters and physical properties. A number of distillation process models are available in the published literature, but the complexity of distillation processes has led to a number of assumptions that may limit the universality of the models. Most of the mechanistic models of distillation systems have assumed equilibrium cases for the stages. Such models deviate

from reality, and so these will not give a true representation. To overcome this, the non-equilibrium or rate-based approach has been introduced. This involves chemical reactions and mass transfer equations for each phase, which adds to the complexity of the problem. Furthermore, such mechanistic models are computationally demanding, making them unsuitable for real-time optimisation. To overcome these problems, AI can be utilized to solve complex simulations for different distillation columns or other similar unit operations such as absorption and desorption columns.

Chetouani[11] used a multi-layer perceptron – artificial neural network (MLP-ANN) method to develop a model to simulate the distillation column in steady-state and non-steady-state conditions. Khayet and Cojocaru[12] modelled an air gap membrane distillation process using an ANN. The input variables to the ANN model were air gap thickness, the condensation temperature, the feed inlet temperature, and the feed flow rate of salt aqueous solutions, while the response or output was the performance index, which takes into consideration both the permeate flux and the salt rejection factor. The results obtained from the developed model showed good agreement with experimental data. Ochoa-Estopier *et al.*[13] represented the crude oil distillation column and its integrated heat exchanger network using an ANN. Using the ANN model, the authors determined the operating conditions that enhanced the overall process economics. Osuolale and Zhang[14] used ANN to model and optimise the exergy efficiency in distillation columns. They mentioned that mechanistic modelling of distillation columns for real-time optimisation is not applicable. For such cases the ANN model can be used, as the required computational time for the ANN is very short, which is helpful in real-time optimisation.

14.2.1.2 Absorber and Desorber

Absorption and desorption are very important unit operations that play a vital role in different industries. One of the most well-known examples of these processes is CO_2 removal from different gas streams, such as flue gases, natural gas, and biogas using different types of solvents.[15] Similar to distillation columns, mechanistic modelling of absorption and desorption columns requires complex equation solving with many details.[16,17] Black-box modelling using AI can be beneficial for such a complex system and researchers can predict the required outputs from black-box models with high accuracy more rapidly than using mechanistic modelling.

Some published studies used different AI tools to model absorption and desorption columns and in this section, we review some recent ones. Zhou *et al.*[18] developed four adaptive neuro-fuzzy inference system (ANFIS) models to describe the CO_2 capture process (absorption and desorption) using amines. They considered four different scenarios by changing different parameters as input and output. The ANFIS modelling procedure is illustrated in Figure 14.2.

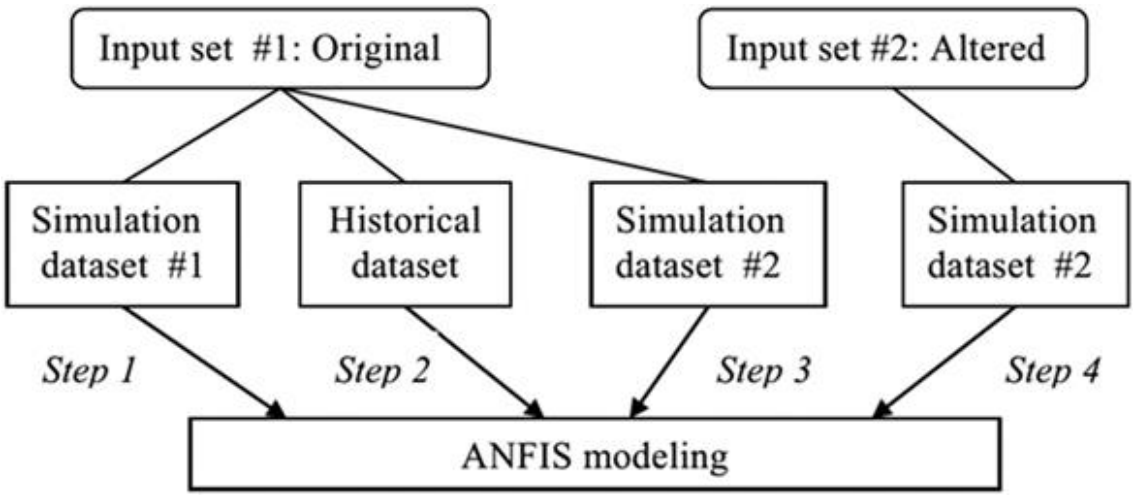


Figure 14.2 ANFIS modelling procedure developed by Zhou *et al.*[18] Reproduced from ref. 18 with permission from Elsevier, Copyright 2013.

The authors reported that the problems presented in scenarios #1 and #2 could not be completely resolved unless changes were made to the definition of the input parameter sets. One important observation is that the parameter of steam flow rate through reboiler (FT-103C) is a critical component of the input parameter set. Hafizi *et al.*[19] compared adaptive neuro-fuzzy and ANN to model the flow rate of rich amine in CO_2 capture absorption columns and reported that the adaptive neuro fuzzy inference systems were more successful in predicting this parameter than the ANN model. Mohammadzadeh *et al.*[20] utilized an ANN to predict sweet gas concentrations in absorption columns that used diethanolamine (DEA) as the solvent. Experimental pilot plant data were used for validation of the model that includes DEA weight per cent in lean amine, concentrations of H_2S, CO_2 and H_2O, temperature, pressure and flow rate of input and output gases and lean amine. The statistical parameters

and error analysis of the study were satisfactory. Cozma *et al.*[21] used airlift reactors to absorb CO_2 physically from biogas using water as the solvent. The authors used two algorithms, a combination of Clonal Selection (CS) and ANN and the new algorithm nCS-MBK; the second algorithm resulted in better model agreement with experimental data. Ahmad *et al.*[22] used ANN to perform the dynamic modelling of absorption columns. The inputs to the network were the absorption or stripping stages and absorption or stripping factors, and the output was the absorption or stripping stage efficiency. Different error analysis and statistical parameters showed the strength of their model.

14.2.1.3 Adsorption

Adsorption is another separation process that has attracted a lot of attention recently, and many studies have conducted research to speed up the commercialisation of this process. The main advantage of adsorption in comparison to absorption is the lower energy consumption. It is recognised that a considerable amount of energy is consumed during the desorption of solute from solvent.[23]

AI has considerable applications in this field. Salehi *et al.*[24] applied Least Squares Support Vector Machines to predict the equilibrium adsorption of copper ions from the water on modified CS/PVA membrane-sorbents. The authors used 72 experimental data points and reported good statistical parameters. Fawzy *et al.*[25] presented an ANFIS model to describe the adsorption of divalent nickel and divalent cadmium ions from aqueous solutions using a biosorbent, which is known as Typha domingensis. By using the ANFIS model the authors recognised the sequence of experimental parameters affecting metal ion removal efficiency. Ghaedi and Vafaei[26] reviewed the application of different AI methods for adsorption of dyes from aqueous solutions. Mazaheri *et al.*[27] used different methods, including AI, to model the adsorption of methylene blue (MB) and divalent cadmium ion from binary aqueous solution by natural walnut carbon. Sensitivity analysis revealed that the adsorbent mass and pH were the critical factors for the removal. Alalm and Nasr[28] adsorbed chlorothalonil, which is a hazardous material, from aqueous solutions using activated carbon and then predicted the adsorption capacity using the ANN model. Using a sensitivity analysis with the developed ANN model, they found that pH was the most significant factor

on adsorption capacity. Mahmoud *et al.*[29] focused on the removal of phosphate ions from aqueous solutions using encapsulated nanoscale zero-valent iron. The authors used an ANN with a structure of 5.7.1 (5 input nodes, 7 hidden layer nodes, and 1 output node) to model this adsorption process and reported a coefficient of determination of 97.6%. Sensitivity analysis of the model demonstrated that pH was the most influential input.

14.2.2 Application of AI for Modelling and Simulation of Reactors

Reactors and their design can be considered as the most specialised unit operation that chemical engineers deal with. Reactors have wide applications, ranging from polymerisation to energy conversion processes. Modelling of reactors is a complex and time-consuming procedure, so researchers often apply black-box modelling using AI methods to model the reactor.

Azarpour *et al.*[30] developed a hybrid model incorporating a first principle model and an ANN to model the industrial fixed-bed catalytic reactors experiencing catalyst deactivation. The elements of the hybrid model are presented in Figure 14.3.

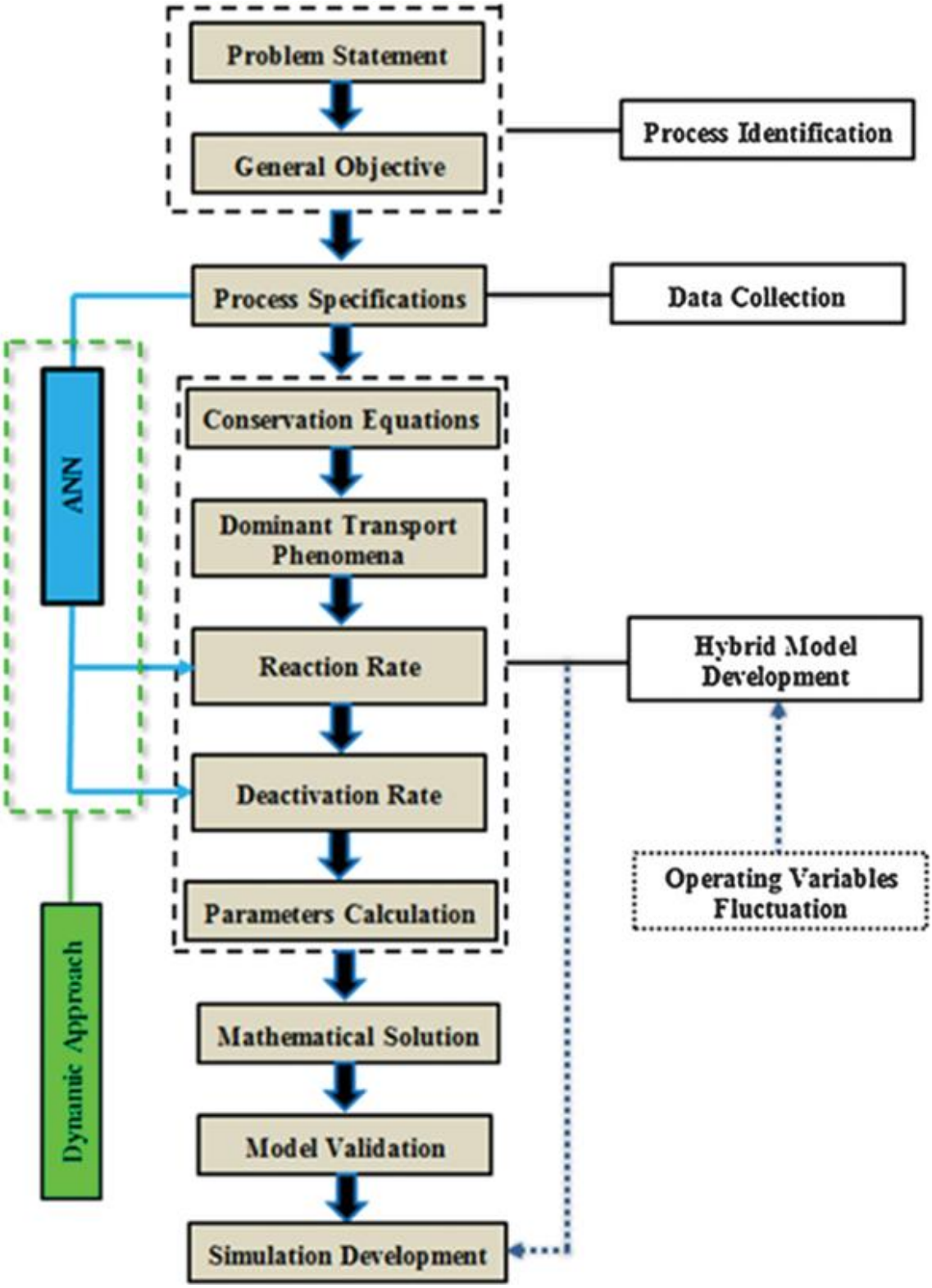

Problem Statement
General Objective
Process Identification
Process Specifications
Data Collection
ANN
Conservation Equations
Dominant Transport Phenomena
Reaction Rate
Hybrid Model Development
Deactivation Rate
Operating Variables Fluctuation
Parameters Calculation
Dynamic Approach
Mathematical Solution
Model Validation
Simulation Development

Figure 14.3 The hybrid model developed for industrial fixed-bed catalytic reactors (FBCRs) experiencing catalyst deactivation.[30] Reproduced from ref. 30 with permission from Elsevier, Copyright 2017.

Joshi and Singhal[31] coupled GA and ANN to optimise the operational condition of zeaxnathin production. The authors evaluated the effects of varying air flow rate and inoculum size for different incubation times. Volumetric power input and energy input to the bubble column were then correlated using the GA-ANN optimised conditions. Zhu *et al.*[32] introduced a deep learning approach to automatically identify tube regions in a pyrolysis reactor from the raw images. Using this approach, the exact temperature and shape on the pyrolysis tubes can be monitored.

14.2.3 Application of AI for Modelling and Simulation of Whole Plants

Mechanistic modelling of a whole chemical plant is a far more complex and challenging task than modelling of a single unit operation. Nevertheless, some studies report using AI to model and analyse whole chemical plants.

A full-scale industrial wastewater treatment process has been modelled using a hybrid ANN model[33] which combines mechanistic and ANN modelling. The authors reported satisfactory results obtained from this hybrid model. Tunckaya and Koklukaya[34] investigated the impact of gas emissions of a real 180 MW coal-fired thermal power plant by means of ANN, autoregressive integrated moving average (ARIMA) and multiple linear regression (MLR) approaches within eight proposed process parameters that affect the stack gas emissions. The ANN model demonstrated better performance than ARIMA and MLR according to statistical parameters and error analysis. Nourani *et al.*[35] utilized different AI methods such as FFNN, ANFIS, SVM, as well as MLR to predict the performance of the Nicosia wastewater treatment plant by means of effluent biological oxygen demand, chemical oxygen demand and total nitrogen. The authors reported that the ANN model was the most successful model.

14.2.4 Application of AI for Modelling and Simulation of Energy Management

Another important area in which AI has been applied recently is energy management and energy systems. Many research and review papers have been published in this field, and therefore, only a limited number of them are reviewed here.

Zahraee *et al.*[36] summarised prior research concerning the use of optimisation in artificially intelligent algorithms for designing and planning and also for controlling problems in the field of hybrid energy systems. The authors reviewed more than one hundred relevant papers. Jha *et al.*[37] illustrated the importance of AI in achieving goals related to renewable energy (RE), such as the development of novel technology for optimum production from available natural resources, environmental awareness, and better management and distribution systems. They reviewed the application of AI in different types of RE such as solar, wind, geothermal, hydro, ocean, bioenergy and hydrogen. Sharifzadeh *et al.*[38] quantified uncertainties in the electricity grid and examined the predictability of their behaviour. The predictive methods considered in their study were ANN, support vector regression (SVR) and Gaussian process regression. All models were able to predict wind and solar power, but only the ANN model was effective for the electricity demand. Considering the dynamics of the electricity grid, it was realized that the prediction process for renewable wind and solar power generation, and electricity demand was fast and accurate enough to effectively replace the alternative electricity storage and standby capacity.

14.3 Control and Operation

If chemical engineering is the science of converting one thing into another, at scale, then control and operation are integral to the economic success of any process. A common overall aim of process control in all chemical engineering sectors is to achieve stable, steady-state operation, where the facility is operating at or close to its designed point of highest efficiency. Often plants will spread across multiple reactors and have seemingly endless piping, with instrumentation dotted at strategic locations in order to gain insight as to how the reactors are performing and to characterise the input and output streams. The greater the complexity and integration of the plant, the more difficult the

process control and operation becomes, which is where new forms of process control are becoming available, notably through machine learning.

14.3.1 The Baseline of Process Control

Proportional, integral, and derivative (PID) controllers are taught to most undergraduate engineering students and are widely recognised as being the standard for process control. PID controllers utilise the difference between a set point variable and the actual process variable value to generate an error, which is utilised to calculate the control signal. Variations of these PID loops can be used in series, where the output from one controller provides the set point for another variable controller, thus taking away some of the lag time between actions and simplifying the complexity of overall process control to individual variable control; this is called cascade control.

If we consider complex processes, with multiple variables (temperature, pressure, flow rate, fluid level, and quality) that each need controlling, but are inherently interlinked through the laws of thermodynamics, then PID controllers no longer become the best solution to use. In these simulations, multivariable control techniques can be utilised, which can decouple the competing influences of interlinked variable controls and thus minimise the effort/cost/energy required to achieve an optimised steady state operation. These multivariable control systems can work well if the interaction between variables is well understood (or non-existent), but will fail if there are slight deviations in process operation. For example, if a steam methane reforming reactor was designed to operate at a steam to carbon ratio of 3, but later someone decides it should be 3.5 then the multivariable controller decoupling would not account for this deviation and would be unable to control the reactor accurately.

A potentially better alternative that can overcome process complexity, variable interconnectivity and interdependence, process-measurement lag time, random disturbances, and minimise the effort/cost/energy of reaching a stable set point, is machine learning control or at least it could have been if Model Predictive Control did not come along.[1]

Model predictive control (MPC) is a very commonly implemented technique of controlling dynamic systems that can be simulated by a linear model while satisfying a set of constraints. MPC uses a model of the process to control dependant variables in the current time step, while calculating and

optimising the future values in a moving prediction horizon that takes into account the past values. This control technique is so widely used because processes are often too complex for PID loops alone and because back in the 1980s it was easier and cheaper to implement an MPC algorithm than a control algorithm using machine learning. However, MPC falls down when processes are dependent upon hundreds of variables or the processes are non-linear, which they regularly are, although some processes can be considered linear over a small range of operating conditions. There is the option of combining multiple MPC's together, which can provide a stable control system if the non-linear system can be broken down into multiple independently linear systems;[39,40] this approach has been demonstrated for application in refineries,[41] and copolymerisation reactors.[42]

14.3.2 Machine Learning Used to Control Chemical Processes

Neural networks were demonstrated a few decades ago to be suitable for controlling complex chemical processes.[43] For example, Savkovic-Stavanovic showed that ANN's could be used to control an azeotrope distillation process and demonstrated the ability to improve process control by time-shifting the model's data inputs, which acts to smooth out the controller.[44] Ramchandram and Rhinehart also utilised an ANN to control a distillation process and found the ANN provided superior control compared with a feedforward PI controller, as it was able to account for the “phenomenological process characteristics”, which a standard PI controller simply has no understanding of.[45] These ANN are commonly applied in parallel with the process where they build up experience and knowledge as they watch and predict the process, see Figure 14.4.

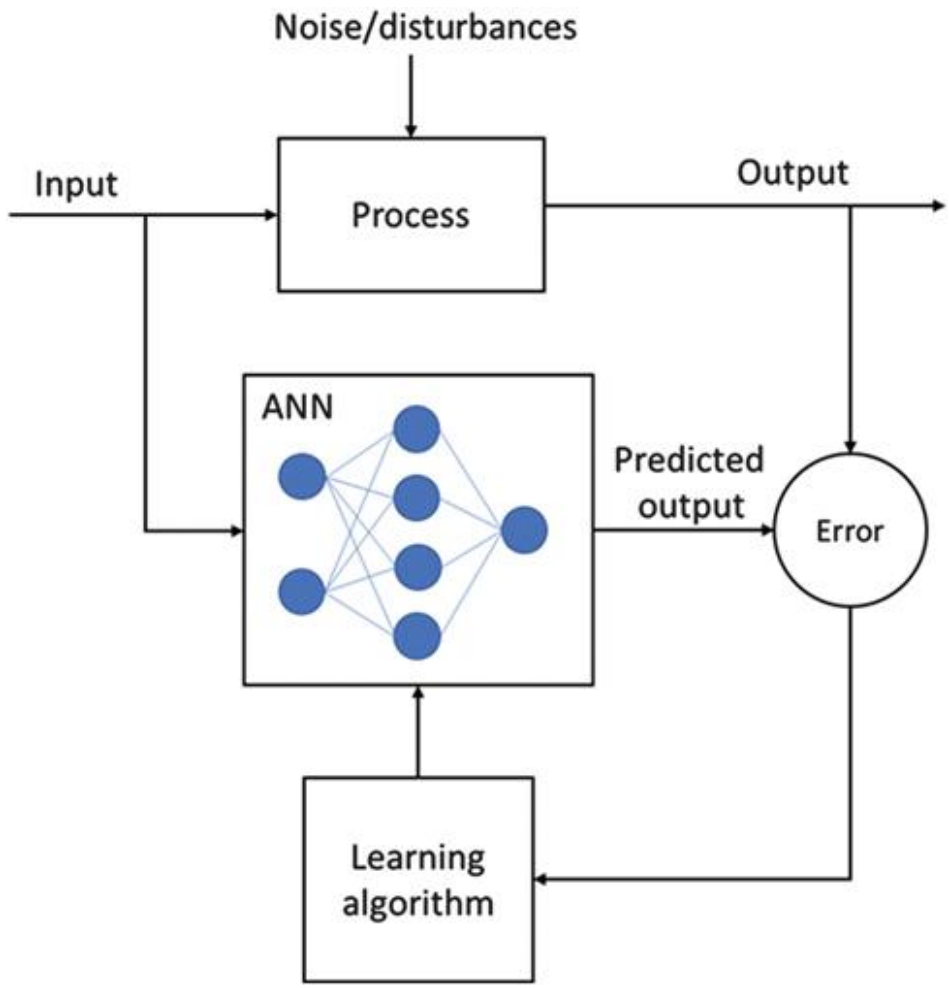


Figure 14.4 *ANN training in parallel with a process using the error of its predictions to increase its confidence in process control.*[46] Reproduced from ref. 46 with permission from Elsevier, Copyright 1992.

ANNs have also been utilised to estimate the liquid levels and compositions of distillation columns, thereby acting as a soft sensor.[47] Soft sensors are often based on ANNs and have been utilised by several researchers to predict unmeasured variables without specific/expensive hardware in order to achieve improved process control,[48] for example in coal-fired power plants,[49] polymerisation processes,[50] ethanol concentration determination in wine production,[51] chemical plants,[52] and some bioprocesses.[53,54] Updating soft sensors online, while a process/reactor is fully operational, is key to the long-term impact of these devices. Work by Lu and Chiang[55] has highlighted a way in which this can be achieved; they utilised two key performance indicators

and applied Hotelling's T^2 multivariate statistic to determine if the T^2 value exceeds some alarm limit where action needed to be taken. The key novelty of this work is that the soft sensors can work at the very edge of where training data exists and system properties are known, and that the soft sensors 'calibration' is updated while online and in operation.[55] A key limitation of soft sensors is that they may only be applicable to an individual particular system or reactor set up and, therefore, are not considered to be compatible with another similar process. Processes are often dependent upon unknown/unmeasured variables or information that cannot be easily programmed into a computer (for example, the shape of a reactor, or the precise locations of sensors and their accuracy) thus making soft sensor coding less portable; however, they are relatively quick and easy to train up if a new system was to implement one.

ANN models can work well; however, it should be noted that they are based on training data and are,therefore, only as good as the training data they are fed with; often termed "garbage in, garbage out". If an ANN has not seen a process state before or if a process variable goes beyond a normal range, then the ANN will apply its mathematics and spit out an answer; whether that answer makes any sense is a different matter. As you can imagine, in a chemical manufacturing facility, it is possible that a thermocouple could randomly dropout, causing an ANN control model to make a corrective action, which could unwittingly cause damage to equipment, the environment, or people. It is analogous to training a model to understand the difference between a cat and a dog by showing it thousands of cat and dog images, but then adding in a photo of a giraffe. The model would have no idea what this patchy long-necked creature is, but will diligently assess the image and give you a prediction of cat or dog, without understanding the image, the context, or knowing that this image lies outside its remit of knowledge. And for any chemical facility this could be very dangerous; understanding the context of a process is vital and it is this kind of knowledge that plant operators build up over many years of experience.[1] A solution could be to give the model decades of data, decades of experience of different scenarios where random alarm trips, sensor failures, blockages *etc.* have occurred, but what if this data doesn't exist, or if it's a new process, a first-of-a-kind reactor, what should be done then? Aside from using data from another similar reactor/process, two options are available: the first is unsupervised learning, where the model itself looks for patterns in the data without any feedback on how it is doing; the

second is reinforcement learning where the model is punished or rewarded such that the model aims to maximise its reward.

Training ANNs in an unsupervised manner means that a training database is not provided; instead, the model can vary parameters itself and build its own training database, this training database can be updated as the system is in operation. A key benefit of this type of training is that we do not introduce our own human biases into the model. As humans, we are fantastic at making algorithms to solve problems. For example, let us consider a simple reactor with air flow through the centre and an external on/off heating control system, where we want to control the reactor's interior temperature at a specific set point value. The objective function of the control programme minimises the amount of ‘heating on’ time in order to save money. A simple control algorithm for this reactor might read:

1. If the reactor is too cold, then turn the heating on and wait for it to heat up
2. If the reactor is too hot, then turn the heating off and wait for it to cool down

We could, of course, train an ANN model to control this reactor, but by defining our reactor's objective function too imprecisely or by not considering all possible scenarios, then the ANN could decide that the best way to minimise the amount of ‘heating on’ time is to not turn the heating on at all.[56] Yes, this would mean that the reactor never heated up and the ANN failed to adhere to one of the objectives but as far as the model is concerned it was successful in saving ‘heating on’ time and the reactor temperature was less important.

Evidently, applying ill-defined objective functions to whole chemical plants could be dangerous, which is why researchers are now exploring ways of not defining the objective function for machine learning chemical reactor control systems and instead allowing the models to build up their own context of the overall process.[57–59] This has been demonstrated to achieve better and safer control than can be achieved by giving the models ‘human knowledge’, as it is difficult to distil every piece of human knowledge into a set of objectives.[60] Deep reinforcement learning has also been applied to a multitude of other simple control situations with high degrees of success but is limited by the amount of time it takes to train the programme.[61] However, by allowing the

programme to self-learn and by reinforcing correct control behaviour, the control algorithm generated is transportable to other control problems. Genetic programming has also been successfully applied in a flow control scenario, whereby the objective function was to minimise the recirculation area of a backwards-facing step by varying the incoming jet of water (80% reduction achieved).[62] A multi-agent control system with self-learning has been applied to a continuous cement production line by breaking down the key points of process control (quality control, decision maker, data analyst, knowledge holder).[57] This way, the overall control system is more resilient to random deviations in a process, but also able to adapt to new control rules that are formed by the machine learning control system.

14.3.3 Traditional Controllers with Integrated Machine Learning

Machine learning controllers are often implemented to provide an input signal to a traditional PID or MPC controller. This input signal could be the predicted value or a predicted error estimate; the additional information is utilised to provide control that is closer to a desired setpoint, see Figure 14.5 for an example of the improvement that can be achieved with machine learning enhanced reactor control.

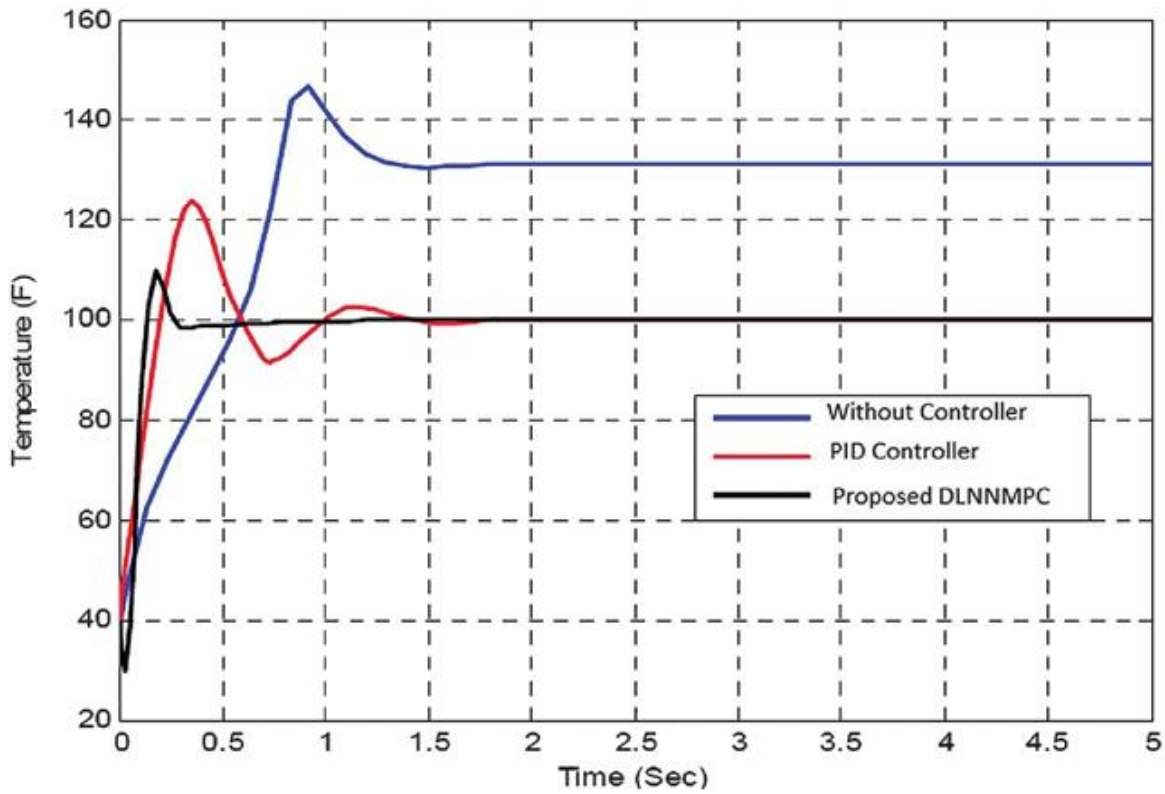


Figure 14.5 Temperature response of a continuously stirred tank reactor utilising different forms of reactor control. The black line indicates the machine learning enhanced controller.[63] Reproduced from ref. 63 with permission from Elsevier, Copyright 2018.

Deepa and Baranilingesan have utilised an MPC with an ANN in the background to benefit from the ANN's ability to learn and store valuable information about a simulated continuously stirred tank reactor. Their research utilised a particle swarm optimisation and gravitation search algorithm to define the initial weights and then tune the ANN to control the process more effectively.[63]

With PID controllers still being commonplace, some researchers have utilised machine learning to optimise the tuning process to achieve improved control, notably Chen and Huang,[64] whose research took a pragmatic approach for using linearised ANNs to control non-linear processes; however, they noted that their model needed to be more flexible to different operating ranges, and noise in the data significantly influenced control. A similar attempt at using ML-enhanced controllers was described by Chowdhury *et al.* who utilised a fuzzy PID controller (a set of rules with relative importance)

and found the fuzzy controller to be able to track a set point value more accurately and reduce the magnitude of deviations from the set point.[65]

The control of a chemical reactor or chemical process is highly dependent on understanding the process in its entirety, which is not always possible due to the size of the reactor and the complexity of the chemistry involved. This is where ML can be utilised to estimate or predict operating parameters.

14.4 Estimation and Prediction

The application of AI, particularly ML, in chemical engineering is not limited to modelling and control, but it can also be implemented as an estimator for difficult-to-measure parameters in chemical process, systems for system optimisation, or predictors to predict the properties of materials or products. In this section, we give a detailed overview of AI and ML algorithms implemented as estimators in chemical process systems, along with their practical applications, advantages, and limitations. It aims to provide a guide to researchers in selecting and developing AI-based estimators or predictors in chemical engineering.

14.4.1 Available AI Algorithms for Estimators in Chemical Engineering

It is well recognised that some important process parameters or properties of products in chemical process systems are difficult or impossible to measure in real-time due to the limitations of physical sensors or process technology. The application of AI or ML algorithms as estimators and predictors in chemical process systems, with the benefits of robustness, easy formulation, simple design, and flexible adaptation capabilities, can solve these issues. Several algorithms such as ANNs, support vector regression, fuzzy logic (FL), GAs, and ESs are used to design the estimators to estimate the difficult-to-measure parameters in chemical process systems.

Recently, a vast amount of research has been conducted by applying AI as estimators in chemical process systems (*e.g.* product compositions, combustion kinetics, temperature, pressure, and heat transfer of chemical reactor). However, it can be difficult to choose a suitable algorithm for estimators due to the variety and unique features of these algorithms and

process complexity. Therefore, an exhaustive survey of each algorithm for estimation and prediction and its applications in chemical process systems is provided in this section, which can guide researchers designing and developing their own estimator in particular chemical process systems (see Table 14.1).

Table 14.1 Some applications of AI algorithms for estimation and prediction in chemical engineering.

Algorithm	Aims	Highlights	References
ANN	Predict the SO_2 emissions from circular fluidised-bed boilers	Able to predict the SO_2 emissions under different conditions	99
ANN	Predict the pyrolysis, gasification, and combustion rates of different solid fuels	Good accuracy with sufficient training data	73, 100–105
ANN	Estimate the CO_2 capture rate	Less than 0.2% error for the closed loop absorber/desorber plant	70, 106
ANN	Estimate the composition of synthesised calcium phosphate powders	Synthesis condition can be optimised through the prediction of compositions of powders	107
ANN	Inferential estimation of polymer quality by bootstrap aggregated ANNs	Bootstrap aggregated ANNs have better prediction accuracy than individual ANNs, and confidence bounds can show the confidence in estimation.	69, 108
ANN	Predict slag viscosity over a	Better prediction performance than any other	109

	broad range of temperatures and slag compositions	mathematic models	
ANN	Prediction of density, porosity, and hardness in aluminium–copper-based composite materials	Satisfactory results can be estimated rather than measured	110
ANN	Modelling of heat transfer coefficient in the furnace	Good agreement with numerical and experimental results	111
ANN	Prediction of wastewater treatment plant performance	Efficient predictions for the high nonlinearity of the biological treatment plant	112
ANN	Vapour–liquid equilibrium predictions	High prediction performance	113
SVR	Prediction of overall gas hold-up in bubble column reactors	97% prediction accuracy with average absolute relative error of 12.11%	114
SVR	Prediction of pipe burst rate	The prediction results from ANN are better than those of SVR.	115
SVR	Air temperature prediction	The SVR models are slightly more accurate than those of the ANNs.	79
SVR	A soft sensor for a batch distillation column	The developed estimator is highly accurate with the varying reflux ratio.	77
FL	Prediction of cement	Low average percentage error –2.69%	81

	compressive strength		
FL	Prediction of soil erosion in a large watershed	Easy to determine soil erosion	116
FL	Prediction of concrete properties	Experimental results agreed well with ANNs and FL	117
FL	Prediction of the product concentration	Suitable for online estimation	118
GA	Multi-parameter estimation in sorption and phase equilibria problems	Effective tool for multi-objective optimisation	119
GA	Biosorption parameter estimation	Capable of accurately finding optimal parameter estimates	95
GA	Estimation in liquid–liquid extraction columns	Provide information about the sensitivity of the solution to the parameters of the model.	91
ES	Fatigue life prediction	Provide moderate and conservative predictions	85
ES	Physical property predictions	Provide consultancy for properties predictions	120
ANN-FL	Optimising operation and improving the efficiency of the CO_2 process system	Hybrid models will simulate the practical process with high accuracy	18
ANN/SVR-GA	Modelling and optimisation of catalyst design	Learn from the historical data with knowledge of process	121

SVR-GA	Prediction of pressure drop	Improve the prediction accuracy	122

ANNs are the most common algorithm applied as an estimator in chemical engineering. Inspired by biological neurons, these can learn complex non-linear and multivariate relationships. The ANN can act as a "black box" that can predict the desired output from a set of input data. In general, an ANN consists of an input layer, one or more hidden layers, and an output layer, with neurons in each layer. It can determine relationships between the input and output from historical data by calculating the network weights and bias, creating a trained ANN that can predict the output from given input data. For estimation and prediction in chemical process systems, several types of ANN have been reported, such as feed forward neural networks, recurrent neural networks and radial basis function neural networks. A detailed comparison and description of different types of ANNs can be accessed in.[66,67] The size and quality of training data and the ANN architecture determine the performance and accuracy of the models. There is no unique method to obtain the optimum number of hidden layers and neurons in each hidden layer; it is common to use trial and error for this. The degree of generalisation that the model can achieve may also be an issue. For instance, if a trained ANN can perfectly fit the data in the training dataset, but its performance on unseen data is poor, then this typically indicates overfitting. Many methods have been reported to improve generalisation and avoid overfitting such as bootstrap aggregated neural networks,[68–70] regularisation[71,72] and cross validation based early stopping.[73–75]

SVR, which is an ML adaptation based on Support Vector Machines, SVM,[76] is also widely used as an estimator in chemical process systems. The SVR "maps the input data into higher dimensional feature space *via* a nonlinear mapping and after a linear regression problem is obtained and solved in this feature space".[76–78] The SVR is generated by the retained kernels, and the data points related to the retained kernels are referred to as the support vectors (SV). The smaller fraction of support vectors leads to a more general solution, which requires less computation to solve for a new and unknown object.[77] In comparison with ANNs, SVR has the following advantages: good generalisation ability of the regression function, robustness of the solution, sparseness of the regression and automatic control of the solution complexity, and extracting the knowledge of the data points.[76,79]

However, the SVR algorithm is limited by both speed and memory constraints of the available hardware when working with massive datasets.[79]

Apart from ANN and SVR, fuzzy logic, GAs, and ESs have been developed as predictors or estimators for different chemical process systems. Fuzzy logic is a type of multi-valued logic that distinguishes more than just TRUE and FALSE values; it can map from a given set of input data to an output based on a set of fuzzy rules.[67,80,81] It includes four components: fuzzification, the fuzzy rule base, a fuzzy output engine, and defuzzification. Fuzzification converts the input and output into linguistic subsets. The fuzzy rule base includes the construction of fuzzy IF-THEN rules based on expert knowledge and/or available data. The fuzzy output engine considers rules in the fuzzy rule base and learns relationships between a set of inputs and outputs. Finally, defuzzification transfers the fuzzy outputs from the output engine to a numeric value.[80,81] Fuzzy logic can relate input to output in a fashion, which is more in-line with human thought than ANNs.[81]

Expert systems are an applied AI approach that use computers to emulate the decision-making ability of human experts. Knowledge-based ESs consist of two subsystems, the knowledge base, which is a collection of facts and rules, and the inference engine, which can predict results based on the contents of the knowledge base.[7,82] Various applications of ES-based estimators to predict different parameters in chemical process systems can be found in ref. 7 and 82–85. Although there are many reports of the use of ES-based estimators in chemical engineering, a reliable ES is time-consuming and expensive to build, and this has limited its adoption.[1]

The Genetic Algorithm was first introduced by Holland in the 1970s.[86] This numerical optimisation algorithm, which is inspired by genetic and evolutionary mechanisms in nature[87,88] involves initial population generation, fitness evaluation and the genetic operations of selection, crossover and mutation.[87] Detailed discussion of GAs can be found in ref. 86–88. Reports[87,89–97] suggest that the GA is an effective tool to estimate parameters in chemical engineering and assist in system optimisation.

As AI-based algorithms are efficient for estimation and prediction in chemical process systems, researchers have combined algorithms together to enhance their model capabilities and solve more complex problems. For instance, ANN and SVM are commonly integrated with intelligent optimisation algorithms such as the GA, simulated annealing and particle

swarm optimisation to improve macroscopic performance prediction.[98] In addition, other combinations of algorithms such as ANN/SVR with FL, ANN/SVR with ES, or FL with ES can also overcome the drawbacks of using a single algorithm for estimation and prediction in chemical process systems.

14.4.2 Selection and Applications of AI-based Estimators

The selection of a suitable algorithm is a key step in developing an AI-based estimation and prediction system in chemical engineering, because it has a significant impact on the prediction accuracy and generalisation ability. As discussed earlier, each algorithm has its own advantages, and therefore, no one algorithm can be suitable for all problems. Table 14.2 briefly compares the advantages and limitations of the different AI algorithms for estimation and prediction in chemical engineering.

Table 14.2 Comparisons of AI algorithms for estimation and prediction in chemical engineering.

Algorithm	Advantages	Limitations
ANN	•Suitable for non-linear systems •Ability to learn from historical data •Do not require a domain expert for learning the relationships between input and output •Trained ANN models are fast	•The performance of ANN models is largely determined by the size and quality of the training dataset. •Trial and error may be required to decide the optimal structure of the ANN.
SVR	•Excellent generalisation capacity with high prediction accuracy •Less sensitive to the noise of inputs	•High computational demand with large training sets.
FL	•Easy to design and implement •Can deal with imprecise and incomplete data •Provide knowledge of the relationship between input and output	•Affected by the rules and membership functions •Lower prediction accuracy
GA	•Provide several feasible	•Time-consuming

	solutions from which the best can be chosen •Do not require knowledge of the problem •Can deal with both discrete and continuous variables	•Sensitive to the input parameters •Complex and difficult to build the GA model
ES	•Able to predict results from detailed knowledge of the process •Fast and easy to exchange knowledge between expert systems	•Costly and complex to establish the knowledge base •Expansion of the knowledge base depends on the availability of experts
Combined algorithms	•May overcome the drawbacks of single algorithms for estimation and prediction •Improve prediction accuracy •Suitable for complex and unique process	•Implementation requires knowledge about different algorithms •Complex and time-consuming to build hybrid models

A simple workflow for developing AI-based estimators in chemical engineering is shown in Figure 14.6; this can provide a basic guideline for researchers in applying AI algorithms in estimation and prediction in chemical engineering.

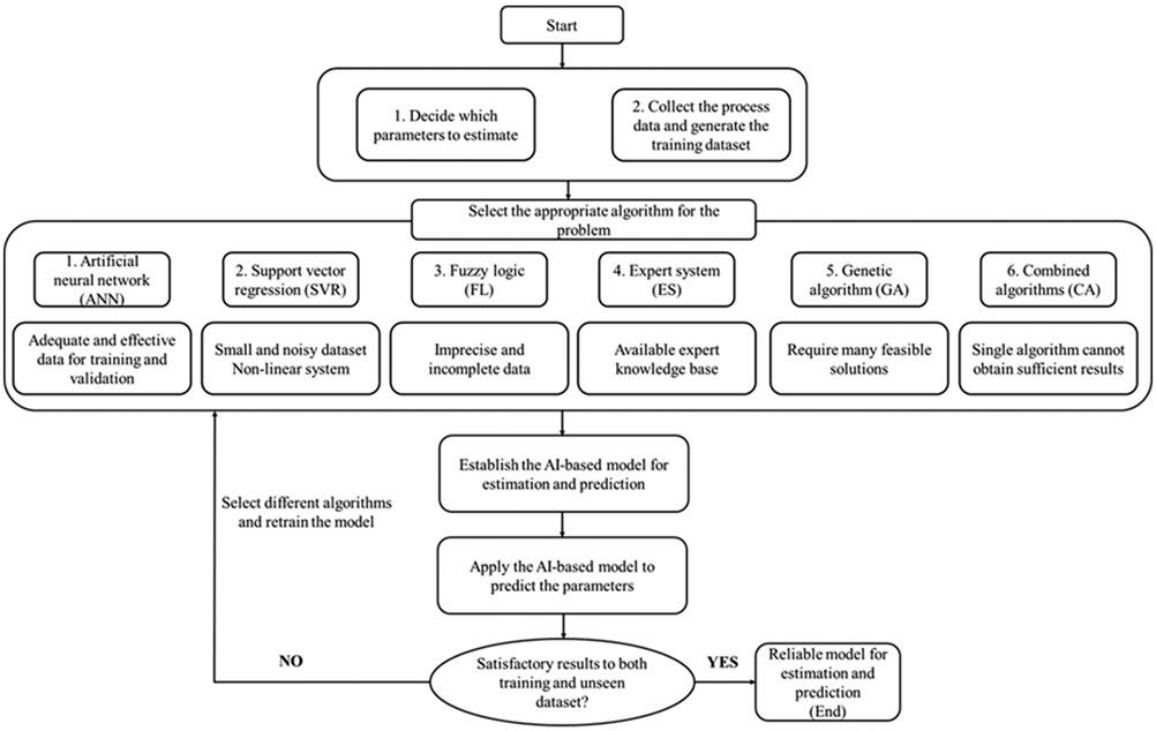


Figure 14.6 Workflow for applying AI algorithms for estimation and prediction in chemical engineering.

Table 14.1 details some of the applications of different AI algorithms that have been applied for estimation and prediction in chemical engineering.

Different properties related to chemical engineering and chemistry can be predicted using AI. Ghiasi and Mohammadi[123] used the least squares support vector machine to model and predict the equilibrium CO_2 absorption in monoethanolamine (MEA), diethanolamine (DEA), and triethanolamine (TEA) aqueous solutions at various alkanolamine concentrations and different temperatures over a wide range of CO_2 partial pressure. They reported a coefficient of determination of 0.98 and the absolute average relative deviation per cent (%AARD) to be less than 6.5% for all studied systems. Norouzbahari *et al.*[124] predicted CO_2 loading using an ANN for a ternary solution of CO_2, Piperazine and H_2O. They collected the required experimental data from available literature, reporting high values of coefficient of determination for training, test and all the datasets. Afkhamipour *et al.*[125] predicted the heat capacity of different amine solutions that are utilized frequently for CO_2 absorption processes, using an ANN. They

considered the heat capacity of amines as a function of system temperature, concentration of the solvents, CO_2 loading, and the molecular weight of solvent. The authors reported that AARD between ANN model results and experimental data of CP for 47 systems of amine solutions studied was 4.3%. For conventional amines, the AARDs for an ANN model and a thermodynamic model in comparison with experimental data were 0.59% and 0.57%, respectively. Borhani *et al.*[23] used experimental data of partial pressure of CO_2 in MEA solution and also k-value data extracted from Aspen Plus and predicted these properties using the GA in combination with MLR and then utilized this equation in process modelling of a rotating packed bed desorber column.

An important recent application of AI is chemometric studies. Quantitative structure–property/activity relationships, proposed and developed by Hansch, Fujita and co-workers,[126,127] express links between molecular structure features (descriptors) of materials and their properties/activities. The modelling (regression) approaches in QSPR/QSAR studies differ from linear techniques like MLR, partial least-regression, and principal component regression[128] and such nonlinear ML techniques ANNs, genetic programming, SVM, and adaptive neuro fuzzy inference systems.[129,130] In QSPR studies, especially when using MLR, different types of classical algorithms such as stepwise forward selection, or evolutionary or metaheuristic algorithms such as GA, PSO, SA, and ant colony have been used in descriptor selection to reduce the number of descriptors and keep the most influential ones in the prediction of the property under study.[131] Different type of properties can be estimated using this method.[132] The developed models can be used in other types of modelling; a general flowchart of QSAR/QSPR methods is illustrated in Figure 14.7.

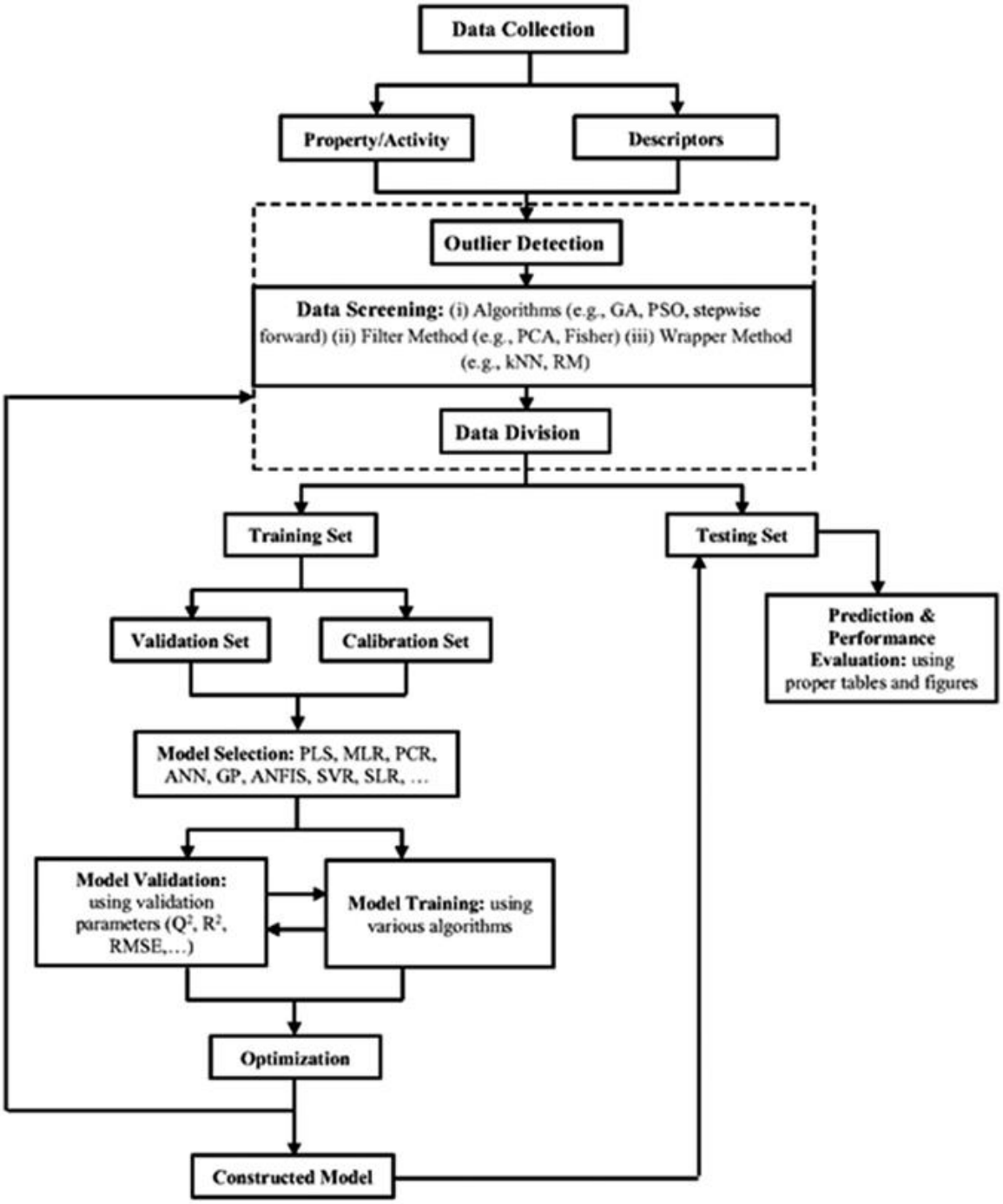


Figure 14.7 A flowchart of the QSPR/QSAR model development.

14.5 Future Trends and Outlook

Materials development, process estimation, and fault prediction are three areas that are set to be the easiest win areas for AI in the coming years.[1] Deep and Convoluted Neural Networks are the favoured forms of ANN's at present; however, hybrid models where multiple types of AI are utilised together or where AI is combined with first-principles knowledge are being explored in greater detail. It is not impossible to imagine that, despite the current interest in neural networks and AI in general, AI might slip out of investors' focus again (note the previous two 'AI winters'[133]) or that the currently favoured ANN's are replaced by a new type of AI that promises greater things. For chemical engineering at least, there is a lot of untapped potential to be explored with AI and, no doubt, some interesting developments lie ahead.

One final aspect of AI that is beginning to change is the explicability of the AI model itself. It is no longer acceptable by the general public, stakeholders, or process plant managers that AI is a black-box that spits out an answer with no explanation; this degrades trust in the model and its results. Therefore, AI models in the future will need to be able to explain their decisions and thought-processing steps. AI interpretation methods are available including LBP,[134] SHAP,[135] DeepLIFT,[136] and LIME.[137] This explicability should not be underestimated as an easy accomplishment to achieve, AI models are not set up (or rather the underlying mathematics is not set up) to output thought-processing steps; even the AlphaGo programme which defeated the world champion in the game Go, played moves which were unseen before and seemingly wrong but ended up aiding it to beat its human counterpart.[138] For the application of ML in chemical engineering, it is essential that we must produce explainable and interpretable models, such that a controller, modeller, or operator can understand how it has reached its decisions so that they can be validated, checked, and confirmed. All this needs to happen before investors, and companies are willing to fund the widespread implementation of AI in Chemical Engineering applications, but when they do, they can expect to reduce operating and maintenance costs and improve process safety and reliability.

References

1. V. Venkatasubramanian, The promise of artificial intelligence in chemical engineering: Is it here, finally?, *AIChE J.*, 2019, **65**, 466–478.

2. A. Duckett, BP invests in AI to accelerate upstream projects - News - The Chemical Engineer, https://www.thechemicalengineer.com/news/bp-invests-in-ai-to-accelerate-upstream-projects/, (accessed 18 September 2019).
3. A. Jasi, Exxon partners with IBM to advance quantum computing - News - The Chemical Engineer, https://www.thechemicalengineer.com/news/exxon-partners-with-ibm-to-advance-quantum-computing/, (accessed 18 September 2019).
4. A. Duckett, Initiative seeks to unlock potential of health and safety data - News - The Chemical Engineer, https://www.thechemicalengineer.com/news/initiative-seeks-to-unlock-potential-of-health-and-safety-data/, (accessed 18 September 2019).
5. T. Kellner, *GE And BP Will Expand The Digital Oilfield - GE Reports*, https://www.ge.com/reports/post/123572457345/deep-machine-learning-ge-and-bp-will-connect-2/, (accessed 18 September 2019).
6. J. Koumoutsakis, Data is Best - Features - The Chemical Engineer, https://www.thechemicalengineer.com/features/data-is-best/, (accessed 18 September 2019).
7. R. Bañares-Alcántara, A. W. Westerberg and M. D. Rychener, Development of an expert system for physical property predictions, *Comput. Chem. Eng.*, 1985, **9**, 127–142.
8. R. Bañares-Alcántara, A. W. Westerberg, E. I. Ko and M. D. Rychener, Decade—A hybrid expert system for catalyst selection—I. Expert system consideration, *Comput. Chem. Eng.*, 1987, **11**, 265–277.
9. J. Valadi and P. Siarry, *Applications of Metaheuristics in Process Engineering*, Springer International Publishing, 2014.
10. G. P. Rangaiah, *Multi-objective Optimization in Chemical Engineering: Developments and Applications*, John Wiley & Sons, 2013.
11. Y. Chetouani, Using Artificial Neural networks for the modelling of a distillation column, *Int. J. Comput. Sci. Appl.*, 2007, **4**, 119–133.
12. M. Khayet and C. Cojocaru, Artificial neural network modeling and optimization of desalination by air gap membrane distillation, *Sep. Purif. Technol.*, 2012, **86**, 171–182.
13. L. M. Ochoa-Estopier, M. Jobson and R. Smith, Operational optimization of crude oil distillation systems using artificial neural networks, *Comput. Chem. Eng.*, 2013, **59**, 178–185.

14. F. N. Osuolale and J. Zhang, Energy efficiency optimisation for distillation column using artificial neural network models, *Energy*, 2016, **106**, 562–578.
15. T. N. Borhani and M. Wang, Role of solvents in CO_2 capture processes: the review of selection and design methods, *Renewable Sustainable Energy Rev.*, 2019, **114**, 109299.
16. T. N. Borhani, A. Azarpour, V. Akbari, S. R. Wan Alwi and Z. A. Manan, CO_2 capture with potassium carbonate solutions: A state-of-the-art review, *Int. J. Greenhouse Gas Control*, 2015, **41**, 142–162.
17. T. N. Borhani, E. Oko and M. Wang, Process modelling and analysis of intensified CO_2 capture using monoethanolamine (MEA) in rotating packed bed absorber, *J. Cleaner Prod.*, 2018, DOI: 10.1016/j.jclepro.2018.09.089.
18. Q. Zhou, C. W. Chan, P. Tontiwachwuthikul, R. Idem and D. Gelowitz, Application of neuro-fuzzy modeling technique for operational problem solving in a CO_2 capture process system, *Int. J. Greenhouse Gas Control*, 2013, **15**, 32–41.
19. A. Hafizi, M. Koolivand-Salooki, A. Janghorbani, A. Ahmadpour and M. H. Moradi, An Investigation of Artificial Intelligence Methodologies in the Prediction of the Dirty Amine Flow Rate of a Gas Sweetening Absorption Column, *Pet. Sci. Technol.*, 2014, **32**, 527–534.
20. S. Mohammadzadeh, H. Zargari and M. A. Ghayyem, The Application of Intelligent Computation (Artificial Neural Network—ANN) Prediction of Sweet Gas Concentration in a Gas Absorption Column, *Energy Sources, Part A*, 2015, **37**, 485–493.
21. P. Cozma, E. N. Drăgoi, I. Mămăligă, S. Curteanu, W. Wukovits, A. Friedl and M. Gavrilescu, Modelling and optimization of CO_2 absorption in pneumatic contactors using artificial neural networks developed with clonal selection-based algorithm, *Int. J. Nonlinear Sci. Numer. Simul.*, 2015, **16**, 97–110.
22. Z. Ahmad, J. Zhang, T. Kashiwao and A. Bahadori, Prediction of absorption and stripping factors in natural gas processing industries using feedforward artificial neural network, *Pet. Sci. Technol.*, 2016, **34**, 105–113.
23. T. N. Borhani, E. Oko and M. Wang, Process modelling, validation and analysis of rotating packed bed stripper in the context of intensified CO_2

capture with MEA, *J. Ind. Eng. Chem.*, 2019, DOI: 10.1016/j.jiec.2019.03.040.

24. E. Salehi, J. Abdi and M. H. Aliei, Assessment of Cu(II) adsorption from water on modified membrane adsorbents using LS-SVM intelligent approach, *J. Saudi Chem. Soc.*, 2016, **20**, 213–219.
25. M. Fawzy, M. Nasr, S. Adel, H. Nagy and S. Helmi, Environmental approach and artificial intelligence for Ni(II) and Cd(II) biosorption from aqueous solution using Typha domingensis biomass, *Ecol. Eng.*, 2016, **95**, 743–752.
26. A. M. Ghaedi and A. Vafaei, Applications of artificial neural networks for adsorption removal of dyes from aqueous solution: A review, *Adv. Colloid Interface Sci.*, 2017, **245**, 20–39.
27. H. Mazaheri, M. Ghaedi, M. H. Ahmadi Azqhandi and A. Asfaram, Application of machine/statistical learning{,} artificial intelligence and statistical experimental design for the modeling and optimization of methylene blue and Cd(II) removal from a binary aqueous solution by natural walnut carbon, *Phys. Chem. Chem. Phys.*, 2017, **19**, 11299–11317.
28. M. G. Alalm and M. Nasr, Artificial intelligence, regression model, and cost estimation for removal of chlorothalonil pesticide by activated carbon prepared from casuarina charcoal, *Sustainable Environ. Res.*, 2018, **28**, 101–110.
29. A. S. Mahmoud, M. K. Mostafa and M. Nasr, Regression model, artificial intelligence, and cost estimation for phosphate adsorption using encapsulated nanoscale zero-valent iron, *Sep. Sci. Technol.*, 2019, **54**, 13–26.
30. A. Azarpour, T. N. G. Borhani, S. R. Wan Alwi, Z. A. Manan and M. I. Abdul Mutalib, A generic hybrid model development for process analysis of industrial fixed-bed catalytic reactors, *Chem. Eng. Res. Des.*, 2017, **117**, 149–167.
31. C. Joshi and R. S. Singhal, Zeaxanthin production by Paracoccus zeaxanthinifaciens ATCC 21588 in a lab-scale bubble column reactor: Artificial intelligence modelling for determination of optimal operational parameters and energy requirements, *Korean J. Chem. Eng.*, 2018, **35**, 195–203.
32. W. Zhu, Y. Ma, M. G. Benton, J. A. Romagnoli and Y. Zhan, Deep learning for pyrolysis reactor monitoring: From thermal imaging toward

smart monitoring system, *AIChE J.*, 2019, **65**, 582–591.
33. D. S. Lee, C. O. Jeon, J. M. Park and K. S. Chang, Hybrid neural network modeling of a full-scale industrial wastewater treatment process, *Biotechnol. Bioeng.*, 2002, **78**, 670–682.
34. Y. Tunckaya and E. Koklukaya, Comparative analysis and prediction study for effluent gas emissions in a coal-fired thermal power plant using artificial intelligence and statistical tools, *J. Energy Inst.*, 2015, **88**, 118–125.
35. V. Nourani, G. Elkiran and S. I. Abba, Wastewater treatment plant performance analysis using artificial intelligence – an ensemble approach, *Water Sci. Technol.*, 2018, **78**, 2064–2076.
36. S. M. Zahraee, M. K. Assadi and R. Saidur, Application of Artificial Intelligence Methods for Hybrid Energy System Optimization, *Renewable Sustainable Energy Rev.*, 2016, **66**, 617–630.
37. S. K. Jha, J. Bilalovic, A. Jha, N. Patel and H. Zhang, Renewable energy: Present research and future scope of Artificial Intelligence, *Renewable Sustainable Energy Rev.*, 2017, **77**, 297–317.
38. M. Sharifzadeh, A. Sikinioti-Lock and N. Shah, Machine-learning methods for integrated renewable power generation: A comparative study of artificial neural networks, support vector regression, and Gaussian Process Regression, *Renewable Sustainable Energy Rev.*, 2019, **108**, 513–538.
39. D. Dougherty and D. Cooper, A practical multiple model adaptive strategy for single-loop MPC, *Control Eng. Pract.*, 2003, **11**, 141–159.
40. A. Abdelkarim, K. Jouili and N. B. Braiek, *Studies in Computational Intelligence*, Springer, Cham, **vol. 635**, 2016, pp. 655–668.
41. C. R. Porfırio, E. Almeida Neto and D. Odloak, Multi-model predictive control of an industrial C3/C4 splitter, *Control Eng. Pract.*, 2003, **11**, 765–779.
42. L. Özkan, M. V. Kothare and C. Georgakis, Control of a solution copolymerization reactor using multi-model predictive control, *Chem. Eng. Sci.*, 2003, **58**, 1207–1221.
43. N. Bhat and T. J. McAvoy, Use of neural nets for dynamic modeling and control of chemical process systems, *Comput. Chem. Eng.*, 1990, **14**, 573–582.
44. J. Savkovic-Stevanovic, Neural net controller by inverse modeling for a distillation plant, *Comput. Chem. Eng.*, 1996, **20**, S925–S930.

45. S. Ramchandran and R. R. Rhinehart, A very simple structure for neural network control of distillation, *J. Process Control*, 1995, **5**, 115–128.
46. K. J. Hunt, D. Sbarbaro, R. Żbikowski and P. J. Gawthrop, Neural networks for control systems—A survey, *Automatica*, 1992, **28**, 1083–1112.
47. A. Rani, V. Singh and J. R. P. Gupta, Development of soft sensor for neural network based control of distillation column, *ISA Trans.*, 2013, **52**, 438–449.
48. L. Fortuna, *Soft Sensors for Monitoring and Control of Industrial Processes*, Springer, 2007.
49. X. Liang, Y. Li, X. Wu, J. Shen and L. Pan, in *2018 IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, 2018, pp. 1002–1007.
50. J. C. B. Gonzaga, L. A. Meleiro, C. Kiang and R. Maciel Filho, ANN-based soft-sensor for real-time process monitoring and control of an industrial polymerization process, *Comput. Chem. Eng.*, 2009, **33**, 43–49.
51. D. Osorio, J. Ricardo Pérez-Correa, E. Agosin and M. Cabrera, Soft-sensor for on-line estimation of ethanol concentrations in wine stills, *J. Food Eng.*, 2008, **87**, 571–577.
52. H. Kaneko and K. Funatsu, Adaptive soft sensor based on online support vector regression and Bayesian ensemble learning for various states in chemical plants, *Chemom. Intell. Lab. Syst.*, 2014, **137**, 57–66.
53. P. Sagmeister, P. Wechselberger, M. Jazini, A. Meitz, T. Langemann and C. Herwig, Soft sensor assisted dynamic bioprocess control: Efficient tools for bioprocess development, *Chem. Eng. Sci.*, 2013, **96**, 190–198.
54. E. C. Rivera, D. I. P. Atala, F. M. Filho, A. Carvalho da Costa and R. M. Filho, Development of real-time state estimators for reaction–separation processes: A continuous flash fermentation as a study case, *Chem. Eng. Process. Process Intensif.*, 2010, **49**, 402–409.
55. B. Lu and L. Chiang, Semi-supervised online soft sensor maintenance experiences in the chemical industry, *J. Process Control*, 2018, **67**, 23–34.
56. D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman and D. Mané, Concrete Problems in AI Safety, arXiv 2016, Retrieved from arXiv:1606.06565.

57. N. Sahebjamnia, R. Tavakkoli-Moghaddam and N. Ghorbani, Designing a fuzzy Q-learning multi-agent quality control system for a continuous chemical production line – A case study, *Comput. Ind. Eng.*, 2016, **93**, 215–226.
58. B. Recht, A Tour of Reinforcement Learning: The View from Continuous Control, *Annu. Rev. Control. Robot. Auton. Syst.*, 2019, **2**, 253–279.
59. S. P. K. Spielberg, R. B. Gopaluni and P. D. Loewen, in *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, IEEE, 2017, pp. 201–206.
60. K. R. Varshney and H. Alemzadeh, On the Safety of Machine Learning: Cyber-Physical Systems, Decision Sciences, and Data Products, *Big Data*, 2017, **5**, 246–255.
61. T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and D. Wierstra, Continuous control with deep reinforcement learning, arXiv, Retrieved from arXiv:1509.02971, 2015.
62. N. Gautier, J.-L. Aider, T. Duriez, B. R. Noack, M. Segond and M. Abel, Closed-loop separation control using machine learning, *J. Fluid Mech.*, 2015, **770**, 442–457.
63. S. N. Deepa and I. Baranilingesan, Optimized deep learning neural network predictive controller for continuous stirred tank reactor, *Comput. Electr. Eng.*, 2018, **71**, 782–797.
64. J. Chen and T.-C. Huang, Applying neural networks to on-line updated PID controllers for nonlinear process control, *J. Process Control*, 2004, **14**, 211–230.
65. J. I. Chowdhury, D. Thornhill, P. Soulatiantork, Y. Hu, N. Balta-Ozkan, L. Varga and B. K. Nguyen, Control of Supercritical Organic Rankine Cycle based Waste Heat Recovery System Using Conventional and Fuzzy Self-tuned PID Controllers, *Int. J. Control. Autom. Syst.*, 2019, 1–13.
66. I. N. da Silva, D. H. Spatti, R. A. Flauzino, L. H. B. Liboni and S. F. dos Reis Alves, *Artificial neural networks: A practical course*, Springer, 2016.
67. J. Mohd Ali, M. A. Hussain, M. O. Tade and J. Zhang, Artificial Intelligence techniques applied as estimator in chemical process systems - A literature survey, *Expert Syst. Appl.*, 2015, **42**, 5915–5931.
68. J. Zhang, Developing robust non-linear models through bootstrap aggregated neural networks, *Neurocomputing*, 1999, **25**, 93–113.

69. J. Zhang, E. B. Martin', A. J. Morris and C. Kiparissides, Inferential Estimation of Polymer Quality Using Stacked Neural Networks, *Comput. Chem. Eng.*, 1997, **vol. 21**, S1025–S1030.
70. F. Li, J. Zhang, E. Oko and M. Wang, Modelling of a post-combustion CO_2 capture process using neural networks, *Fuel*, 2015, **151**, 156–163.
71. K. Hagiwara and K. Kuno, Regularization learning and early stopping in linear networks, *Proceedings of the International Joint Conference on Neural Networks*, IEEE, **vol. 4**, 2000, pp. 511–516.
72. S. Mc Loone and G. Irwin, Improving neural network training solutions using regularisation, *Neurocomputing*, 2001, **37**, 71–90.
73. K. M. T. Q. Zhu, J. M. Jones and A. Williams, The predictions of coal-char combustion rate using an artificial neural, *Fuel*, 1999, **78**, 1755–1762.
74. J. Behler, Neural network potential-energy surfaces for atomistic simulations, *Chem. Modell.*, 2010, 1–41.
75. P. M. Granitto, P. F. Verdes and H. A. Ceccatto, Neural network ensembles: Evaluation of aggregation algorithms, *Artif. Intell.*, 2005, **163**, 139–162.
76. K. Desai, Y. Badhe, S. S. Tambe and B. D. Kulkarni, Soft-sensor development for fed-batch bioreactors using support vector regression, *Biochem. Eng. J.*, 2006, **27**, 225–239.
77. P. Jain, I. Rahman and B. D. Kulkarni, Development of a soft sensor for a batch distillation column using support vector regression techniques, *Chem. Eng. Res. Des.*, 2007, **85**, 283–287.
78. E. Skouras, S. Pavlov, H. Bendella and D. N. Angelov, Support Vector Method for Function Approximation, Regression Estimation and Signal Processing, *Advances in Neural Information Processing Systems*, 1997.
79. R. F. Chevalier, G. Hoogenboom, R. W. McClendon and J. A. Paz, Support vector regression with reduced training sets for air temperature prediction: A comparison with artificial neural networks, *Neural Comput. Appl.*, 2011, **20**, 151–159.
80. M. Özger and Z. Şen, Prediction of wave parameters by using fuzzy logic approach, *Ocean Eng.*, 2007, **34**, 460–469.
81. S. Akkurt, G. Tayfur and S. Can, Fuzzy logic model for the prediction of cement compressive strength, *Cem. Concr. Res.*, 2004, **34**, 1429–1433.

82. X. Ju, F. Wang, Y. Liu and L. Cai, Application on expert system to predict the effect of fracturing measures, *Advanced Materials Research*, **vol. 616–618**, 2013, pp. 1033–1037.
83. J. Gasteiger, M. G. Hutchings, B. Christoph, L. Gann, C. Hiller, P. Löw, M. Marsili, H. Saller and K. Yuki, A new treatment of chemical reactivity: Development of EROS, an expert system for reaction prediction and synthesis design, *Top. Curr. Chem.*, 1987, **137**, 19–73.
84. R. Gupta, M. A. Kewalramani and A. Goel, Prediction of Concrete Strength Using Neural-Expert System, *J. Mater. Civ. Eng.*, 2006, **18**, 462–466.
85. Y. H. Kim, J. H. Song and J. H. Park, An expert system for fatigue life prediction under variable loading, *Expert Syst. Appl.*, 2009, **36**, 4996–5008.
86. J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, 1992.
87. M. Mansour, A Genetic Algorithm identification technique for the estimation of process derivatives and model parameters in on-line optimization, *5th International Conference on Systems and Control (ICSC)*, IEEE, 2016, pp. 120–125.
88. D. E. Goldberg and J. H. Holland, Genetic Algorithms and Machine Learning, *Mach. Learn.*, 1988, **3**, 95–99.
89. R. B. Kasat and S. K. Gupta, Multi-objective optimization of an industrial fluidized-bed catalytic cracking unit (FCCU) using genetic algorithm (GA) with the jumping genes operator, *Comput. Chem. Eng.*, 2003, **27**, 1785–1800.
90. S. V. Inamdar, S. K. Gupta and D. N. Saraf, Multi-objective Optimization of an Industrial Crude Distillation Unit Using the Elitist Non-Dominated Sorting Genetic Algorithm, *Chem. Eng. Res. Des.*, 2004, **82**, 611–623.
91. A. Hasseine, A. H. Meniai, M. Korichi, M. B. Lehocine and H. J. Bart, A genetic algorithm based approach to coalescence parameters: Estimation in liquid-liquid extraction columns, *Chem. Eng. Technol.*, 2006, **29**, 1416–1423.
92. S. K. Gupta and M. Ramteke, Applications of genetic algorithms in chemical engineering II: Case studies, *Applications of Metaheuristics in Process Engineering*, Springer International Publishing, Cham, 2014, pp. 61–87.

93. M. J. Azarhoosh, H. A. Ebrahim and S. H. Pourtarah, Simulating and Optimizing Auto-Thermal Reforming of Methane to Synthesis Gas Using a Non-Dominated Sorting Genetic Algorithm II Method, *Chem. Eng. Commun.*, 2016, **203**, 53–63.
94. V. H. Alvarez, R. Larico, Y. Ianos and M. Aznar, Parameter estimation for VLE calculation by global minimization: the genetic algorithm, *Braz. J. Chem. Eng.*, 2008, **25**, 409–418.
95. K. H. Chu, X. Feng, E. Y. Kim and Y.-T. Hung, Biosorption Parameter Estimation with Genetic Algorithm, *Water*, 2011, **3**, 177–195.
96. K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.*, 2002, **6**, 182–197.
97. M. C. A. F. Rezende, C. B. B. Costa, A. C. Costa, M. R. W. Maciel and R. M. Filho, Optimization of a large scale industrial reactor by genetic algorithms, *Chem. Eng. Sci.*, 2008, **63**, 330–341.
98. Y. Liu, T. Zhao, W. Ju and S. Shi, Materials discovery and design using machine learning, *J. Mater.*, 2017, **3**, 159–177.
99. J. Krzywanski, T. Czakiert, A. Blaszczuk, R. Rajczyk, W. Muskala and W. Nowak, A generalized model of SO_2 emissions from large- and small-scale CFB boilers by artificial neural network approach Part 2. SO_2 emissions from large- and pilot-scale CFB boilers in O_2/N_2, O_2/CO_2 and O_2/RFG combustion atmospheres, *Fuel Process. Technol.*, 2015, **139**, 73–85.
100. T. Abbas, M. M. Awais and F. C. Lockwood, An artificial intelligence treatment of devolatilization for pulverized coal and biomass in co-fired flames, *Combust. Flame*, 2003, **132**, 305–318.
101. Z. Yildiz, H. Uzun, S. Ceylan and Y. Topcu, Application of artificial neural networks to co-combustion of hazelnut husk-lignite coal blends, *Bioresour. Technol.*, 2016, **200**, 42–47.
102. S. Sunphorka, B. Chalermsinsuwan and P. Piumsomboon, Artificial neural network model for the prediction of kinetic parameters of biomass pyrolysis from its constituents, *Fuel*, 2017, **193**, 142–158.
103. K. Luo, J. Xing, Y. Bai and J. Fan, Prediction of product distributions in coal devolatilization by an artificial neural network model, *Combust. Flame*, 2018, **193**, 283–294.

104. Ö. Çepelioğullar, İ. Mutlu, S. Yaman and H. Haykiri-Acma, A study to predict pyrolytic behaviors of refuse-derived fuel (RDF): Artificial neural network application, *J. Anal. Appl. Pyrolysis*, 2016, **122**, 84–94.
105. R. Mikulandrić, D. Lončar, D. Böhning, R. Böhme and M. Beckmann, Artificial neural network modelling approach for a biomass gasification process in fixed bed gasifiers, *Energy Convers. Manage.*, 2014, **87**, 1210–1223.
106. N. Sipöcz, F. A. Tobiesen and M. Assadi, The use of Artificial Neural Network models for CO_2 capture plants, *Appl. Energy*, 2011, **88**, 2368–2376.
107. M. Asadi-Eydivand, M. Solati-Hashjin, A. Farzadi and N. A. A. Osman, Artificial neural network approach to estimate the composition of chemically synthesized biphasic calcium phosphate powders, *Ceram. Int.*, 2014, **40**, 12439–12448.
108. J. Zhang, Inferential estimation of polymer quality using bootstrap aggregated neural networks, *Neural Networks*, 1999, **12**, 927–938.
109. M. A. Duchesne, A. MacChi, D. Y. Lu, R. W. Hughes, D. McCalden and E. J. Anthony, in *Fuel Processing Technology*, Elsevier, **vol. 91**, 2010, pp. 831–836.
110. A. M. Hassan, A. Alrashdan, M. T. Hayajneh and A. T. Mayyas, Prediction of density, porosity and hardness in aluminum-copper-based composite materials using artificial neural network, *J. Mater. Process. Technol.*, 2009, **209**, 894–899.
111. J. Krzywanski and W. Nowak, Modeling of heat transfer coefficient in the furnace of CFB boilers by artificial neural network approach, *Int. J. Heat Mass Transf.*, 2012, **55**, 4246–4253.
112. F. S. Mjalli, S. Al-Asheh and H. E. Alfadala, Use of artificial neural network black-box modeling for the prediction of wastewater treatment plants performance, *J. Environ. Manage.*, 2007, **83**, 329–338.
113. R. Sharma, D. Singhal, R. Ghosh and A. Dwivedi, Potential applications of artificial neural networks to thermodynamics: vapor–liquid equilibrium predictions, *Comput. Chem. Eng.*, 1999, **23**, 385–390.
114. A. B. Gandhi, J. B. Joshi, V. K. Jayaraman and B. D. Kulkarni, Development of support vector regression (SVR)-based correlation for prediction of overall gas hold-up in bubble column reactors for various gas-liquid systems, *Chem. Eng. Sci.*, 2007, **62**, 7078–7089.

115. A. Shirzad, M. Tabesh and R. Farmani, A comparison between performanc support vector regression and artificial neural network in prediction of pipe burst rate in water distribution networks, *KSCE J. Civ. Eng.*, 2014, **18**, 941–948.
116. B. Mitra, H. D. Scott, J. C. Dixon and J. M. McKimmey, Applications of fuzzy logic to the prediction of soil erosion in a large watershed, *Geoderma*, 1998, **86**, 183–209.
117. I. B. Topçu and M. Saridemir, Prediction of rubberized concrete properties using artificial neural network and fuzzy logic, *Constr. Build. Mater.*, 2008, **22**, 532–540.
118. P. R. Patnaik, Application of fuzzy logic for state estimation of a microbial fermentation with dual inhibition and variable product kinetics, *Food Bioprod. Process. Trans. Inst. Chem. Eng. Part C*, 1997, **75**, 239–246.
119. P. K. Kundu, A. Elkamel, F. M. Vargas and M. U. Farooq, Genetic algorithm for multi-parameter estimation in sorption and phase equilibria problems, *Chem. Eng. Commun.*, 2018, **205**, 338–349.
120. R. Bañares-Alcántara, A. W. Westerberg and M. D. Rychener, Development of an expert system for physical property predictions, *Comput. Chem. Eng.*, 1985, **9**, 127–142.
121. S. Nandi, Y. Badhe, J. Lonari, U. Sridevi, B. S. Rao, S. S. Tambe and B. D. Kulkarni, Hybrid process modeling and optimization strategies integrating neural networks/support vector regression and genetic algorithms: study of benzene isopropylation on Hbeta catalyst, *Chem. Eng. J.*, 2004, **97**, 115–129.
122. S. K. Lahiri and K. C. Ghanta, Prediction of Pressure Drop of Slurry Flow in Pipeline by Hybrid Support Vector Regression and Genetic Algorithm Model, *Chinese J. Chem. Eng.*, 2008, **16**, 841–848.
123. M. M. Ghiasi and A. H. Mohammadi, Rigorous modeling of CO_2 equilibrium absorption in MEA, DEA, and TEA aqueous solutions, *J. Nat. Gas Sci. Eng.*, 2014, **18**, 39–46.
124. S. Norouzbahari, S. Shahhosseini and A. Ghaemi, Modeling of CO_2 loading in aqueous solutions of piperazine: Application of an enhanced artificial neural network algorithm, *J. Nat. Gas Sci. Eng.*, 2015, **24**, 18–25.

125. M. Afkhamipour, M. Mofarahi, T. N. G. Borhani and M. Zanganeh, Prediction of heat capacity of amine solutions using artificial neural network and thermodynamic models for CO_2 capture processes, *Heat Mass Transf.*, 2018, DOI: 10.1007/s00231-017-2189-y.
126. C. Hansch, P. P. Maloney, T. Fujita and R. M. Muir, Correlation of Biological Activity of Phenoxyacetic Acids with Hammett Substituent Constants and Partition Coefficients, *Nature*, 1962, **194**, 178–180.
127. T. Fujita, J. Iwasa and C. Hansch, A New Substituent Constant, π, Derived from Partition Coefficients, *J. Am. Chem. Soc.*, 1964, **86**, 5175–5180.
128. T. N. Borhani, S. García-Muñoz, C. Vanesa Luciani, A. Galindo and C. S. Adjiman, Hybrid QSPR models for the prediction of the free energy of solvation of organic solute/solvent pairs, *Phys. Chem. Chem. Phys.*, 2019, **21**, 13706–13720.
129. T. N. G. Borhani, A. Afzali and M. Bagheri, QSPR estimation of the auto-ignition temperature for pure hydrocarbons, *Process Saf. Environ. Prot.*, 2016, DOI: 10.1016/j.psep.2016.07.004.
130. T. N. G. Borhani, M. Saniedanesh, M. Bagheri and J. S. Lim, QSPR prediction of the hydroxyl radical rate constant of water contaminants, *Water Res.*, 2016, **98**, 344–353.
131. M. Bagheri, T. N. G. Borhani, A. H. Gandomi and Z. A. Manan, A simple modelling approach for prediction of standard state real gas entropy of pure materials, *SAR QSAR Environ. Res.*, 2014, **25**, 695–710.
132. A. R. Katritzky, M. Kuanar, S. Slavov, C. D. Hall, M. Karelson, I. Kahn and D. A. Dobchev, Quantitative Correlation of Physical and Chemical Properties with Chemical Structure: Utility for Prediction, *Chem. Rev.*, 2010, **110**, 5714–5789.
133. A. I. Is Another, Winter Coming?, *Thomas Nield*, https://hackernoon.com/is-another-ai-winter-coming-ac552669e58c, (accessed 19 September 2019).
134. J. Kauffmann, M. Esders, G. Montavon, W. Samek and K.-R. Müller, From clustering to cluster explanations via neural networks, arXiv:1906.07633, 2019.
135. S. M. Lundberg and S.-I. Lee, A Unified Approach to Interpreting Model Predictions Scott, *Adv. Neural Inf. Process. Syst.*, 2017, 4765–4774.

136. A. Shrikumar, P. Greenside and A. Kundaje, Learning important features through propagating activation differences, *34th International Conference on Machine Learning, ICML*, 2017, pp. 3145–3153.
137. M. T. Ribeiro, S. Singh and C. Guestrin, “Why should I trust you?” Explaining the predictions of any classifier, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16*, ACM Press, New York, USA, 2016, pp. 1135–1144.
138. How Google's AI Viewed the Move No Human Could Understand WIRED, https://www.wired.com/2016/03/googles-ai-viewed-move-no-human-understand/, (accessed 19 September 2019).

CHAPTER 15

Representation Learning in Chemistry

JOSHUA STAKER(0000-0001-6301-5574),*[a] GABRIEL MARQUES(0000-0002-4882-563X)[b] AND J. DAKKA[b]

[a] Schrödinger, Inc.101 SW Main Street, Portland, Oregon 97204, United States
[b] Schrödinger, Inc.120 West 45th Street, New York, New York 10036, United States
*Email: joshua.staker@schrodinger.com

15.1 Introduction

The applications of chemistry are vast. Nearly everything we interact with and consume, from the molecules that make up our medicine or the electronics that power technology, to the materials in our shoes and eyewear, are the result of chemistry. Advances made in chemistry have widespread implications toward generally improved quality of life worldwide. A significant part of working in the field of chemistry is leveraging technology to both increase the rate at which we can design and improve the quality of developed molecules and materials. In many applications, technology is used to build models and simulations of molecules to predict, *in silico*, how molecules are expected to interact and

perform in the real world. The faster and more accurate our models are, the more effective our discovery efforts become.

There are many varieties of models for chemistry, including machine learning models, and those that are mathematically derived from physics. Rigorous physical calculations, such as *ab initio* methods for estimating the electronic properties in molecules, are difficult to outperform in terms of accuracy.[1,2] However, rigorous methods can be expensive to calculate and cost-prohibitive to run over large datasets or chemical systems containing many atoms. Machine learning (ML) is often computationally less expensive relative to physics-based calculations and can be made quite accurate when flexible models are trained with sufficient data.[1] Consequently, ML is useful for estimating the properties of large numbers of molecules.

Many ML approaches in chemistry utilize molecules as inputs, though other features describing the chemical environment can be used instead, or included in addition to the molecules. There is one difficulty frequently encountered when using molecules in ML: most machine learning algorithms cannot operate over standard molecular formats, *e.g.*, SMILES or connection tables, and methods often first require molecules to be described in a form understandable by the algorithm. We call this new form its "representation" and the process by which this form is obtained as "featurization." Unfortunately, many ML algorithms require representations that are fixed-length vectors, which is not a format well suited for the description of molecules in an unambiguous way.

One way to describe molecules as fixed-length vectors is by calculating various properties and descriptors for the molecules and then arranging the calculated outputs into "feature vectors." In this approach, it is difficult to know which features to calculate, and the choice of features is highly dependent on the endpoint of interest. Another approach is to calculate molecular fingerprints wherein the presence of substructures is determined and then encoded into fixed-length binary vectors. However, binary fingerprints can experience bit collisions with single bits encoding multiple substructures, decreasing the exactness of the representation. Both molecular fingerprints and calculated feature-vectors are coarse-grained representations that irreversibly discard information about the molecules,

forcing learning algorithms to rely on incomplete representations of molecules. Additionally, these types of “non-learned” featurizations rely on significant preprocessing external to the predictive model.

Alternatively, flexible ML models capable of accepting more complex data structures as input provide the opportunity to learn over more complete representations of molecules. Having access to more complete information enables learning algorithms to automate the featurization process and leverage data to learn features that are relevant to the task. Recent neural network architectures support learning over more complex inputs, including graphs, 3D grids, and variable-length sequences, which are formats more suitable for representing molecules. Learning representations not only simplify the featurization process by operating on minimally processed data but can lead to more accurate models.

Herein, we make the distinction between learned and non-learned representations as follows: Non-learned representations are derived from molecules using heuristics or physics with the outputs arranged into feature vectors, while learned representations of molecules are intermediate, latent vectors produced by transformations learned directly from data by an ML model. Our definition of learned *versus* non-learned is not necessarily strict or rigorous from a computational or mathematical standpoint, but it is useful in describing and understanding the various methods that fall within each category and the general advantages and limitations of each.

Recent advances in deep learning have renewed interest across industries in utilizing vast amounts of data to learn the featurization process, in contrast to relying on pre-defined heuristics for feature generation. With the significant amount of chemical data currently available in the literature, public repositories, and internally generated in commercial companies, there is now greater opportunity for scientists to leverage more flexible models. Consequently, deep learning is becoming more popular in chemistry as an approach to learn representations and build more powerful models. Although both learned and non-learned representations can be used in further downstream modeling, including in neural networks, the ability to learn over more complete molecular representations is an important distinction—for many endpoints, such as protein–ligand binding, it is very difficult to construct feature vectors that contain sufficient information to make reliable predictions.

In this chapter, we describe conventional, non-learned featurization methods that are often used in chemistry, along with their several limitations, then discuss how deep learning can help circumvent those limitations. Next, we explore the motivation behind learned representations, namely, how typical molecular data formats, *e.g.*, SMILES or connection tables, can be transformed using a learned set of parameters into a new n-dimensional space that is useful for various forms of analysis and model building. And, finally, we provide overviews of several families of deep learning architectures used to obtain learned representations. It is impossible to review all relevant representation learning approaches recently seen in the literature, even those that relate directly to chemistry. Instead, we provide a high-level overview of several families of architectures as an entry point for understanding, implementing, and improving molecular representations.

Our discussion is specifically interested in the conversion of minimally processed input data into latent forms, and we encourage the reader to explore the literature for details regarding other aspects of model-building, including optimization, the various hyperparameters, and other implementation specifics. Representation learning for molecules is still relatively new, and much work is still needed to find more general-purpose representations. Throughout our discussion, we also cover some of the needs, challenges, and opportunities in developing better representations, especially in the context of the underlying physics.

15.2 Non-learned Molecular Representations

Often, the data available to practitioners in chemical science include a collection of molecules in a standard format such as SMILES, SDF, or connection table, along with some property of interest. One use of such a dataset is to train a model given the molecules and labels, then use the trained model to make property predictions for new molecules that are not previously seen by the model. Common ML methods used in chemistry often rely on relatively simple models, for example, logistic regression or multilayer perceptrons (MLPs), which require fixed-length vectors of

features as input. An immediate question that arises before building such a model is: how can one take a collection of molecules and convert them into a format useable by the model? Fortunately, molecules can be converted into fixed-length vectors through several forms of featurization, each with trade-offs, though choosing which features to generate during featurization is not straightforward and requires making assumptions about the types of patterns in molecules expected to correlate with the endpoint. Additionally, many features are expensive to compute; furthermore, it may be non-obvious how to obtain them, such as how to properly characterize specific interactions between proteins and ligands relevant to binding prediction.

From the 1950's onward, molecular featurization has centered around building quantitative structure–activity or property relationship (QSAR/QSPR) models[3] and has historically been performed through non-learned transformations. Conventional non-learned featurization methods include hashing molecular fragments into “fingerprints” or calculating various physical properties arranged into “feature-vectors.” Molecular features can be simple, such as molecular weight, or obtained through complex physics-based calculations. Examples of commonly used properties include logP, quantum mechanical calculations, synthetic feasibility, number of rotatable bonds, number of chiral centers, and many others. It is worth noting that many molecular properties used as inputs to machine learning models are themselves approximations, *e.g.*, logP or DFT energies; although one could approximate these input properties using the learned featurizations discussed later, they have historically been obtained either using relatively simple models or through physics-based calculations. We encourage the reader to review Cherkasov *et al.*[3] for a more comprehensive discussion around common forms of featurization and a history of their development in relation to QSAR/QSPR.

Topological molecular fingerprints are another form of featurization with wide adoption.[4] Fingerprints are time-tested in the literature and continue to be used as comparative baselines in contemporary deep learning methodologies.[5,6] Other forms of non-learned featurization exist, but topological fingerprints and calculated properties are most prevalent and are briefly described in the following subsections.

15.2.1 Molecular Fingerprints

Not every molecule is the same size; indeed, molecules can be small, with just a few atoms, or quite large such as natural products or polypeptides. Molecular fingerprint generation concerns the encoding of 2D or 3D molecular graphs most often into fixed-length, binary strings of bits with "on" bits indicating the presence of a particular substructure, while "off" bits indicate the lack of such substructure. Fingerprint lengths are fixed, regardless of molecule size, providing a suitable form of representation for use in ML.

There is not a single, standard form of fingerprint, and several varieties have been developed. The overarching principle shared across approaches relevant to machine learning is the encoding of molecular fragments into fixed-length vectors. Cereto-Massagué *et al.* describe various forms of fingerprints and how they are produced.[7]

Primary differences between methods concern how substructures are detected and which substructures get encoded in the on-bits. For example, the ECFP method considers atom neighborhoods radially outward from central atoms[8] while MACCS takes SMARTS patterns provided by the user to define the substructure matching rules.[9] The total number of possible substructures is vast. Rather than encoding each substructure independently into an extremely long, sparse vector, substructures can be hashed, constraining the vector into a fixed length, with the length specified by the user. Hashing substructures provides a convenient way to encode molecules into fixed-length vectors of specific lengths regardless of the size, variety, and the number of molecules to be modeled. While longer fingerprints decrease the risk of bit collisions (where single bits encode multiple substructures), the vectors become very sparse, which not all model types handle well, and computational cost almost invariably increases.

Fixed-length fingerprint vectors are well suited to be used directly in ML without the need to make specific assumptions around which properties should be calculated and included as features. Another benefit of fingerprints is their interpretability. Although it is not possible to reconstruct original molecular graphs from fingerprints, the on-bits indicate which substructures are present in the molecule and can be visualized and examined.

Fingerprints have proven extremely useful and are frequently adopted in the literature.[6] However, there remain some endpoints that are difficult to model using fingerprints. For example, protein–ligand binding involves the interaction of molecules with proteins in 3D space, and protein information is not typically included in molecular fingerprints. Also, fingerprints discard overall graph connectivity, *i.e.*, they contain information about which substructures are present but not how they are connected to form the whole molecule, making long-range interactions and stereochemistry more difficult to model. Another disadvantage is that fingerprints sometimes need to be very long, and are consequently extremely sparse, to encode large numbers of different substructures; these are less efficient and expressive than distributed representations.[10,11] They also ignore many internal degrees of freedom, such as the 3D spatial-relationships between atoms, and do not contain enough information to account for dynamics.

In summary, fingerprints are easy to calculate, have lengths that are fixed and independent of molecule size, are binary and sparse (enabling lightweight storage), and are interpretable. Various forms of fingerprints exist which can be selected and evaluated empirically during experimentation. Nevertheless, these benefits come at the cost of potential bit collisions and the discarding of overall graph connectivity.

15.2.2 Computable Features

Another time-tested approach to feature generation for molecular modeling is the calculation of molecular properties, either through heuristics or physics-based calculations. Desired features are first identified and calculated, then arranged into vectors for training a model. The variety of useful molecular properties is extensive. For example, Ahneman *et al.*[12] used 120 different calculated features, including energetics, electrostatics, vibrational frequency, and molecular shape. Some software packages can generate thousands of descriptors.[13,14] The specific features and the number of different features used are part of the modeling design process and are highly dependent on the endpoint type. Some properties are theoretically motivated, such as quantum mechanical approximations, while others are purely empirical or practically defined, *e.g.*, synthetic accessibility or Lipinski rules for drug-likeness.[15] Topological descriptors containing

information about shape, surface area, volume, and flexibility can also be useful, and even basic physical properties, like molecular weight, may provide a valuable signal. Apart from calculated or heuristically obtained features, properties approximated through independently built models can be used as input features to other models.[16]

Some features are based on energetics and can contain information about molecular orbitals and charge density.[17] Similarly, HOMO and LUMO energies can be calculated and used,[18] as well as other properties approximated through DFT and other levels of theory in quantum mechanical calculations.[17,19,20] There is some confidence gained when physics is used in models. Utilizing physically derived features may help encourage downstream approximations to follow the physics more closely. Additionally, models relying on physics may improve as our understanding and implementation of physics improves.

The examples mentioned here are just a small sample of the many such features and approaches found in the literature, and detailed resources containing descriptions of features and their uses have been published elsewhere.[13,21–23] We encourage the reader to identify and consider which specific properties may be useful for the task at hand.

Computable features, especially those calculated using rigorous physics or simulations, are immensely useful, and the expansive breadth of features grants much flexibility in selecting which to use. Flexibility, in combination with high-quality features, enables researchers to build more robust models with smaller datasets. It is not always clear, however, which features to choose, and the selection process often takes significant expertise and experimentation to determine. In the case of physics-based calculations, the quality of the generated features depends on the correctness of the implementations, which are sometimes bound to a certain level of theory, where higher levels are more costly to compute. Many features are also themselves generated from other models or approximations and provided through black-box implementations in software. Empirical, externally provided features are difficult to validate and may be unreliable outside a specific region of chemical space. The determination of which features to use, the assumptions made regarding applicable chemical space in feature-

generation implementations, and the cost to compute physics-based features, make feature-learning an attractive prospect.

15.3 The Need for Learned Representations

Although non-learned featurizations are powerful and have held up well in the literature, they often do not capture important patterns in the data, discarding information necessary to build robust predictive models. Molecular properties can also be expensive to procure, *e.g.*, quantum mechanical calculations. When using the non-learned features described so far, common questions may arise:

- Which features should be generated for the data?
- What patterns in the data should features capture?
- Should all features be used, or only a selection, and if only a selection, which ones?
- Do features sufficiently encode the input or is useful signal unintentionally discarded?
- Should expensive quantum mechanical calculations be included, or can they be avoided?
- How can we be certain features correlate with the output and not just add noise?

Questions like these were primary motivators behind the last decade of breakthroughs in the ML literature, especially within the subdomain of deep learning using neural networks. A primary goal in deep learning is to learn the feature extraction process as part of the model, instead of relying on external preprocessing for feature generation. Compared with traditional approaches, incorporating featurization as part of the model-building process unlocks the ability to learn the featurization in a data-driven manner, granting significantly improved model flexibility at reduced engineering complexity and potentially lower total computational cost.

Learned featurization provides several attractive benefits. Firstly, learning featurization makes it less imperative for researchers to spend engineering effort to generate and validate a variety of features, or for

practitioners to have deep expertise knowing which features to use and their implications in model building. Additionally, features that would have been costly to compute may no longer be required. A most significant benefit, however, is that data-driven featurizations can produce more informative representations, leading to improved model performance.

Perhaps, the most well-known example of feature-learning success is the convolutional neural network applied to images. Early in this last decade's renaissance in deep learning, Krizhevsky *et al.* introduced the convolutional architecture known as “AlexNet” which showed excellent performance over a challenging image dataset using only minimal preprocessing.[24] Just a few years later, convolutional models were shown to exceed human performance on that same image dataset.[25] In the last decade, deep learning has transformed the world's approaches toward image recognition, game playing, language, audio generation, and many other problems in a variety of domains. More general advancements such as attention,[26] generative modeling,[27] and reinforcement learning[28] are not restricted to specific problem-spaces and provide fresh, new approaches to tackling wide varieties of challenging problems. At the heart of nearly all of these methods is deep learning and its ability to transform relatively raw data into learned representations.

One can imagine how difficult it would be to design a featurization scheme by hand to describe the many nuances in images with all their variations in color, lighting, object shapes, and input quality. Even more difficult to describe by hand are problems requiring the combination of multiple input data modalities or the generation of highly complex output. Some relatively extreme examples include transcribing video into text[29] or animating still images.[30] Accomplishing any one of these tasks programmatically, or coming up with features by hand that sufficiently describe the input is overwhelming and virtually impossible. While these (and virtually all) deep learning applications are still very narrow—only performing well within the same data distribution seen during training—they remain powerful for many applications. One can appreciate similarly complex and highly non-linear tasks in chemistry such as protein–ligand binding prediction, quantum mechanical energy estimation, devising synthesis routes, and many others.

It is also possible to visualize and analyze learned representations (often called latent representations). Gómez-Bombarelli *et al.* performed dimensionality reduction and visualized latent representations of molecules in $\mathbb{R}^2$ space, and intuitive patterns emerged.[31] Latent representations of data points can be compared using any number of similarity metrics to estimate the distance between points in latent space. When conducting analysis or drawing conclusions using learned representations, it is important to remember, however, that the latent space is empirical and a function of both the data and the learning algorithm. Distances are virtually guaranteed to change when retraining the model with different parameters or data. Even with this important caveat, exploring latent representations can be intuitively pleasing and may hint at how a model is learning.

Learned representations are also powerful in the context of generative models. Generative models can use latent representations to produce new data not seen during training.[32] In addition to generating new examples, Berthelot *et al.*[33] used a variational autoencoder to smoothly interpolate between one image and another.

Learning representations, instead of relying on rigorous feature engineering and selection, can both simplify the model-building process and lead to more expressive models. However, these benefits do not come for free. Learning useful representations from raw data often requires more data, additional training time, and more computation. It is essential to consider the tradeoffs between simple models and neural networks when attempting new projects. ML for molecules is also very different than for images or text. Often, molecules must be considered holistically; that is, every atom plays a role in the behavior of a molecule in the context of some endpoint. Depending on the task, molecules may contain little discardable noise (atoms or regions of molecules not critical to learning). This is in contrast with natural images or language, where low-level patterns are often less sensitive to subtle variations in the input. For example, in images, not every pixel is individually essential, and an object may still be easily recognizable after changing a few pixels. Similarly, in text, many words can be replaced with other, similar words without significantly changing the meaning of a sentence.

It is not always clear which form of representation is most appropriate for a given application, especially when applied to chemistry. Furthermore, it is important to realize that deep learning is not always the best choice, and non-learned featurizations may be simpler to work with or perform better in some applications, especially when data are limited.[6] Choosing which family of algorithms to pursue is often left to intuition and experimentation. When building models, there are many considerations to keep in mind, with only guiding principles to help inform our decisions. Most choices regarding data selection and model design are necessarily based on extensive trial-and-error and need to be rigorously validated using well-designed test sets. Choosing between non-learned and learned featurization, or selecting which types of deep learning models to try, is a balance between flexibility, practicality, engineering complexity, compute cost, and experimentation time required to both build and validate models. For more information regarding the theory and mathematics behind representation learning, and a comprehensive overview of many approaches seen throughout the machine learning community, we highly recommend reviewing Bengio *et al.*[34]

15.4 Learning Molecular Representations

Even with its weaknesses, learned featurization is an exciting and powerful paradigm. Defining the representation automatically from relatively raw data helps eliminate the need to make difficult assumptions about which molecular features should be included to predict a given endpoint. Learning representations can also better disambiguate explanatory factors in data and produce representations that are more highly correlated with many endpoints, leading to improved models.

Up until now, we have spoken only rather vaguely about learned representations and how they have been useful in other fields. However, we have not yet talked about, in practical terms and examples, the methods used to produce learned representations of molecules and the architectures best suited for learning such representations. In this section, we review representation learning algorithms explored in the literature, with particular

focus on the deep learning architectures used. Whether learning representations from inputs in an unsupervised way or in the context of an endpoint in supervised learning, the heart of representation learning for chemistry is the choice of architecture used to define the learning mechanism and its parameters. When developing and training an ML model, we make certain assumptions about the nature and structure of the data and the model to be learned. These assumptions comprise the "inductive bias" of our model and play a significant role in the effectiveness of learning. The inductive bias affects how fast we can hope to converge and any guarantees of convergence, the parameter space we assume the solution lies within, the generalizability of the final model, and the amount of data required to fit the model. The stronger the inductive bias, especially when assumptions are correct, the more all aspects of learning improve, and the resulting models are usually also of higher quality. The selection of which architecture family to use in an algorithm comprises a significant portion of the inductive bias of the model.

In the following subsections, we outline several families of architectures often seen in the chemistry ML literature. We first consider graph models (often called graph convolutions) that take in 2D representations of molecules and pass "messages" between atoms iteratively until latent representations of atoms and molecules are obtained. Next, we discuss grid models where information is propagated in 3D space without respect to any bonding structure, *i.e.*, atoms are treated as independent particles. We then describe sequence architectures that operate over molecules represented as strings of characters called SMILES. Lastly, we explore architecture designs inspired directly by physics.

Although there are many additional developments and results in the literature related to the discussed architectures, we focus on the representation learning portion of the work and how they enable researchers to learn over molecules with minimal preprocessing and encourage the reader to further investigate the literature for additional details. With an appreciation for both the value of representation learning and its tradeoffs and caveats, we can now consider the different families of architectures used to learn molecular representations.

15.4.1 Graph Models

There is an abundance of graph-structured data in chemistry, from directed graphs of chemical reactions, interaction networks of proteins, to the topological structure of small molecules themselves. Assuming a graphical prior in neural architectures helps improve learning over graphs, encouraging robustness and easing optimization. However, there are several approaches to introducing graphical priors in models, the choice of which is highly dependent on the chemistry to be modeled.

Before discussing graph neural networks (GNNs), let us revisit molecular fingerprints in the context of chemical graphs. The extended-connectivity circular fingerprint (ECFP) algorithm provides a solid basis for understanding graph convolutions wherein both algorithms apply the same operation everywhere within a molecule before integrating information through a global pooling step. Appreciating similar operations in non-learned fingerprints will pave the way for understanding the derivation of their differentiable generalization, Neural Graph Fingerprints,[35] an early graph convolutional (GC) method applied to molecules.

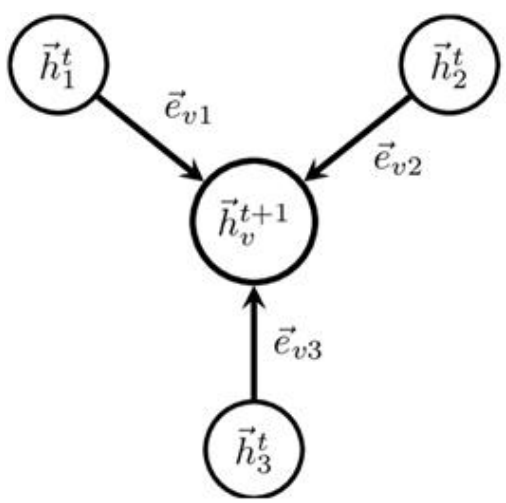
The ECFP algorithm is commonly denoted ECFP(N), where N is some positive, non-zero integer denoting the circular radius in number of bonds. As we iterate up to the full bond radius, for each atom in the molecular graph we gather its neighbors up to the current radius then pass the resulting molecular fragment to a hashing function where an index is assigned to the fragment. The hashing function ensures that a change in a fragment results in a change in its index. An indexing function then aggregates all indices into a single, sparse vector of fixed length. In this soup of fragments, there can be collisions, and fingerprints must be large to avoid overlap. Large, sparse vectors are also computationally expensive for downstream processing. These shortcomings motivate the development of more efficient representations that are smaller and denser, only encoding the relevant features for the task at hand. However, it is not trivial to find good hashing and indexing functions that can produce effective, yet dense, representations of chemical space. Furthermore, different prediction tasks may benefit from using different functions, but it is not always easy to anticipate which functions to use and when. Fortunately, the machinery of deep learning provides the ability to learn useful functions from data,

diminishing the need to define or select functions manually. Whereas circular fingerprints combine a hashing function and an indexing operation, neural fingerprints replace these with differentiable equivalents. The use of differentiation allows for gradient-based optimization of a loss function, making neural fingerprints learnable. The gradient of the loss with respect to the parameters can be used to steer the neural fingerprint parameters in such a way that predictive performance improves when compared with circular fingerprints. The superior neural-fingerprint prediction performance is primarily due to its ability to learn important features relevant to the predictive task at hand.[6,36]

Neural fingerprints encode fragments by processing input features through a series of neural network layers. The learned parameters (network weights) and their resulting activations over features provide a learned function that is counterpart to the hashing function in non-learned fingerprints. The learned function is smooth, enabling activations to be similar when molecular structure is similar. The index function is replaced with a softmax over the output vector of the learned hash function. In this way, each atom classifies itself as belonging to some learned category before being summed up across the molecule to produce the learned fingerprint. When this learned fingerprint was trained end-to-end with a deep feed-forward network it outperformed standard fingerprints using the same deep feed-forward architecture on several tasks, including predicting solubility and photovoltaic efficiency.[35]

In addition to neural graph fingerprints, many other GC approaches exist. There have been a few attempts to generalize GC architectures into a common mathematical framework. One such approach often referenced in the literature is the message passing neural networks (MPNNs).[5] The message passing framework generalizes the neural information passing algorithm on chemical graphs by splitting the method into two main phases: a message-passing phase and a readout phase. In the message-passing phase, for each atom in the molecule, the vector representations of neighboring atoms are passed through a learned, differentiable message function then summed, yielding the message vector for that atom. Depending on the choice of message function, messages can incorporate information from the representations of the current atom, neighboring

atoms, and the bonds between them. Typically, the same message function is used for all nodes in the neural network. This application of the same function everywhere is what makes it a convolutional network, and grants additional invariance by sharing parameters across the whole input. The message vector for an atom is then passed through a learned differentiable update function, along with the atom's representation, to produce the new representation for that atom. The output of the message passing step is a new representation vector for each atom in the chemical graph and is performed for N iterations, allowing information to propagate over a distance of N bonds. After all iterations of message passing are completed, the learned representations of all atoms in the chemical graph are integrated together using a learned readout function. The readout function produces a molecule-level representation that can be used in further processing. See Figure 15.1 for an illustration of MPNNs.



$$\vec{m}_v^{t+1} = \sum_{w \in N(v)} M_t(\vec{h}_v^t, \vec{h}_w^t, \vec{e}_{vw})$$

$$\vec{h}_v^{t+1} = U_t(\vec{h}_v^t, \vec{m}_v^{t+1})$$

$$\hat{y} = R(\{\vec{h}_v^T | v \in G\})$$

Figure 15.1 Aggregation of messages for a single node *v* in a single step *t* through message function *M* and update function *U*. After *T* steps, all the node vectors are passed to the readout function *R*.

For most architectures that fit in the MPNN framework, bond information is not considered. Bond-level representations can be learned but require an additional network. Other implementations such as the Directed MPNN (D-MPNN)[6] operate over directed bond-level representations, avoiding loops in message passing that would introduce

noise. D-MPNNs also combine computed molecular features with learned representations to overcome performance limitations of learned representations on small datasets to achieve state-of-the-art on many proprietary and public datasets.

However, GNNs are not without their limitations. Care has to be taken when designing certain architectural components of the neural networks. For example, not all GNNs are invariant to permutation of their indices, and in MPNNs, the readout function must be designed to handle such invariance. Networks lacking invariance to graph isomorphism can produce different atom or molecule level outputs for the same compound with different atom indexing. In these cases, canonicalization of the graph, usually through canonicalization of SMILES, should be done before generating graphs to pass to the network. Special consideration must also be taken when working with non-local properties on graphs. GNNs usually have a fixed propagation distance and cannot represent long-range quantum nonlocality.[37] Attempts to improve long-range message propagation include using a master atom connected to all other atoms in the molecule.[5] The master atom's latent vector acts as a scratchpad the neural network can read and write to during message passing. The authors of MPNN also tried adding a separate virtual edge type to help ease information flow between non-bonded atoms. Another design consideration in GC approaches relevant to capturing long-range interactions is the choice of graph traversal algorithm. Matlock *et al.* suggest a bond-wise, breadth-first traversal of the whole chemical graph for propagating information from a chosen center atom to the farthest atoms, then back again.[37] The forward-then-backward passing of information across the molecule can be repeated multiple times. They demonstrate the efficacy of their method on aromaticity and conjugation in large systems. Stereochemistry is another strong limitation of GNNs. While stereochemical information can be included in atom and bond level features before passing to the graph neural network, it is especially difficult to represent certain types such as conformational or relative stereochemistry. Such stereochemical limitations motivate the use of grid methods for learning molecular representations.

15.4.2 Grid Models

In graph models, propagation of information often only occurs between neighboring, bonded atoms. Additionally, most graph models ignore positional information and the spatial relationships between atoms. The exclusion of spatial information in data leads to learning representations that lack sufficient context for some applications, for example, predicting quantum mechanical energies or protein–ligand binding.[36] Attempts to capture non-bonded interactions in a graph-only approach becomes especially difficult in larger systems. Instead of relying solely on the graph, atoms can be positioned in a 3D coordinate space, represented as a grid, granting models access to those spatial relationships. Positioning Cartesian coordinates of inputs into 3D grids essentially discretizes the data and makes the data more suitable for processing using 3D convolutions. The familiar 2D convolutional neural networks (CNNs) often used in image recognition are most effective when used with planar (2D) inputs, *e.g.*, images, and are not well suited for 3D grids, and 3D convolutions are used instead. Similar to CNNs for images, 3D convolutional filters can extract features describing local covariance.

Models with access to spatial information can learn to assess the impact of interacting neighborhoods of atoms, enabling efficient learning of long-range interactions between atoms that are proximal in 3D space but not necessarily close in the graph. In many cases, molecular bonds are not explicitly included in the input data. However, bonding can be ascertained implicitly; the distances and angles between atoms, relative to the strength of their repulsion or attraction, are dependent on whether atoms are bonded. However, relying on the model to infer bonding may lead to ambiguity in some cases, especially when grid resolutions are too low or when data quality is poor and distances are noisy.

An area of strong interest for 3D grid models is predicting small-molecule binding to target proteins in structure-based drug discovery.[38,39] An early example of this approach is the AtomNet model,[38] which utilizes 3D-convolutions over a 3D grid containing the protein–ligand complex. The 3D grid is centered at the target protein's binding site, and small-molecule conformers are sampled within the site. For an approach like this to work, the location of each atom, both in the binding site and the molecule, is required. The additional 3D-structural information enables the model to

learn arbitrary, data-driven 3D features that can lead to better generalization. One goal of learning such a model is to capture favorable and unfavorable interactions between protein and ligand atoms then apply the model to new proteins and ligands not previously seen during training. Though it is difficult to find in the literature compelling examples of 3D grid models applied prospectively to structure-based drug-discovery projects, the approach remains promising. In principle, having access to both the protein and ligand may help 3D models become better at generalizing relative to ligand-only techniques.

Deep learning over 3D space has seen increased interest both in drug discovery and in other 3D applications, *e.g.*, processing point-cloud data or generative design in manufacturing. One challenge in 3D applications is that when operating over 3D grids, representations are often not invariant to rotation and translation. Instead of discretizing 3D data and placing in a grid, the precise 3D coordinate information can be preserved, enabling greater rotation and translation invariance. A particularly notable approach exploiting rotational symmetries in molecules is Tensor Field Networks.[40] In their method, geometric tensors are generated using spherical harmonics and Wigner D-matrices. To achieve SO(3) equivariance, each input (molecular structure) is converted to a set of 3D points with an associated SO(3) group (vectors). Essentially, SO(3) groups are used to describe the possible rotational symmetries of an object, as well as the orientation of an object in space, leading to the preservation of molecular features in 3D space. We further explore strategies for preserving invariances in the Physically inspired Architectures section, introduced later.

15.4.3 Sequence Models

An alternative to computing over molecular graphs with convolutions is to use recurrent neural networks (RNNs) with SMILES strings as input.[41] SMILES contain the same graphical information typically expressed in molecular 2D graphs, with the graph encoded in a 1D list of characters. The exact character ordering for a given molecule is determined by a ruleset specific to the implementation used to generate the SMILES. Atoms are represented with a one or two-letter elemental symbol, *e.g.*, “C” for carbon, with additional characters used to signify branching, chirality, and other

relationships between atoms in the graph. One reason to use SMILES rather than 2D graphs is that 1D inputs can readily be used in many RNN architectures essentially out-of-the-box. Sequence algorithms used in other domains that leverage attention,[26] reasoning,[42] or memory[43] can be adapted to use SMILES as input. Most often, RNNs are used to read in SMILES, though 1D convolutions have also been tried.[44] When generating molecules, it is also more straightforward to use RNNs to output character sequences than to generate 2D graphs.

GNNs usually pass messages along bonds to only immediate neighbors, and it can take several messaging passing steps to propagate information from one end of the molecule to another. In the case of sequence models, there is generally a single hidden state that all atoms have the opportunity to update successively, given the previous state. Having access to the full state when computing an update means that atoms have global access to the whole of the molecule (as encoded by the state), and long-range interactions are capturable. In the Yang *et al.* approach to GCs, message passing is centered around bonds instead of atoms, and messages are forced to travel along a directed path across the molecule to avoid totters (messages getting passed back and forth between two atoms, amplifying noise).[6] Interestingly, this is in parallel with how RNNs work with SMILES. In RNNs, the state is passed and updated from atom to atom in a directed, ordered fashion. Also, incorporating messages from both forward and backward directions is reminiscent of bi-directional RNNs where both forward and backward passes are made along the sequence.

After encoding a sequence, the RNN state is an intermediate, latent representation of a molecule and can be used to predict an endpoint.[45] Making predictions using RNNs is often done in a supervised learning setting, where both SMILES and labels are provided. However, there is also recent interest in using RNNs to learn from SMILES in a self-supervised way. Self-supervision (often called unsupervised learning) is where models are tasked with reproducing their input, often using autoencoders. Briefly, autoencoders work by taking input, learning to compress the input into a latent representation, then decoding the latent representation into a reconstruction of the input. Through the compression process, the model learns the salient features of the data and the structured patterns that best

describe the input. One or more constraints are often applied to the learning process to discourage autoencoders from merely learning the identity function. One constraint used to encourage more disentangled latent variables is the use of a Gaussian prior (μ=0, σ=1) applied to the latent vector, enforced through minimizing KL-divergence. This type of constrained autoencoder is called the variational autoencoder (VAE) and is based on Bayesian inference.[46]

VAEs and other forms of RNNs trained using self-supervision have been successfully applied in chemistry with compelling early examples in *de novo* applications.[31,47] Latent representations produced by autoencoders can be optimized for particular properties, with new molecules then generated from the optimized representations (often generated into SMILES strings).[31] A recent example in the literature tested generated molecules *in vitro*.[48] The motivation behind utilizing self-supervision with SMILES is that statistics about the distribution of latent variables describing a region of chemical space can be learned using large datasets without needing specific endpoint labels, *e.g.*, binding affinities. Although not utilizing SMILES as input, another interesting example of the application of VAEs to chemistry is learning reaction coordinates and trajectories.[49] Learning latent representations of trajectories enabled researchers to utilize physics in conjunction with ML, providing the ability to switch between the two during sampling.

When considering RNNs for an application, it is important to recognize that using RNNs with SMILES comes at a representational cost compared to GC-like approaches. RNNs are sensitive to input order,[50] and in SMILES, some character order must be assumed, *i.e.*, some atom has to come first. Additionally, treating the graph as a sequence requires the model to learn not only the relationship between the molecules and the endpoint (the desired task) but also how to infer graph structures from sequences. SMILES also utilize characters for annotating relationships between atoms, *e.g.*, parentheses for branching, which the model needs to learn in order to understand the graph instead of the model encoding this directly in its architecture (stronger inductive bias toward graphs) as in GC approaches.

15.4.4 Physically Inspired Architectures

Machine learning can be used in conjunction with physics-based calculations, where ML models first identify the most interesting molecules, with a portion then selected for further evaluation in more rigorous simulations. Similarly, empirical models can utilize physical calculations as input features. Though using physical features negates the benefit of avoiding the calculation expense, combining physics with ML can lead to methods that leverage benefits from each paradigm. Another strategy toward improved learning systems is the application of physical constraints on models, ensuring that learned representations capture more of the invariances naturally found in chemistry.

Much of the research within the chemistry ML community has focused on adopting successes seen in other domains, namely image recognition and language modeling, to chemistry. Many similarities can be observed between graph convolutions and the convolutional networks used in image classifiers.[51] Similarly, RNNs with SMILES are architected after recent successes in language modeling and other sequence modeling tasks. The rapid progression in the field of deep learning for chemistry with the varieties of approaches and applications seen in the growing body of literature is mainly through analogies drawn and lessons learned from other areas adapted to chemistry. Nevertheless, many challenges still remain.

Chemical space is vast, and, often, not much can be discarded as "noise" or "out of distribution," depending on the task and the degree of generalization desired. With 10^{60} molecules in drug-like space alone, datasets of realistic size are necessarily limited in how many relevant patterns they can contain. The difficulty in developing useful and large enough datasets causes many models to exploit biases and learn spurious correlations rather than learning more generally useful and invariant features. Furthermore, metrics often miss these weaknesses, making it difficult to ascertain model and method reliability.[52,53] In language modeling, individual words carry significant meaning and rich context, but in molecular data formats, atoms themselves are usually represented as discreet. Consequently, one subfragment in a molecule may have a completely different meaning in virtually all other chemical contexts, for example, the binding of a molecule in one protein *versus* its interaction with another. The difficulty in finding generally applicable patterns in even small

subfragments tends to result in more fragile ML models with significantly narrower reliability and makes transfer learning for chemistry especially difficult.[54] This is often referred to as a model's “domain of applicability” and requires researchers to be very aware of the robustness of models relative to chemical space.

A natural research direction toward learning more invariant molecular representations is to explore inductive biases closer to the underlying physics. Finding and applying stronger assumptions tied to physics instead of relying on overly flexible models with priors that are too general can significantly enhance learning and generalizability, and can help in building more robust models when trained with fewer data.

There are two general approaches to leveraging physical priors. The first is to retain significant flexibility in the learning algorithm (weak inductive bias) and learn to approximate generally applicable physical equations, such as those describing quantum mechanics, using large amounts of data. The goal of developing a model that approximates the equations directly is that with sufficient data the learned functions have some degree of transferability. Even with large volumes of data, however, we assume that the data distribution during testing is the same, or at least very similar to the data distribution seen during training. This approach generally leads to sufficiently good interpolation but is unreliable at best when attempting to extrapolate. Mills *et al.* utilize this data-first approach in approximating the Schrödinger equation.[55] When data are more limited, one way to compensate for the lack of invariance built into the model is through data augmentation. For example, randomly rotating molecules in 3D space can help artificially increase data volume.[39]

The second approach to leveraging physics is to find stronger priors that are closer to our understanding of physics and incorporate these priors and invariances into models directly. Kondor *et al.*, in their body of work around covariance in neural networks for chemistry, state their goal of using ML for molecules as not necessarily to learn the physical laws *per se*, but to understand and utilize the structure of physical laws to inform the types of invariances in molecules and physical interactions between atoms that can be captured in architectures.[56–58] They place particular emphasis on positional invariances (permutation, rotation, and translation) of atoms

when computing atom or groups-of-atoms representations and their interactions. They show how expanded invariances baked into the model help outperform similar state-of-the-art ML models using QM9 properties and MD-17 conformational energies. The method in Smith *et al.* is another example of employing stronger inductive bias where rotation invariance implemented at the model level relieves the network from needing to learn how to encode the same molecules in different orientations.[59] In Gomes *et al.*, the authors used the thermodynamics of protein–ligand binding to guide the interactions between components in their architecture.[60] Additionally, Levin *et al.* explore the connection between deep learning and quantum entanglement.[61]

The work of Schütt *et al.* emphasizes the importance of atom positions and provides a direction for developing ML models that can compute energies and forces for molecules in non-equilibrium configurations (positions and trajectories outside a local minimum in the potential energy landscape).[62] They describe designing their models around assumptions that are physically inspired, such as energy-conserving forces, to encourage the capture of more physically meaningful representations. Molecular physics has no concept of rotation or translation of atoms relative to a fixed coordinate system. Rather than using 3D Cartesian coordinates as part of the atom features, their model utilizes distances between atoms. To ease training, radial basis functions are used to provide additional non-linearity in the distances. Learning models that are robust to rotation relative to a coordinate system would necessitate augmenting the data with different molecular orientations, and with no guarantee that the same molecule in different orientations would produce the same result. Other researchers have also employed similar radial processing.[59,60] Schütt *et al.* additionally utilized a shifted softplus activation function to obtain smooth potential energy surfaces that are continuous in all orders of derivatives. The work by Gilmer *et al.* and Smith *et al.*, and likely others, show near-chemical accuracy compared with DFT but at a fraction of the computational cost.[1,5,59]

As a brief aside from chemistry, Lutter *et al.* used Lagrangian mechanics to learn physics describing robot motion.[63] Instead of approximating the inverse dynamics directly from data, as most deep learning systems would

do, they implemented the underlying equation explicitly as part of the model then approximated the physical parameters and subequations. Schmidt and Lipson also discuss learning natural physical laws from data.[64] The physics of chemistry is deeply rooted in mathematics, and it is not implausible to consider that stronger physical priors in our models may be possible. Nevertheless, physics is very complex. Energies and potentials of molecules are not isolatable atomistically but arise from complex interactions between many subatomic particles. Pfau *et al.* went beyond atomistic representations and used individual electrons as inputs to their deep neural network.[65] In limited hydrogen and nitrogen systems, they produced results that remarkably go beyond the accuracy of coupled-cluster. Although limited in scope thus far, modeling individual particles using deep learning is the beginning of a new and exciting approach compared with the atomistic models mostly explored thus far.

Regardless of whether physics is learned implicitly from data, or implemented directly in models, predicting physical endpoints are valuable benchmarks. Physical benchmarks help the community understand how well our models learn representations directly from data that sufficiently encode physical relationships, energies, and forces. For example, results were compared to DFT-computed ground state energies and electron density, and also compared to coupled-cluster predictions.[66] Similarly, Schütt *et al.* trained a model to learn the wave function describing quantum mechanics, which, if the representation adequately captures the physics, enables the prediction of virtually any ground-state property without the need for any additional training.[67] Like Schütt's work, many of the current lines of research assume molecules to be at ground state, and more work will be needed to eventually extend these methods to non-equilibrium conditions.

In a recent talk,[68] Dr. Léon Bottou discussed how common learning algorithms employ statistics only as a proxy, and causal relationships are generally not modeled effectively at all. He demonstrated that the removal of spurious correlations in favor of invariant relationships enabled much more effective extrapolation. His talk is not chemistry-specific but prompts valuable questions regarding the application of ML to chemistry. All molecule datasets contain some form of bias, primarily due to the sheer

vastness of chemical space, and assumptions need to be made when building a dataset, *e.g.*, the maximum number of atoms allowed. Some chemical datasets are extremely biased, such as in enumerations developed for protein–ligand binding. In enumerations, molecules often contain very similar cores, with only minor structural perturbations between molecules. Most features learned for such a dataset using typical ML methods have little hope of being useful in other learning tasks. The model will likely pick up on the strong bias contained in the dataset, and anywhere that bias is no longer present, the model will fail. Wallach and Heifets showed that many learning approaches employed in the drug discovery space are learning the specific biases contained within the datasets,[52] *e.g.*, the spurious correlations suggested by Bottou. Perhaps, if models can instead learn features more invariant across datasets, for example, something closer to the physics of chemistry, there is a higher chance of learning representations that are useful in a wide variety of learning tasks. More invariant features enable improved transfer learning (building a model from a large dataset and then fine-tuning the model using a much smaller dataset for another application), or even extrapolation if no additional training data are available. Learning chemical invariances also unlocks the opportunity for improved modeling of larger systems, such as protein-structure-based machine learning, and may increase the viability of ML when applied to biologics endpoints or protein folding prediction.

Although predicting physical endpoints remains a valuable research direction, much work is still needed to validate further the usefulness of deep learning in its current form for adequately approximating physics. The examples mentioned in this section showcase the value of designing architectures that are informed by invariances found in natural chemistry and the improvements toward generalization they can provide. Even stronger inductive biases closer to the underlying physics of chemistry may yet lead to even better performance across a variety of applications and datasets.

15.5 Data Considerations

Data quality and volume are prime concerns for building good models in nearly any application or domain. Apart from influencing the goodness of a model, data directly affects the quality of the learned representation; therefore, it is valuable to at least briefly consider the data's influence on representation quality. Because learned representations are data-driven, their quality is directly dependent on data quality. Care must be taken when curating data to ensure data are correct, and that validation sets remain useful for rigorous testing of models. Wu *et al.* discuss how the splitting of molecular datasets into training and validation sets must be done with care.[69] For example, if molecules with very similar scaffolds appear in both training and validation sets, validation may not catch overfitting on those particular scaffolds. Also, neural networks are incredibly flexible, and the latent representations learned are highly dependent on the volume and variety of molecules presented to the network. Consequently, due to the vastness of chemical space, essentially all chemical libraries will contain some bias. Care should be employed when both selecting training data and when applying a trained model to new data.

Properties of molecules are also extremely dependent on context. For example, the binding of a particular molecule may be very different in one protein compared to another protein, and mixing data from multiple assays may be disastrous. Laboratory experiments can also be notoriously difficult to reproduce due to subtle variations in temperature, pressure, materials, and in nearly any other component or condition of an experiment. Published experiments may also lack sufficient assay details to reproduce.[70] Even the same assay performed in different labs, or the same lab at different times, may not be consistent, and expertly measured experimental data may still be noisy. Furthermore, publicly available data are often highly variable in quality. Many principles exist for proper data handling, and we encourage the reader to explore the literature for insights relevant to their particular datasets.[70–72]

Another question that often arises in drug discovery and other areas of chemical discovery and development relates to the diversity of molecules in datasets. Screening libraries are often developed to contain a diverse sampling of chemical matter to help maximize chances of finding a hit, or finding hits with different scaffolds in the case when multiple hits are

found. The challenge, however, is that diversity is loosely defined and can be evaluated in a number of different ways using various metrics. It is also challenging to know if diversity will help in learning a robust model, in the context of an endpoint. One advantage of learned representations is that they are data-driven, and can be compared in the context of the training distribution. Common distance metrics can be calculated for representations produced using the learned featurization methods described in this chapter to measure similarity between molecules. In most cases, the learned representations are continuous, and Tanimoto, Euclidean, and cosine similarity metrics can be used to calculate distances. Dimensionality reduction of learned representations can also be used to visualize similarity. Popova *et al.* show examples of dimensionality reduction and how molecules with similar properties group together.[73] Measuring similarity using learned representations can also inform researchers regarding which molecules may be farthest from the training data and, hence, useful for adding to the training dataset.

15.6 Current Challenges and Directions for Future Research

When applying models, it is important to keep in mind the expectations we place on models regarding interpolation *versus* extrapolation. Most deep learning is highly interpolative, and a key assumption made when validating or deploying models is that the distribution of chemical space fed to the model is contained within the same distribution as the training data. A strategy for building more robust models includes increasing the size and variety of the training data, but large datasets for many endpoints are not always easy to obtain. Many datasets in chemistry are relatively small, and the ability to leverage other, much larger datasets to improve performance in applications when data is sparing could be transformative. A key benefit of representation learning is its ability to help facilitate transfer learning; yet, transfer learning for chemistry remains challenging.[54] In assessing whether transfer learning is a reasonable expectation for chemical datasets, a key question is what patterns can we expect models to learn that are

transferable to other systems? In protein–ligand binding, for example, binding mechanisms vary across proteins. Most interactions between protein and ligand that are relevant in one protein system may not be useful in another.

Pretraining over a large corpus of data, then fine-tuning a trained network over a few examples for a new application, has seen success in image recognition. In transfer learning, we assume that patterns are common among tasks, and that learned features are invariant, *i.e.*, they can pick up the same patterns across different data. In the discussion above regarding physically inspired architectures, we mentioned that more closely approximating physics may help with transfer learning. In the extreme case, if we imagine that it is possible to approximate the physics perfectly, then we will have effectively learned a model that could readily perform zero-shot transfer learning. Of course, approximating physics to that extent is not currently possible. However, we can consider cases where approximating physics may help and also consider directions to explore toward improving the approximations.

Building models closer to physics is scientifically interesting and may help with learning more invariant features, but current deep learning approaches may not be sufficient for learning such features. For example, most of the architectures described above produce distributed vectors, which are likely not physically meaningful. Even when disentangling latent variables is especially encouraged, such as in the VAE, we do not have any explicit control over which variables are expressed, nor an adequate way to interpret causal relationships between variables. Learning causality is an exciting branch of ML research currently being pursued. Considering the deeply rooted relationship between causality and physics, similar questions and lines of research may be especially relevant in ML for chemistry. Regardless of the direction taken, a strong emphasis on physics, either learning the physics directly or at least used to help validate ML models, will be important when applying future ML advancements to chemistry.

15.7 Conclusion

Learning molecular representations directly from data both simplifies the development of machine learning models for chemistry and can greatly improve the usefulness of the trained models. There are many ways to represent input molecules, including graphs or points in 3D space, and several different families of architectures, each suited for learning from different forms of input, can be used to learn over minimally processed data. Regardless of the type of architecture used, the goal of deep learning or any other form of modeling for chemistry is to develop more useful tools that aid in the design and application of new chemical matter and the furthering of science. One advantage of deep learning over many other forms of machine learning is its ability to learn representations directly from data without the need to rely on expertly curated features that may lack general applicability or are otherwise difficult to obtain.

Most of the models published in the literature thus far are inspired by models developed in other domains, namely, image recognition and language modeling, and may not always be well suited for chemistry tasks. Transfer learning remains one such area where successes seen in other domains have not been realized to the same extent in chemistry. Physically inspired architectures is a promising line of research that can potentially lead to more robust and powerful models and may also aid in more reliable extrapolation. Currently, physically inspired architectures are mostly conservative in the extent that physics is included or have only been tested in small systems thus far; more work is, therefore, needed.

Deep learning is currently the dominant approach in learning molecular representations. Thus, all the challenges and weaknesses of deep learning are relevant in our chemistry models, including needing large training sets, architecture sensitivity to hyperparameters, and fragility when applied to new data outside the domain of applicability. It is not yet clear if deep learning is the best paradigm for learning molecular representations, as seen by the difficulty in transfer learning, and physics will continue to play a significant role in validating the usefulness of representations and further inspiring future learning algorithms. With the limitations of deep learning aside, the impact of learning representations from raw data is clear, with many new approaches and applications published utilizing these methods that achieve high accuracy at a relatively reduced computational cost.

Acknowledgements

The authors thank Robert Abel, Kyle Marshall, and Shawn Watts for helpful feedback regarding the manuscript, and Chandler Watson for identifying interesting and relevant literature.

References

1. J. Smith, B. Nebgen, R. Zubatyuk, N. Lubbers, C. Devereux, K. Barros, S. Tretiak, O. Isayev and A. Roitberg, *Nat. Commun.*, 2019, **10**, 2903.
2. A. S. Christensen, T. Kubař, Q. Cui and M. Elstner, *Chem. Rev.*, 2016, **116**, 5301–5337.
3. A. Cherkasov, E. N. Muratov, D. Fourches, A. Varnek, I. I. Baskin, M. Cronin, J. Dearden, P. Gramatica, Y. C. Martin, R. Todeschini, V. Consonni, V. E. Kuz'min, R. Cramer, R. Benigni, C. Yang, J. Rathman, L. Terfloth, J. Gasteiger, A. Richard and A. Tropsha, *J. Med. Chem.*, 2014, **57**, 4977–5010.
4. G. Sliwoski, S. Kothiwale, J. Meiler and E. W. Lowe, *Pharmacol. Rev.*, 2014, **66**, 334–395.
5. J. Gilmer, S. Schoenholz, P. Riley, O. Vinyals and G. Dahl, presented in part at ICML, Sydney, Aug., 2017.
6. K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. P. Kelley, M. Mathea, A. Palmer, V. Settels, T. S. Jaakkola, K. F. Jensen and R. Barzilay, 2019, DOI: 10.26434/chemrxiv.7940594.v1.
7. A. Cereto-Massagué, M. J. Ojeda, C. Valls, M. Mulero, S. Garcia-Vallvé and G. Pujadas, *Methods*, 2015, **71**, 58–63.
8. D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.
9. Dalke Scientific, http://www.dalkescientific.com/writings/diary/archive/2014/10/17/maccs_key_44.html, (accessed Nov 2019,).
10. D. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik and R. Adams, presented in part at NIPS, Montréal, 2015.

11. M. H. S. Segler, M. Preuss and M. P. Waller, *Nature*, 2018, **555**, 604–610.
12. D. T. Ahneman, J. G. Estrada, S. Lin, S. D. Dreher and A. G. Doyle, *Science*, 2018, **360**, 186–190.
13. D. A. Winkler, *Briefings Bioinf.*, 2002, **3**, 73–86.
14. H. Moriwaki, Y.-S. Tian, N. Kawashita and T. Takagi, *J. Cheminf.*, 2018, **10**, 4.
15. L. Z. Benet, C. M. Hosey, O. Ursu and T. I. Oprea, *Adv. Drug Delivery Rev.*, 2016, **101**, 89–98.
16. P. Jain and S. H. Yalkowsky, *Int. J. Pharm.*, 2010, **385**, 1–5.
17. M. Karelson, V. S. Lobanov and A. R. Katritzky, *Chem. Rev.*, 1996, **96**, 1027–1044.
18. E. Eroğlu, H. Türkmen, S. Güler, S. Palaz and O. Oltulu, *Int. J. Mol. Sci.*, 2007, **8**, 145–155.
19. P. Sarmah and R. C. Deka, *J. Comput.-Aided Mol. Des.*, 2009, **23**, 343–354.
20. G. Fayet, D. Jacquemin, V. Wathelet, E. A. Perpète, P. Rotureau and C. Adamo, *J. Mol. Graphics Modell.*, 2010, **28**, 465–471.
21. R. Todeschini and V. Consonni, *Molecular Descriptors for Chemoinformatics*, Wiley-VCH, Weinheim, 2009.
22. S. Winiwarter, M. Ridderström, A.-L. Ungell, T. B. Andersson and I. Zamora, *Comprehensive Medicinal Chemistry II*, Elsevier, 2007, pp. 531–554.
23. K. Roy, S. Kar and R. N. Das, *Understanding the Basics of QSAR for Applications in Pharmaceutical Sciences and Risk Assessment*, Elsevier, 2015, pp. 47–80.
24. A. Krizhevsky, I. Sutskever and G. Hinton, presented in part at NIPS, Lake Tahoe, Dec., 2012.
25. K. He, X. Zhang, S. Ren and J. Sun, presented in part at IEEE ICCV, Santiago, Chile, 2015.
26. T. Luong, H. Pham and C. D. Manning, presented in part at Conference on Empirical Methods in Natural Language Processing, Lisbon, 2015.
27. T. Karras, S. Laine and T. Aila, *arXiv:1812.04948*.
28. D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F.

Hui, L. Sifre, G. van den Driessche, T. Graepel and D. Hassabis, *Nature*, 2017, **550**, 354–359.
29. N. Aafaq, A. Mian, W. Liu, S. Z. Gilani and M. Shah, *ACM Comput. Surv.*, 2019, **52**, 1–37.
30. E. Zakharov, A. Shysheya, E. Burkov and V. Lempitsky, *arXiv:1905.08233*.
31. R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2018, **4**, 268–276.
32. X. Yan, J. Yang, K. Sohn and H. Lee, presented in part at ECCV, Amsterdam, Dec., 2016.
33. D. Berthelot, C. Raffel, A. Roy and I. Goodfellow, *arXiv:1807.07543*.
34. Y. Bengio, A. Courville and P. Vincent, *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013, **35**, 1798–1828.
35. D. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik and R. Adams, presented in part at NIPS, Montréal, 2015.
36. S. Kearnes, K. McCloskey, M. Berndl, V. Pande and P. Riley, *J. Comput.-Aided Mol. Des.*, 2016, **30**, 595–608.
37. M. K. Matlock, N. L. Dang and S. J. Swamidass, *ACS Cent. Sci.*, 2018, **4**, 52–62.
38. I. Wallach, M. Dzamba and A. Heifets, *arXiv:1510.02855*.
39. M. Ragoza, J. Hochuli, E. Idrobo, J. Sunseri and D. R. Koes, *J. Chem. Inf. Model.*, 2017, **57**, 942–957.
40. N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff and P. Riley, *arXiv:1802.08219*.
41. G. Goh, N. Hodas, C. Siegel and A. Vishnu, *arXiv:1712.02034*.
42. C. D. Manning and D. Hudson, presented in part at ICML, Vancouver, Canada, 2018.
43. A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, A. P. Badia, K. M. Hermann, Y. Zwols, G. Ostrovski, A. Cain, H. King, C. Summerfield, P. Blunsom, K. Kavukcuoglu and D. Hassabis, *Nature*, 2016, **538**, 471–476.

44. M. Hirohara, Y. Saito, Y. Koda, K. Sato and Y. Sakakibara, *BMC Bioinf.*, 2018, **19**, 526.
45. Z. Xu, S. Wang, F. Zhu and J. Huang, presented in part at the 8th ACM International Conference on *Bioinformatics, Computational Biology, and Health Informatics – ACM-BCB, Boston*, 2017.
46. C. Doersch, *arXiv:1606.05908*.
47. M. Olivecrona, T. Blaschke, O. Engkvist and H. Chen, *J. Cheminf.*, 2017, **9**, 48.
48. A. Zhavoronkov, Y. A. Ivanenkov, A. Aliper, M. S. Veselov, V. A. Aladinskiy, A. V. Aladinskaya, V. A. Terentiev, D. A. Polykovskiy, M. D. Kuznetsov, A. Asadulaev, Y. Volkov, A. Zholus, R. R. Shayakhmetov, A. Zhebrak, L. I. Minaeva, B. A. Zagribelnyy, L. H. Lee, R. Soll, D. Madge, L. Xing, T. Guo and A. Aspuru-Guzik, *Nat. Biotechnol.*, 2019, **37**, 1038–1040.
49. J. M. L. Ribeiro, P. B. Collado, Y. Wang and P. Tiwary, *arXiv:1802.03420*.
50. O. Vinyals, S. Bengio and M. Kudler, presented in part at ICLR, San Diego, 2015.
51. Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang and P. S. Yu, *arXiv:1901.00596*.
52. I. Wallach and A. Heifets, *J. Chem. Inf. Model.*, 2018, **58**, 916–932.
53. K. McCloskey, A. Taly, F. Monti, M. P. Brenner and L. Colwell, *arXiv:1811.11310*.
54. S. Ramsundar, P. Kearnes, S. Riley, D. Webster, D. Konerding and V. Pande, *arXiv:1502.02072*.
55. K. Mills, M. Spanner and I. Tamblyn, *Phys. Rev. A*, 2017, **96**, 042113.
56. B. Anderson, T.-S. Hy and R. Kondor, *arXiv:1906.04015*.
57. R. Kondor, H. T. Son, H. Pan, B. Anderson and S. Trivedi, *arXiv:1801.02144*.
58. R. Kondor, *arXiv:1803.01588*.
59. J. S. Smith, O. Isayev and A. E. Roitberg, *Chem. Sci.*, 2017, **8**, 3192–3203.
60. J. Gomes, B. Ramsundar, E. N. Feinberg and V. S. Pande, *arXiv:1703.10603*.
61. Y. Levine, D. Yakira, N. Cohen and A. Shashua, presented in part at ICLR, Vancouver, 2018.

62. K. Schütt, P.-J. Kindermans, H. E. S. Felix, S. Chmiela, A. Tkatchenko and K.-R. Müller, presented in part at NIPS, Long Beach, 2017.
63. M. Lutter, C. Ritter and J. Peters, presented in part at ICML, New Orleans, 2019.
64. M. Schmidt and H. Lipson, *Science*, 2009, **324**, 81–85.
65. D. Pfau, J. S. Spencer, A. G. D. G. Matthews and W. M. C. Foulkes, *arXiv:1909.02487*.
66. A. V. Sinitskiy and V. S. Pande, *arXiv:1908.00971*.
67. K. T. Schütt, M. Gastegger, A. Tkatchenko, K.-R. Müller and R. J. Maurer, *arXiv:1906.10033*.
68. L. Bottou, presented in part at ICLR, New Orleans, 2019.
69. Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing and V. Pande, *Chem. Sci.*, 2018, **9**, 513–530.
70. T. Kalliokoski, C. Kramer, A. Vulpetti and P. Gedeck, *PLoS One*, 2013, **8**, e61007.
71. M. Baker, *Nature*, 2016, **533**, 452–454.
72. G. Papadatos, A. Gaulton, A. Hersey and J. P. Overington, *J. Comput.-Aided Mol. Des.*, 2015, **29**, 885–896.
73. M. Popova, O. Isayev and A. Tropsha, *Sci. Adv.*, 2018, **4**, eaap7885.

CHAPTER 16

Demystifying Artificial Neural Networks as Generators of New Chemical Knowledge: Antimalarial Drug Discovery as a Case Study

ALEJANDRO SPECK-PLANCHE(0000-0002-9544-9016)*[a] AND VALERIA V. KLEANDROVA(0000-0002-1928-853X)[b]

[a] Programa Institucional de Fomento a la I+D+i, Universidad Tecnológica Metropolitana, Ignacio Valdivieso 2409, San Joaquín, Santiago, Chile
[b] Laboratory of Fundamental and Applied Research of Quality and Technology of Food Production, Moscow State University of Food Production, Volokolamskoe shosse 11, 125080, Moscow, Russian Federation
*Email: alejspivanovich@yahoo.es

16.1 Introduction

With the advent of machine learning (ML) methods and computational models, which are crucial in most modern scientific projects, drug discovery has undergone a process of constant change and evolution.[1] Nowadays, the important role of ML methods in areas such as data mining and virtual screening is well-established.[1–3] Among them, artificial neural networks (ANNs) have become a focus of attention, particularly due to the emergence of revolutionary subfields such as deep learning.[4] In fact, within early drug

discovery, ANNs have found impressive applications in the *de novo* design of molecules with desired properties.[5–8]

Nevertheless, the efficiency and success of current ANNs are strongly associated with a very high computational cost, and the advanced degree of programming/codification skills that researchers must have; the latter can take too much time to master. In addition to these limitations, two other handicaps prevent the full exploitation of ANNs in drug discovery. From one side, the computational models based on ANNs continue to be treated as black boxes because they lack interpretability; they are not capable of providing a clear interpretation of the physicochemical properties and/or structural features of the chemicals in the databases, which they intend to design and/or predict. On the other hand, the *in silico* models focused on ANNs consider a specific stage (usually the pharmacological activity domain) of the drug discovery process. Considering the multi-scale nature of drug discovery and the vast chemical space that needs to be filtered (10^{60} small to medium-sized organic compounds),[9] one of the greatest challenges is the development of models based on ANNs that can integrate multiple biological data and design seemingly new molecular entities with potent activity, low toxicity, and adequate pharmacokinetic properties.

In the last seven years, several research groups have focused their efforts on the creation of a series of computational tools known as multi-scale models for quantitative structure-biological effect relationships (ms-QSBER).[10] These models can integrate different sources of data, permitting the simultaneous prediction of multiple biological endpoints against many targets of varying degrees of complexity. In this sense, ms-QSBER models (some of them based on ANNs) have been successfully applied to different fields of research such as cancer research,[11] infectious diseases,[12–22] nanomedicine and nanotoxicology,[23–27] immunology and immunotoxicity,[28–31] and neuroscience.[32–35] Also, multi-scale *de novo* drug design[36] has emerged as an encouraging paradigm, combining the predictive capabilities of the ms-QSBER models, chemical reasoning, and concepts from fragment-based drug discovery to accelerate the rational design of molecules that virtually exhibit high activity, low toxicity, and very good pharmacokinetics.[37] Yet, the applications of ANNs in the context of ms-QSBER models have been restricted to performing virtual screening tasks.

Bearing in mind all the previous ideas, this chapter intends to demonstrate that computational models based on ANNs can be used in a chemistry-friendly manner to accelerate the design of virtually new, potent, and safe therapeutics at the preclinical level. Thus, we report the first ms-QSBER model based on an ensemble of ANNs (ms-QSBER-EL) devoted to performing virtual design and prediction of potentially active and safe antimicrobial agents. Here, antimalarial drug discovery is used as a case study for a number of reasons. First, malaria is one of the oldest diseases known by mankind, which has an elevated mortality and morbidity; 445 000 deaths and 216 million cases of malaria occurred worldwide in 2016.[38] Second, the efficacy of antimalarial treatments has diminished because *Plasmodium falciparum*, widely recognized as the main causal agent of malaria, has become resistant to antimalarial drugs.[39] Last, there are serious concerns regarding the adverse reactions associated with the use of antimalarial therapeutics.[40]

16.2 Materials and Methods

16.2.1 Dataset and Molecular Descriptors

The development of the ms-QSBER models has been widely reported,[13–17,37,41] and therefore, only the procedures specific of this work will be explained. All the chemical and biological data employed to build the ms-QSBER-EL model were extracted from the ChEBML database[42–44] in the form of a Microsoft Excel-compatible file. The dataset used in this study contained 18 798 different drugs/chemicals belonging to many diverse chemical families. Each molecule was experimentally assayed by considering 1 out of 7 measures of biological effects (activity, cytotoxicity, and pharmacokinetic properties) and against at least 1 out of 20 targets (including proteins, *Plasmodium falciparum* strains, mammalian cells, and liver microsomes). If a molecule was found to be assayed more than one time against the same target and by considering the same measure of biological effect, its values were averaged. In any case, notice that most of the drugs/chemicals reported in the present dataset were experimentally tested by considering only one measure of biological effect and against only one target.

Thus, the dataset used here ended up containing 21 369 statistical cases and so is characterized by a great dispersion of the data. Nevertheless, the great success of the ms-QSBER models mentioned in the previous section is based on their ability to handle greatly dispersed and heterogeneous data.[13–17,37,41,45]

Each statistical case in the dataset was annotated as positive [$BE_i(c_j)=1$] or negative [$BE_i(c_j)=-1$], $BE_i(c_j)$ being a binary categorical variable that described the biological effect of the *ith* case/chemical under the experimental condition c_j. It should be pointed out that a defined experimental condition c_j depended on both the *qth* measure of biological effect (*me*) reported in a given assay and the *rth* target (*tg*) against which the assay was performed. Therefore, each experimental condition can be expressed as $c_j(me, tg)$. The dataset employed in this work contained 23 unique experimental conditions (combinations of the elements *me* and *tg*).

Assignments of positive and negative cases have been realized by considering different cutoff values that are depicted in Table 16.1; several reasons justify the selection of these rigorous cutoffs. First, an inspection of the dataset extracted from ChEMBL showed that there are FDA-approved drugs reported with biological effects whose values fell either above or below the cutoffs. This aspect highlights the importance of the selected cutoff values because, if a query molecule is predicted as positive (or negative) by a model derived from the present dataset, it will be possible to compare its potential biological effect with those already reported for the clinically relevant drugs. Second, the assignments of the compounds as positive or negative implies that a classification approach was employed instead of a regression one. In contrast to their regression counterparts, classification approaches do not need to predict the exact values of the different properties in a dataset, and, therefore, they do not need to deal with the potentially great uncertainty of the data. Third, the cutoff values selected in this study prevented the imbalance between the number of molecules assigned as positive and the number of molecules annotated as negative. Last, from a phenomenological point of view, even though several clinically relevant antimalarial drugs are present in the dataset, the purpose of the ms-QSBER-EL model is to perform simultaneous predictions and design of new chemicals exhibiting high antimalarial activity, low cytotoxicity, and

adequate pharmacokinetic properties. The design of new molecules with chemical structures differing from those present in antimalarial drugs indicates the idea of searching for novel mechanisms of action of antimalarial activity as well as reducing the toxicity and improving the pharmacokinetic profiles not found in most of the current antimalarial drugs.

Table 16.1 Different cutoff values of biological effects used in this study.

Measure of biological effect (*me*)	Cutoff[a]	Concept	Targets (*tg*)
IC_{50} (nM)	≤370.00	Concentration required to inhibit at 50% of the growth of the parasite.	*Plasmodium falciparum* strains.
Permeability (μcm s^{-1})	≥10	Permeability across the cell membrane.	Caco-2, MDCK, and MDR1-MDCK cells.
PPB (%)	≤85	Percentage of a chemical bound to plasma proteins.	Plasma from mouse, rat, and human.
$t_{1/2}$ (h)_Lvm	≥0.6	Time required to reduce the concentration of a chemical to half of its initial value in liver microsomes.	Liver microsomes from rat and human.
$t_{1/2}$ (h)_Plm	≥1.5	Time required to reduce the concentration of a chemical to half of its initial value in plasma.	Plasma from mouse, rat, and human.
CC_{50} (nM)	≥25 000	Cytotoxic concentration causing death to 50% of viable cells.	Different mammalian cells from hamster, mouse, rat, and human.
IC_{50} (nM)_HERG	≥10 000	Concentration causing 50% inhibition of the ion channel HERG.	The ion channel known as HERG.

[a] Values from which a chemical was assigned as positive.

To characterize the chemical structures of the molecules present in the dataset, we calculated the molecular descriptors known as the total [$TnsAq_k(x)$] and local [$LnsAq_k(x)Z$] atom-based quadratic indices of the non-stochastic matrix.[46] Quadratic indices have been demonstrated to exhibit a greater discriminant power[47] than that shown by combining several families of molecular descriptors calculated by popular programs such as DRAGON.[48] The quadratic indices can be calculated according to the following mathematical formalism:

$$TnsAq_k(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} {}^{k}a_{ij} \cdot x_i \cdot x_j \tag{16.1}$$

$$LnsAq_k(x)Z = \sum_{i=1}^{n}\sum_{j=1}^{n} {}^{k}a_{ijZ} \cdot x_i \cdot x_j \tag{16.2}$$

In eqn (16.1) and (16.2), the element x accounts for any atomic physicochemical property such as hydrophobicity (HYD), electronegativity (E), atomic weight (AW), polarizability (POL), and polar surface area (PSA). At the same time, in eqn (16.1), the element ${}^{k}a_{ij}$ characterizes the adjacency between any two atoms of the molecule, while in eqn (16.2), ${}^{k}a_{ijZ}$ has a similar meaning, except for the fact that it focuses on specific types of atoms (Z) such as hydrogen bond acceptors or donors, carbon atoms (both aliphatic and aromatic), and others. Both classes of atom-based quadratic indices consider an atom i and its chemical environment (composed of the *jth* neighbors) at the topological distance $d=k$.

To calculate the aforementioned topological indices, the file containing the SMILES codes of all the molecules was manually changed from *.txt to *.smi. After, the software Standardizer v19.18.0 was used to transform the SMILES codes to sdf format.[49] The only purpose of converting to sdf format was to obtain the connectivity table for each molecule; no additional standardization actions were used. The sdf file was used as input to calculate quadratic indices by means of the software QuBiLS-MAS v1.0[47,50] For such a calculation, the software QuBiLS-MAS v1.0 employed a predefined

configuration [algebraic form: quadratic; constrains: atom-based; matrix forms: non-stochastic; maximum order: 6; cutoff: keep all; groups: total and local (hydrogen bond acceptors, aliphatic carbon atoms, aromatic carbons, terminal methyl groups, halogens, and heteroatoms); properties: already mentioned in the previous paragraph; aggregation: Manhattan distance].

In eqn (16.1) and (16.2), the quadratic indices cannot differentiate the chemical structure of a molecule when this is tested by considering multiple measures of the biological effects and against different targets. Fortunately, the ms-QSBER models resolve this issue by applying the Box-Jenkins operators[17,37,51] to generate new molecular descriptors able to consider both the chemical structure and the set of experimental conditions (*e.g.*, target, the measure of activity, or toxicity, *etc.*) against which a molecule has been assayed:

$$avgQI(c_j) = \frac{1}{n(c_j)} \times \sum_{a=1}^{n(c_j)} QI_a \tag{16.3}$$

In eqn (16.3), $n(c_j)$ is the number of molecules annotated as positive, which were assayed under the same experimental condition c_j, *i.e.*, by considering both the same measure of biological effect and the same target.[52] In addition, QI_a refers to any quadratic index (total or local), while $avgQI(c_j)$ is the arithmetic mean of the QI_a values for all the molecules assigned as positive and tested under the same assay condition c_j. In a second step related to the application of the Box-Jenkins operators, the following expression is used:

$$DQI_a(c_j) = \frac{QI_a - avgQI(c_j)}{(QI_{MX} - QI_{MN}) \times \sqrt{p(c_j)}} \tag{16.4}$$

In eqn (16.4), $DQI_a(c_j)$ measures how much a molecule structurally deviates from a set of chemicals assigned as positive, all of them being tested under the same assay condition c_j. On the other hand, QI_{MX} and QI_{MN} represent the maximum and minimum values for a defined molecular descriptor, respectively; they don't depend on the experimental conditions.

Last, $p(c_j)$ is the *a priori* probability of finding a chemical tested under a specific experimental condition c_j:

$$p(c_j) = \frac{n(c_j)}{N_T} \tag{16.5}$$

In eqn (16.5), the element $n(c_j)$ has been defined in eqn (16.3), and N_T refers to the total number of cases/chemicals present in the training set.

16.2.2 Building the Ms-QSBER_EL model

The steps leading to the generation of the present ms-QSBER-EL model are depicted in Figure 16.1. The dataset formed by the 21 369 cases/chemicals was randomly split into two series: training and test sets. The training set was used to search for the best model and was formed by 6591 cases assigned as positive and 9571 annotated as negative, making a total of 16 162 cases (75.63% of the dataset). The purpose of the test set was to validate the model by assessing its predictive power; this set was formed by 5207 cases (24.37%), 2064 annotated as positive and 3143 assigned as negative.

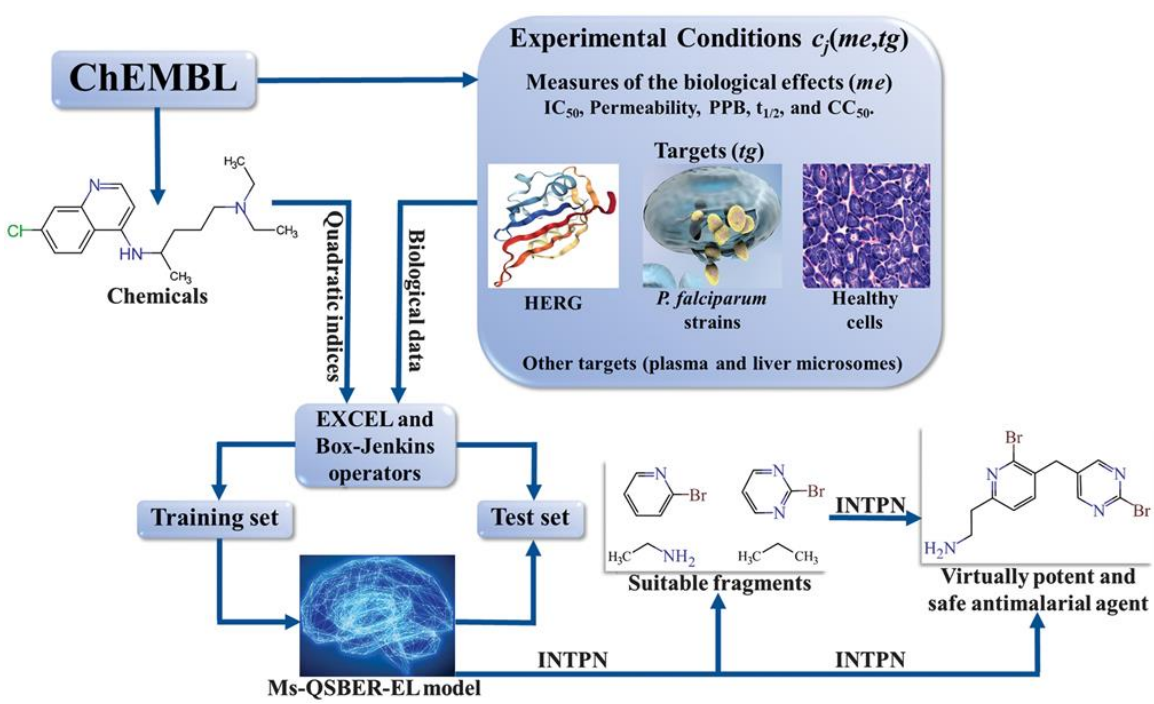


Figure 16.1 Workflow focused on the development and application of the ms-QSBER-EL model. The abbreviation "INTPN" stands for interpretation, which is given for each molecular descriptor from a physicochemical and structural point of view.

To find the best the ms-QSBER-EL model, the two most popular ANN architectures were tested: radial basis function (RBF) and multi-layer perceptron (MLP). The ANN package available in the software STATISTICA v13.5.0.17 was employed to generate the ms-QSBER-EL model.[53] When using the ANN package, the first run was performed to assess the appropriateness of three elements: 1) the most significant ANN architectures, 2) the most influential molecular descriptors, and 3) the correlation between the descriptors.

From one side, the most significant ANN architectures were determined by analyzing the training and test errors (as low as possible) and the values of different statistical indices (set to be as high as possible) such as sensitivity [*Sn*(%)], specificity [*Sp*(%)], accuracy [*Ac*(%)], and the Matthews' correlation coefficient (*MCC*).[54] These statistical indices were used to estimate the internal quality (training set) and the predictive power (test set) of the ms-QSBER-EL model.

On the other hand, the most influential molecular descriptors in the ms-QSBER-EL model were assessed by means of a sensitivity analysis, which was performed by the software STATISTICA v13.5.0.17. This analysis was applied as a variable selection strategy, quantifying the effect of each molecular descriptor by calculating its sensitivity/importance value (*SV*). In this context, *SV* was calculated as the error generated in a defined ANN when estimating the value of a molecular descriptor using a missing value procedure, divided by the error generated in that ANN by using the real value of the same molecular descriptor. Only those molecular descriptors with $SV>1$ were considered to enter in the ms-QSBER-EL model. Also, when selecting the most significant descriptors, the correlation between them was assessed by calculating the Pearson's correlation coefficient (*PCC*),[55] and the cutoff interval used to indicate the lack of redundancy/correlation was $-0.7<PCC<0.7$.

As defined in the introductory section, the abbreviation "ms-QSBER-EL" stands for multi-scale model for quantitative structure-biological effect relationships combined with ensemble learning. Therefore, as the final step in the development of the ms-QSBER-EL model, instead of selecting a single

ANN, the best ensemble of ANNs was chosen. This comes from the fact that when solving a particular problem, and by providing a consensus response, models based on ensembles of different ANNs can often perform better than the single ANNs from which they were created.[56]

16.3 Results and Discussion

16.3.1 Ms-QSBER-EL model

The best ms-QSBER-EL model developed in this work was an ensemble comprised of three ANNs with the profiles MLP 8-61-2, MLP 8-67-2, and MLP 8-80-2. The first ANN was constructed by using the logistic function in the hidden layer and softmax function for the output layer. The second and third ANNs had the same activation function in the hidden layer (hyperbolic tangent). For the output layer, the second ANN employed a logistic function while for the third ANN, the softmax function was employed. Notice that all the ANNs contain 8 nodes (molecular descriptors) in the input layer. The number two is the same for all of them in the output layer because they are all predicting the two possible labels (the values −1 and 1) of the categorical variable of biological effect [$BE_i(c_j)$]. In any case, the three ANNs have a different number of neurons in their hidden layers. The symbols and definitions of each molecular descriptor present in the ms-QSBER-EL model are reported in Table 16.2, while all the chemical and biological data are available in the **Supplementary Information 16.1**.[†] The files corresponding to the three ANNs can be obtained upon request to the author.

Table 16.2 Molecular descriptors present in the ms-QSBER-EL model.

Molecular Descriptor[a]	Definition
$\boldsymbol{D[TnsAq6(AW)]c_j}$	Deviation of the total non-stochastic (atom-based) quadratic index of order 6 weighted by the atomic weight.
$\boldsymbol{D[LnsAq1(POL)A]c_j}$	Deviation of the local non-stochastic (atom-based) quadratic index of order 1 weighted by the

	polarizability. This descriptor focuses on the atoms acting as hydrogen bond acceptors and their chemical environments.
***D*[*LnsAq*4(*AW*)*C*]c_j**	Deviation of the local non-stochastic (atom-based) quadratic index of order 4 weighted by the atomic weight. This descriptor focuses on the aliphatic carbons and their chemical environments.
***D*[*LnsAq*2(*E*)*G*]c_j**	Deviation of the local non-stochastic (atom-based) quadratic index of order 2 weighted by the electronegativity. This descriptor focuses on the halogens and their chemical environments.
***D*[*LnsAq*1(*POL*)*G*]c_j**	Deviation of the local non-stochastic (atom-based) quadratic index of order 1 weighted by the polarizability. This descriptor focuses on the halogens and their chemical environments.
***D*[*LnsAq*6(*AW*)*M*]c_j**	Deviation of the local non-stochastic (atom-based) quadratic index of order 6 weighted by the atomic weight. This descriptor focuses on the methyl groups and their chemical environments.
***D*[*LnsAq*2(*AW*)*P*]c_j**	Deviation of the local non-stochastic (atom-based) quadratic index of order 2 weighted by the atomic weight. This descriptor focuses on the aromatic carbons and their chemical environments.
***D*[*LnsAq*6(*HYD*)*Y*]c_j**	Deviation of the local non-stochastic (atom-based) quadratic index of order 6 weighted by the hydrophobicity. This descriptor focuses on the heteroatoms and their chemical environments.

[a] All these descriptors depend on the chemical structure and the experimental condition c_j; the latter gives information regarding the *q*th measure of biological effect employed in the assay (*me*) and the *r*th target (*tg*) against which the molecules were tested. Thus, c_j can be expressed as c_j(*me*,*tg*).

The ms-QSBER-EL model had *Ac*(%)=91.13% in the training set, meaning that 14 728 out of 16 162 cases were correctly classified, while the same statistical index in the test set was equal to 89.86% (4679 correctly classified cases/chemicals out of 5207). The other statistical indices mentioned in the previous section support the good performance displayed by the ms-QSBER-EL model. Thus, in Table 16.3, for both training and test

sets, *Sn*(%)>85% while *Sp*(%)>90%. Also, the statistic *MCC* is higher than 0.75. Notice that *MCC* can take values from −1 (poor prediction) to +1 (ideal prediction), with the value 0 characterizing a random predictor. Therefore, it can be inferred that in the ms-QSBER-EL model developed here, there is a strong convergence between the observed and predicted values of the variable of biological effect $BE_i(c_j)$. The classification results are available in **Supplementary Information 16.2**.[1]

Table 16.3 Measures of internal and external performance of the ms-QSBER-EL model.

SYMBOLS[a]	Training Set	Test Set
$N_{\mathbf{Positive}}$	6591	2064
$CC_{\mathbf{Positive}}$	5795	1769
***Sn*(%)**	87.92%	85.71%
$N_{\mathbf{Negative}}$	9571	3143
$CC_{\mathbf{Negative}}$	8933	2910
***Sp*(%)**	93.33%	92.59%
MCC	0.816	0.787

[a] $N_{\mathbf{Positive}}$ – Total number of molecules annotated as positive; $N_{\mathbf{Negative}}$ – Total number of molecules annotated as negative; $CC_{\mathbf{Positive}}$ – Molecules correctly classified as positive; $CC_{\mathbf{Negative}}$ – Molecules correctly classified as negative; ***Sn*(%)** – Sensitivity (percentage of molecules correctly classified as positive); ***Sp*(%)** – Specificity (percentage of molecules correctly classified as negative); ***MCC*** – Matthews' correlation coefficient.

One should note that the statistical indices *Sn*(%) and *Sp*(%) reflect only the global performance of the model. The magnitudes $[Sn(\%)]c_j$ and $[Sp(\%)]c_j$ are the local measures of performance of the ms-QSBER-EL model which depends on each specific experimental condition $c_j(me, tg)$. The results obtained from the **Supplementary Information 16.3**[1] indicate that $[Sp(\%)]c_j$>84% in both training and test sets, while $[Sn(\%)]c_j$ exhibited values higher than 75% for most experimental conditions. For only 3 out of the 23 experimental conditions reported in the dataset, the local measure of sensitivity $[Sn(\%)]c_j$ indicated a poor performance. The first two experimental conditions involved cytotoxicity assays (measuring CC_{50})

MRC5 and L929 cells, while the third one focused on determining the half-life in human plasma. In any case, the fact that 20 out of 23 experimental conditions (86.96%) were assessed with high $[Sn(\%)]c_j$ and $[Sp(\%)]c_j$ demonstrates the ability of the ms-QSBER-EL model to integrate and accurately predict antimalarial activity, cytotoxicity, and pharmacokinetic profiles of drugs/chemicals.

16.3.2 Applicability Domain

The reliability of the classifications/predictions performed by the ms-QSBER-EL model was estimated by defining its applicability domain according to the descriptor space approach.[57] Classically, when using the applicability domain defined by the descriptor space, all the compounds in the training set are used, and the maximum and minimum values of each descriptor present in the model demarcate the boundaries that should not be surpassed; any compound beyond such boundaries will be considered to be outside the applicability domain. Here, the applicability domain derived from the descriptor space approach was applied only to those chemicals correctly classified by the ms-QSBER-EL model in the training set (**Supplementary Information 16.3**[1]). Using information regarding the maximum and minimum values of the different molecular descriptors in the ms-QSBER-EL model, local scores were assigned to each molecular descriptor. When the descriptor value of a query molecule was between the aforementioned boundaries, the local score was equal to 1; otherwise, the local score was equal to 0. Subsequently, a total score (TSAD) was defined as the sum of the local scores. As the ms-QSBER-EL model was built from eight molecular descriptors, only those compounds with TSAD=8 were considered to be inside the applicability domain.

16.3.3 Gathering Physicochemical and Structural Information from the Molecular Descriptors

Nowadays, it is a common practice to use computational models to perform predictions of large and heterogeneous chemical libraries. As a result, the physicochemical and structural information contained in the molecular descriptors employed to construct the models is often neglected, and no mechanistic interpretation is provided. The ms-QSBER-EL model developed

in this work predicts very complex data involving several types of biological effects and different targets. Therefore, it is not possible to gain a mechanistic insight regarding the diverse biological and/or biochemical processes that take place when a molecule interacts with a target, either molecular (*e.g.*, proteins) or non-molecular (cells, subcellular components, and *Plasmodium falciparum* strains). However, even though stereochemistry is not considered by the molecular descriptors of the ms-QSBER-EL model, interpreting these descriptors can help find 2D-molecular patterns that are common to the different biological effects regardless of the targets on which they were measured. The purpose here is to interpret the different molecular descriptors present in the ms-QSBER-EL model from a physicochemical and structural point of view so that it will be possible to get a better understanding regarding the features that a chemical should possess to increase its antimalarial activity while decreasing its cytotoxicity and improving its pharmacokinetic profile. In this sense, the interpretations given in this subsection will be based on the combination of four aspects: a) the analysis of the influence of each molecular descriptor in the ms-QSBER-EL model, b) the elucidation of the tendency of variation of the different molecular descriptors, c) the information content of the molecular descriptors, and d) the application of a fragment-based analysis. The first aspect is illustrated in Figure 16.2, where the molecular descriptors are represented according to their sensitivity values (*SV*). The higher the *SV* of a molecular descriptor, the greater will be its influence in the ms-QSBER-EL model.

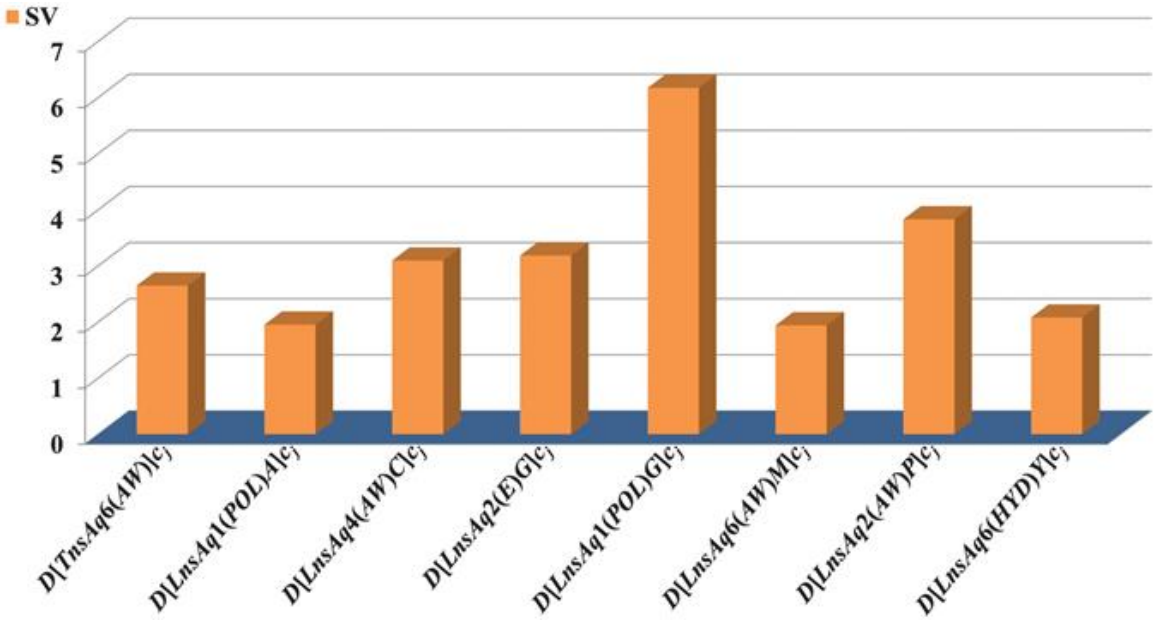


Figure 16.2 Relative importance of the molecular descriptors in the ms-QSBER-EL model.

The second aspect is reported in Table 16.4. It is assumed that, by comparing the class-based mean values of each molecular descriptor, one can estimate how a molecular descriptor should vary to cause an increase of the antimalarial activity, a decrease of the cytotoxicity, and/or enhancement of the pharmacokinetic properties. This approach involving the class-based mean values was first applied by Speck-Planche and coworkers as a way to gather information from the molecular descriptors in non-linear models.[58] It should be emphasized that the aforementioned approach was applied only to the chemicals that were correctly classified in the training set. Basically, for each molecular descriptor, two means are calculated: one for the set of compounds assigned and correctly classified as positive, and the other for the set of chemicals annotated and properly classified as negative. Comparison of the two means for each molecular descriptor permits us to determine if the descriptor value should be increased or diminished. One must always keep in mind that we are dealing here with a non-linear model, and therefore, these tendencies of variations elucidated from the molecular descriptors will be relative. Consequently, the approach used here will probably not contain the entire chemical space that the ms-QSBER-EL model may cover. Nevertheless, the approach is intended to focus on a defined region of the

chemical space where the compounds with the desired properties may be found, in this case, potentially active and safe antimalarial agents.

Table 16.4 Tendency of variation of the molecular descriptors in the ms-QSBER-EL model according to the classes-based means approach.

	Mean Values		
Symbol	Positive	Negative	Tendency[a]
$\boldsymbol{D[TnsAq6(AW)]c_j}$	−0.005	0.116	Decrease
$\boldsymbol{D[LnsAq1(POL)A]c_j}$	−0.011	0.162	Decrease
$\boldsymbol{D[LnsAq4(AW)C]c_j}$	0.004	0.009	Decrease
$\boldsymbol{D[LnsAq2(E)G]c_j}$	0.010	−0.043	Increase
$\boldsymbol{D[LnsAq1(POL)G]c_j}$	−0.002	−0.022	Increase
$\boldsymbol{D[LnsAq6(AW)M]c_j}$	−0.006	0.061	Decrease
$\boldsymbol{D[LnsAq2(AW)P]c_j}$	−0.038	0.343	Decrease
$\boldsymbol{D[LnsAq6(HYD)Y]c_j}$	−0.003	0.062	Decrease

[a] Tendency–It indicates the variation (increase or decrease) in the value of a descriptor that can simultaneously lead to the increase of the antimalarial activity, the decrease of the cytotoxicity, and the improvement of the pharmacokinetic properties.

Regarding the third aspect, as commented before, the ms-QSBER-EL model was built from the deviations of the total and local non-stochastic (atom-based) quadratic indices, which are symbolized as $D[TnsAq_k(x)]c_j$ and $D[LnsAq_k(x)Z]c_j$, respectively. With x being any of the physicochemical atomic properties mentioned above and c_j indicating the different experimental conditions, the topological distance ($d=k$, where k is the *kth* order of the quadratic index). The topological distance is the number of bonds (without counting the bond multiplicity) that exists between any two atoms. The use of this simple but useful concept helps us to rapidly get an accurate idea of how the different types of atoms (*e.g.*, aliphatic and aromatic carbons, halogens, *etc.*) should be distributed throughout the entire structure of a molecule.

The last aspect is focused on the fact that any topological (graph-based) descriptor calculated for a molecule can be represented as a linear

combination of the frequency with which diverse fragments (connected and disconnected) appear in that molecule.[59] Consequently, this property of the topological descriptors allows us to directly define different molecular fragments whose presence contributes positively to variation in the value of any topological index; such fragments can then be used to design new molecules.[60–62] The quadratic indices used in the present study are a class of topological descriptors, and therefore, they comply with the aforementioned property. Thus, while interpreting, we will provide a non-exhaustive but useful list of fragments that are suggested to positively contribute to the corresponding variations of each molecular descriptor in the ms-QSBER-EL model (see Figure 16.3). In this illustration, we have focused on small molecular fragments, and therefore, only the descriptors based on topological distance $d \leq 4$ are represented. No fragment-based information will be provided for the descriptors focused on $d \geq 4$, although the physicochemical and structural interpretations for all of them will be explained.

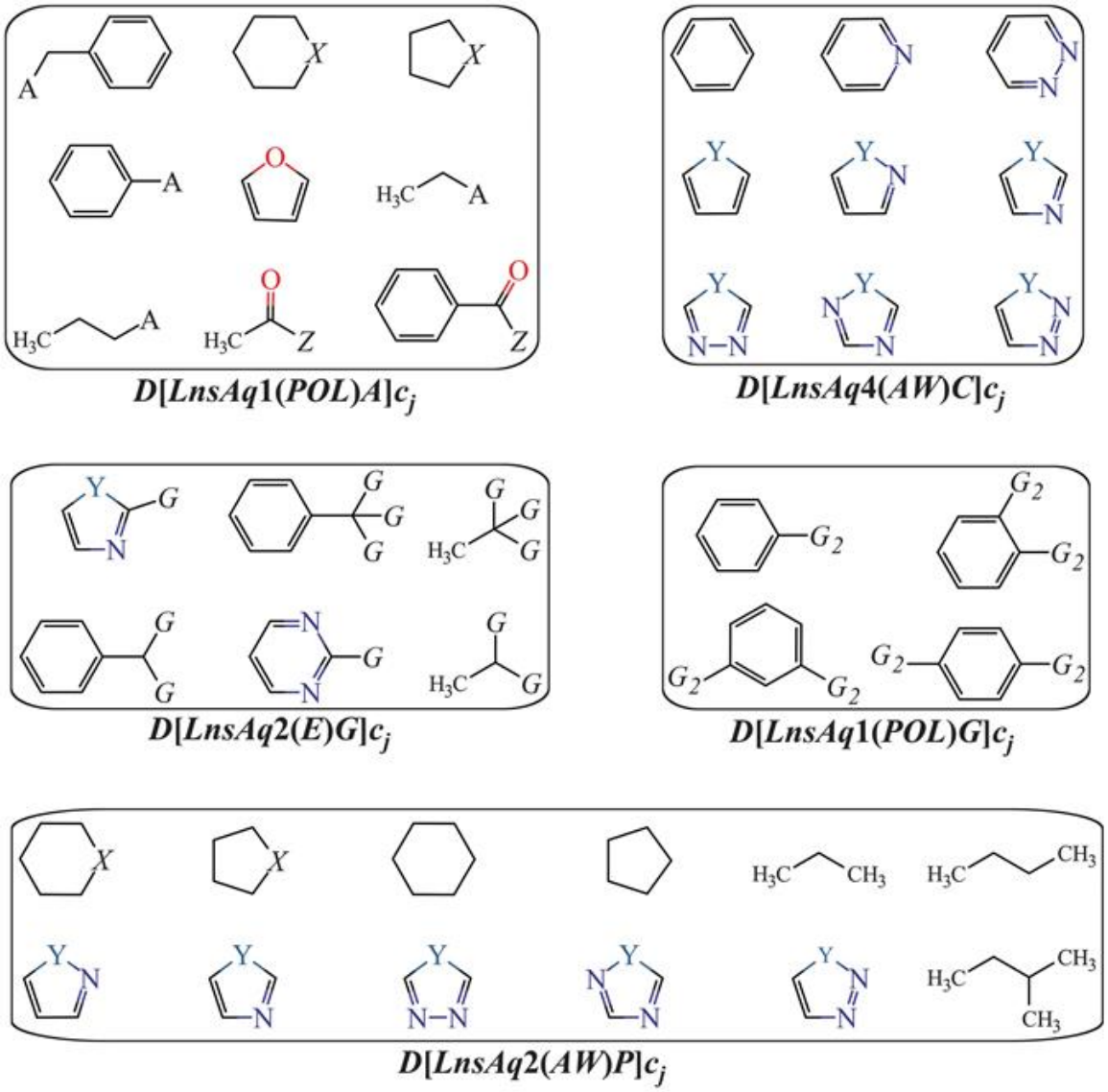


Figure 16.3 Molecular fragments with favorable contributions with respect to the different molecular descriptors in the ms-QSBER-EL model. Here the symbols refer to: A=F, –OH, or $–NH_2$; X=O or –NH–; Z=–OH or $–NH_2$; Y=O or –NH–; G=F, Cl, or Br; G_2=Cl, Br, or I.

Four out of eight descriptors present in the ms-QSBER-EL model are based on the atomic weight, and this already indicates that the most important factors to simultaneously increase the antimalarial activity, decrease the cytotoxicity, and improve the pharmacokinetic profiles are those associated with steric bulk. Notice that, as a property, molecular weight is associated with the size of the molecules. These descriptors based on the

atomic weight will depend on the number of atoms in a molecule; they will have a direct impact on the molecular weight, and consequently, on the size of a molecule. We have $\boldsymbol{D[TnsAq6(AW)]c_j}$, which characterizes the decrease of the atomic weight of any two atoms placed at the topological distance equal to six. This is the fifth most significant descriptor in the model, and it indicates a decrease in the number of atoms in the molecule, while the number of heavy atoms (halogens except fluorine) should be reduced as much as possible. In convergence with $\boldsymbol{D[TnsAq6(AW)]c_j}$, the descriptor $\boldsymbol{D[LnsAq4(AW)C]c_j}$ (the fourth most important) also indicates the diminution of the atomic weight of any two atoms but at a topological distance equal to four, with at least one of them being an aliphatic carbon. This means that the number of aliphatic carbons should be decreased, and if they are present in a molecule, they should be placed at the topological distance of 4 with respect to other atoms with low atomic weight (carbon, nitrogen, or oxygen). Therefore, as illustrated in Figure 16.3, aromatic and heteroaromatic rings are highly desirable to decrease the presence of aliphatic carbons, decreasing the value of $\boldsymbol{D[LnsAq4(AW)C]c_j}$. A third steric descriptor is $\boldsymbol{D[LnsAq6(AW)M]c_j}$, which provides information regarding the diminution of the atomic weight of any two atoms separated by six bonds (topological distance equal to six), with at least one of these atoms being a carbon from a methyl group. This descriptor has the lowest influence in the ms-QSBER-EL model, and its value can be decreased by reducing the number of methyl groups in the molecules. Last, by analyzing $\boldsymbol{D[LnsAq2(AW)P]c_j}$, we have suggested that the atomic weight of any two atoms (at least one of them being an aromatic carbon) placed at the topological distance equal to two should be diminished. Notice that $\boldsymbol{D[LnsAq2(AW)P]c_j}$ is the second most important molecular descriptor, and the presence of aliphatic portions (or rings) and heteroaromatic rings is beneficial for the diminution of the value of $\boldsymbol{D[LnsAq2(AW)P]c_j}$ (see Figure 16.3).

Besides the four quadratic indices based on the atomic weights, two other descriptors also provide steric information. One of them is $\boldsymbol{D[LnsAq1(POL)A]c_j}$, characterizing the diminution of the polarizability of the molecules, which depends on the presence of atoms acting as hydrogen bond acceptors (and their adjacent atoms). This descriptor (the seventh most influential) can be diminished by placing fluorine, nitrogen, and oxygen

atoms in the periphery of the molecules. In the specific case of the nitrogen atoms, those belonging to primary and secondary aliphatic amine are preferred. The other descriptor that confirms the importance of the steric effect is ***D*[*LnsAq*1(*POL*)*G*]*c*$_j$**, focused on the increase of the polarizability of the molecules by considering only halogens and their adjacent atoms. Notice that this is the most important descriptor in the ms-QSBER-EL model (see Figure 16.2), and that its value can be increased by attaching halogens (specifically Cl, Br, and I) to aromatic carbons (see Figure 16.3).

As described by ***D*[*LnsAq*1(*POL*)*G*]*c*$_j$**, the presence of halogens in certain regions of the molecules is crucial for the increase of antimalarial activity, the diminution of cytotoxicity, and the enhancement of the pharmacokinetic properties. This information is supported by ***D*[*LnsAq*2(*E*)*G*]*c*$_j$**, which has the third greatest influence in the ms-QSBER-EL model. This descriptor indicates the increment of the electronegativity in those regions where the topological distance between any two atoms (at least one of them being a halogen) is equal to two.

Finally, we have ***D*[*LnsAq*6(*HYD*)*Y*]*c*$_j$**, which characterizes the decrease of the joint hydrophobic contribution of any two atoms placed at a topological distance equal to six, with one of these atoms being nitrogen, oxygen, sulfur, or phosphorus. This descriptor is the sixth most significant in the ms-QSBER-EL model. It should be pointed out that the term "joint hydrophobic contribution" refers to the product of the hydrophobicity values of any two atoms. In this sense, according to the approach reported by Ghose and Crippen,[63] each atom in the molecule is described by its neighboring atoms; hydrogen and halogen atoms are classified by the hybridization and oxidation states of the carbon atom to which they are attached, and for hydrogen atoms, heteroatoms attached to a carbon atom in the α position are further considered. Carbon atoms are classified by their hybridization state and depending on whether their neighbors are carbon or heteroatoms.[63]

It can be deduced that there are atoms that, depending on their chemical environments, will have a negative hydrophobic contribution, whereas others will have a positive value in that hydrophobicity scale. Examples of atoms with negative contributions are most aliphatic (saturated) carbons, except for those of the form CHX3, CR2X2, CRX3, and CX4, with X being any electronegative atom (O, N, S, P, Se, halogens). Other atoms with negative

hydrophobic contributions are almost all nitrogen atoms except for those belonging to pyrrole, amines with at least two aromatic substituents, and tertiary amines. A similar rule can be applied to the oxygen-containing counterparts of the amines mentioned here. The other atoms not mentioned here will have a positive contribution to the hydrophobicity. For the specific case of aromatic carbons, while Ghose and coworkers report different values of hydrophobicities,[63] here, the programmers of the software QuBiLS-MAS v1.0 set the hydrophobic contribution to zero for these types of atoms. That being said, a few examples of pairs of atoms (at the topological distance equal to six) whose joint hydrophobic contributions diminish the value of $\boldsymbol{D[LnsAq6(HYD)Y]c_j}$ are G with any nitrogen or oxygen (excluding the previous exceptions), where G represents halogen, sulfur, selenium, or phosphorus.

Altogether, the information provided by the molecular descriptors suggests that the presence of halogens is mandatory in the structure of the molecules, but they must be confined to the peripheral regions. In addition, judging by the decrease of the atomic weight characterized by four of the eight descriptors, halogens may be present in the following amount: 1) one iodine, 2) two chlorine, 3) two bromine, 4) one chlorine and one bromine, 5) one fluorine and one bromine, or 6) one fluorine, one chlorine, and one bromine. The presence of heteroaromatic rings substituted with halogen is crucial because they decrease the number of aromatic carbons (decreasing the value of $\boldsymbol{D[LnsAq2(AW)P]c_j}$), and at the same time, they increase the values of the descriptors $\boldsymbol{D[LnsAq1(POL)G]c_j}$ and/or $\boldsymbol{D[LnsAq2(E)G]c_j}$. Examples of such fragments are 2-halopyridines, 2-halopyrimidines, and 2-haloazoles. On the other hand, the molecules should not have methyl groups, and aliphatic carbons may be used linkers between the heteroaromatic portions.

16.3.4 Computational Design of Virtually Potent and Safe Antimalarial Chemicals

When predicting external chemical libraries, computational models can perform reliable predictions to some extent. This is because the experimental data often contain uncertainties regarding the conditions under which the assays were carried out. Other factors preventing the full exploitation of the computational models as virtual screening tools are the lack of consensus

regarding the definition of the applicability domain, and the non-universality of the molecular descriptors; the latter means that the molecular descriptors can characterize only a reduced fraction of the highly diverse and complex chemical space.

It is also important to keep in mind that most of the models will provide predictions according to predetermined values, which correspond to the descriptors of the molecules that are being predicted. In this context, most of these molecules may not follow the rules of the model. This has nothing to do with the applicability domain or any other method to estimate the reliability of the predictions; it is associated with the fact that in any model, the molecular descriptors in the training set have different degrees of hierarchy, and these embody a phenomenological meaning in the sense of describing the physicochemical and structural features that need to be primarily modified for improving the biological effect under study. Therefore, compounds whose molecular descriptors follow such a degree of the hierarchy/importance in the model are expected to be predicted with a higher accuracy, where the theoretical predictions should better match the experimental results. Although considering the degree of the hierarchy of the molecular descriptors is an aspect neglected by most of the current models devoted to screening vast datasets, recent approaches focused on fragment-based topological design have emphasized its essential role for the optimization of the chemical space.[61,64–67] As the molecular descriptors of a predictive computational model are not able to "control" the chemical space to be predicted, it has been proposed that in order to achieve a higher certainty in the predictions, the chemicals must be designed according to the physicochemical and structural interpretations of the molecular descriptors in the models.[61,64–67]

To prove the veracity of such suggestions and estimate the sensitivity of the ms-QSBER-EL model with respect to small changes in the chemical structures of the molecules, we carried out the following experiment. We designed ten different molecules belonging to different chemical families (see Figure 16.4) according to the following steps. First, for each designed molecule, some of the fragments mentioned above were used as building blocks or seeds. Second, the designed molecules were assembled by fusing or connecting the fragments. Third, when needed, a series of bioisosteric replacements were made, or some atoms or functional groups were added.

Last, a series of molecules were designed by rigorously considering the joint interpretation of the molecular descriptors in the ms-QSBER-EL model (from AMA-001 to AMA-004, AMA-008, and AMA-010), while others were designed from the interpretations retrieved from some of the descriptors (AMA-005 to AMA-007, and AMA-009).

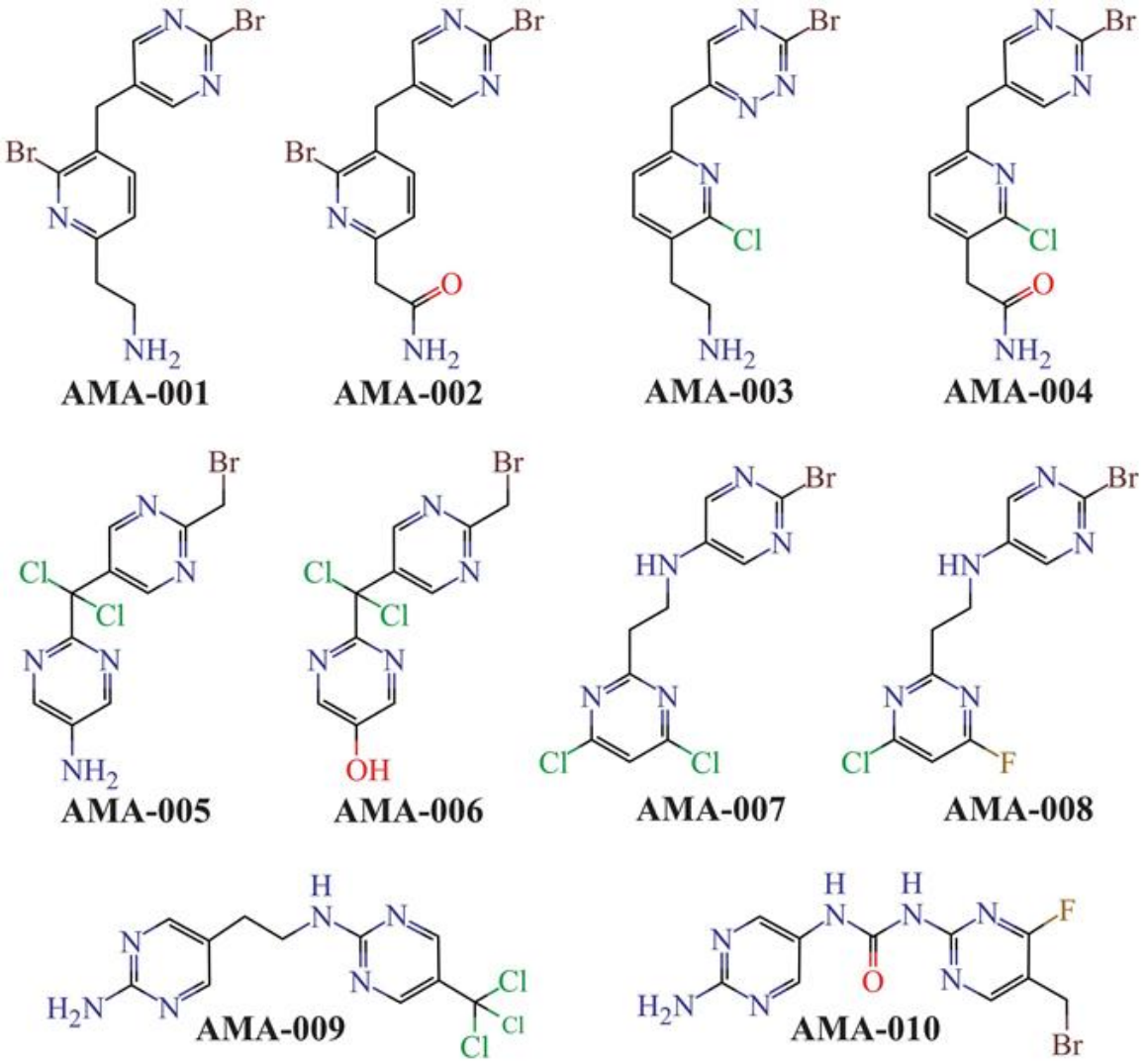


Figure 16.4 Chemical structures of the designed molecules.

The results of the predictions (applicability domain included) for all the designed molecules are reported in **Supplementary Information 16.4**.[1] In addition, Table 16.5 illustrates a summary of such predictions, which permits a fast comparison of the different biological profiles of the different

molecules. As expected, in general, the molecules from AMA-001 to AMA-004 were predicted as very potent and safe antimalarial agents. Nevertheless, we would like to highlight that although excellent predicted biological profiles of these four molecules were very similar between each other, the molecules AMA-002 and AMA-004 were predicted as negative against the cytotoxicity assay performed on RAW264.7 cells. This happened because, in the case of AMA-002, the presence of the carboxamide group is detrimental to $\boldsymbol{D[TnsAq6(AW)]c_j}$, increasing the value of this descriptor (which must decrease). A similar situation can be found in the molecule AMA-004, which in addition, has pyrimidine instead of 1,2,4-triazine; this slightly increases the values of $\boldsymbol{D[LnsAq2(AW)P]c_j}$ and $\boldsymbol{D[LnsAq6(HYD)Y]c_j}$ (they both must decrease).

Table 16.5 Predictions performed by the ms-QSBER-EL model for the designed molecules.

Molecule ID	*me*[a]	*tg*	$\boldsymbol{Pred_BE_i(c_j)^b}$
AMA-001 and AMA-003	IC50 (nM)	*Plasmodium falciparum* (3D7)	1
AMA-001 and AMA-003	IC50 (nM)	*Plasmodium falciparum* (K1)	1
AMA-001 and AMA-003	IC50 (nM)	*Plasmodium falciparum* (W2)	1
AMA-001 and AMA-003	Permeability (ucm s^{-1})	Caco-2	1
AMA-001 and AMA-003	PPB (%)	Plasma (Human)	1
AMA-001 and AMA-003	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	1
AMA-001 and AMA-003	$t_{1/2}$ (h)_Plm	Plasma (Human)	1
AMA-001 and AMA-003	CC50 (nM)	HEK293	1
AMA-001 and AMA-003	CC50 (nM)	RAW264.7	1
AMA-001 and	IC50	HERG	1

AMA-003	(nM)_HERG		
AMA-002 and AMA-004	IC50 (nM)	*Plasmodium falciparum* (3D7)	1
AMA-002 and AMA-004	IC50 (nM)	*Plasmodium falciparum* (K1)	1
AMA-002 and AMA-004	IC50 (nM)	*Plasmodium falciparum* (W2)	1
AMA-002 and AMA-004	Permeability (ucm s^{-1})	Caco-2	1
AMA-002 and AMA-004	PPB (%)	Plasma (Human)	1
AMA-002 and AMA-004	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	1
AMA-002 and AMA-004	$t_{1/2}$ (h)_Plm	Plasma (Human)	1
AMA-002 and AMA-004	CC50 (nM)	HEK293	1
AMA-002 and AMA-004	CC50 (nM)	RAW264.7	−1
AMA-002 and AMA-004	IC50 (nM)_HERG	HERG	1
AMA-005 and AMA-006	IC50 (nM)	*Plasmodium falciparum* (3D7)	−1
AMA-005 and AMA-006	IC50 (nM)	*Plasmodium falciparum* (K1)	−1
AMA-005 and AMA-006	IC50 (nM)	*Plasmodium falciparum* (W2)	−1
AMA-005 and AMA-006	Permeability (ucm s^{-1})	Caco-2	1
AMA-005 and AMA-006	PPB (%)	Plasma (Human)	−1
AMA-005 and AMA-006	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	−1
AMA-005 and AMA-006	$t_{1/2}$ (h)_Plm	Plasma (Human)	−1
AMA-005 and	CC50 (nM)	HEK293	−1

AMA-006			
AMA-005 and AMA-006	CC50 (nM)	RAW264.7	−1
AMA-005 and AMA-006	IC50 (nM)_HERG	HERG	−1
AMA-007	IC50 (nM)	*Plasmodium falciparum* (3D7)	1
AMA-007	IC50 (nM)	*Plasmodium falciparum* (K1)	−1
AMA-007	IC50 (nM)	*Plasmodium falciparum* (W2)	1
AMA-007	Permeability (ucm s^{-1})	Caco-2	1
AMA-007	PPB (%)	Plasma (Human)	−1
AMA-007	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	1
AMA-007	$t_{1/2}$ (h)_Plm	Plasma (Human)	−1
AMA-007	CC50 (nM)	HEK293	−1
AMA-007	CC50 (nM)	RAW264.7	1
AMA-007	IC50 (nM)_HERG	HERG	1
AMA-008	IC50 (nM)	*Plasmodium falciparum* (3D7)	1
AMA-008	IC50 (nM)	*Plasmodium falciparum* (K1)	1
AMA-008	IC50 (nM)	*Plasmodium falciparum* (W2)	1
AMA-008	Permeability (ucm s^{-1})	Caco-2	1
AMA-008	PPB (%)	Plasma (Human)	1
AMA-008	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	1
AMA-008	$t_{1/2}$ (h)_Plm	Plasma (Human)	−1
AMA-008	CC50 (nM)	HEK293	1
AMA-008	CC50 (nM)	RAW264.7	1
AMA-008	IC50	HERG	1

	(nM)_HERG		
AMA-009	IC50 (nM)	*Plasmodium falciparum* (3D7)	−1
AMA-009	IC50 (nM)	*Plasmodium falciparum* (K1)	−1
AMA-009	IC50 (nM)	*Plasmodium falciparum* (W2)	−1
AMA-009	Permeability (ucm s^{-1})	Caco-2	−1
AMA-009	PPB (%)	Plasma (Human)	−1
AMA-009	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	1
AMA-009	$t_{1/2}$ (h)_Plm	Plasma (Human)	−1
AMA-009	CC50 (nM)	HEK293	−1
AMA-009	CC50 (nM)	RAW264.7	1
AMA-009	IC50 (nM)_HERG	HERG	−1
AMA-010	IC50 (nM)	*Plasmodium falciparum* (3D7)	1
AMA-010	IC50 (nM)	*Plasmodium falciparum* (K1)	1
AMA-010	IC50 (nM)	*Plasmodium falciparum* (W2)	1
AMA-010	Permeability (ucm s^{-1})	Caco-2	1
AMA-010	PPB (%)	Plasma (Human)	1
AMA-010	$t_{1/2}$ (h)_Lvm	Liver microsomes (Human)	1
AMA-010	$t_{1/2}$ (h)_Plm	Plasma (Human)	−1
AMA-010	CC50 (nM)	HEK293	1
AMA-010	CC50 (nM)	RAW264.7	1
AMA-010	IC50 (nM)_HERG	HERG	1

[a] The definitions for the different measures of biological effects are provided in Table 1. [b]The symbol$\boldsymbol{Pred_BE_i(c_j)}$ refers to the predicted value of the categorical variable of biological effect

[$BE_i(c_j)$] under a given experimental condition c_j.

The unfavorable predictions for AMA-005 and AMA-006 are not surprising at all. First, the two halogen atoms appear in a central region instead of being restricted to the periphery of the molecules. These halogens detrimentally increase the value $D[TnsAq6(AW)]c_j$. At the same time, these two halogen atoms are attached to aliphatic carbons, in contrast to the recommendation of attaching them to aromatic carbons, which would favorably increase the value of $D[LnsAq1(POL)G]c_j$. Another important fact is that electronegative atoms such as nitrogen and oxygen appear placed at the topological distance different from two, which decreases the value of the descriptor $D[LnsAq2(E)G]c_j$.

The difference between the predicted biological profiles of AMA-007 and AMA-008 is due to the presence of a fluorine atom in AMA-008, which favorably increases the value of $D[LnsAq2(E)G]c_j$ while also desirably decreasing $D[TnsAq6(AW)]c_j$. In the case of AMA-009, as recommended, the three halogen atoms are in the periphery of the molecules. However, the presence of these three chlorine atoms leads to a noticeable increase in the value of $D[TnsAq6(AW)]c_j$. Notice that according to the joint physicochemical and structural interpretation of the molecular descriptors, only two chlorine atoms are recommended instead of three. Finally, AMA-010 is very different from the other nine molecules, and yet, it was predicted to exhibit high activity, low cytotoxicity, and good pharmacokinetic properties except for its half-life against human plasma. Perhaps a simple modification such as switching the positions of the halogens may lead to an analog of AMA-010 with improvements in the predicted biological profiles.

The differences between the predicted biological effects of the designed molecules demonstrate the sensitivity and discriminatory power of the molecular descriptors used to develop the ms-QSBER-EL model. On the other hand, it is clear that each prediction of the biological effect of a molecule does not depend on a single descriptor but in the combination of all the descriptors present in the model. In any case, some physicochemical properties and structural features have more influence than others, and we have explained them by providing the specific descriptors responsible for the

observed differences in the chemical structures of the molecules, which translated in different predicted biological profiles.

The most promising molecules are from AMA-001 to AMA-004, AMA-008, and AMA-010. Because of the potentially high antimalarial activity, low cytotoxicity, and adequate pharmacokinetic properties, we searched these molecules in well-known databases such as ChEMBL, ZINC,[68] and ChemSpider[69] for finding similar molecules. For a similarity cutoff value ≥0.8, only one match was found. In this sense, the molecule ChemSpiderID 51375502 was identified to be similar to AMA-001. However, different structural aspects support the chemical dissimilarity of these two molecules. For instance, ChemSpiderID 51375502 contains a 4-bromopyrimidine ring (instead of the crucial fragment 2-bromopyrimidine reported for AMA-001). Also, ChemSpiderID 51375502 is lacking a second bromine atom in the pyridine ring, instead having an ethyl group. In the end, from the point of view of the physicochemical and structural information extracted from the ms-QSBER-EL model, the 4-bromopyrimidine ring in ChemSpiderID 51375502 is slightly detrimental for the increase of the descriptor $\boldsymbol{D[LnsAq2(E)G]c_j}$. At the same time, the lack of the second bromine atom also unfavorably affects the values of $\boldsymbol{D[LnsAq2(E)G]c_j}$ and $\boldsymbol{D[LnsAq6(HYD)Y]c_j}$. Finally, the presence of the methyl moiety in the ethyl group increases the value of $\boldsymbol{D[LnsAq6(AW)M]c_j}$, whose undesirable effect has been explained before. Thus, in principle, ChemSpiderID 51375502, although promising, should be slightly less desirable than AMA-001 in terms of antimalarial activity and/or safety.

16.3.5 Druglikeness

The druglikeness of each molecule was assessed by verifying the compliance with Lipinski's rule of five,[70] which establishes that, in order to exhibit a good oral bioavailability, a molecule should have no more than 5 hydrogen bond donors (HBD), 10 or fewer hydrogen bond acceptors (HBA), a molecular weight (MW) lower than 500 Da, and an octanol-water partition coefficient (logP) not greater than 5. A variant of the Lipinski's rule of five suggests logP in the range −0.4 to +5.6, molar refractivity in the interval 40–130 $cm^3\ mol^{-1}$, MW in the range 180–480 Da, and the number of atoms (nAT) from 20 to70. In addition, Veber and coworkers suggested that only

two aspects should be considered when analyzing the potentially good oral bioavailability of a chemical: the polar surface area (PSA) to be lower than ≤140 Å^2, and the number of rotatable bonds (RBN) to be ≤10.[71] All the properties mentioned above were calculated for the six most promising molecules mentioned above (see Table 16.6) by using the software AlvaDesc v1.0.14.[72] From the physicochemical point of view, all the designed molecules were in agreement with the Lipinski's rule of five and its variants.

Table 16.6 Molecular properties calculated for the designed molecules.

ID[a]	HBD	HBA	MW (Da)	MlogP	AlogP	nAT	MR (cm^3 mol^{-1})	RBN	PSA (Å^2)
AMA-001	2	4	372.08	2.082	2.392	30	79.516	4	64.69
AMA-002	2	5	386.06	1.621	2.288	29	79.623	4	81.76
AMA-003	2	5	328.62	1.719	2.109	29	76.562	4	77.58
AMA-004	2	5	341.61	1.493	1.757	29	76.972	4	81.76
AMA-008	1	6	332.58	2.123	2.226	26	71.097	4	63.59
AMA-010	4	9	342.16	1.116	0.592	29	74.494	2	118.71

[a] The following symbols are: HBD – number of hydrogen bond donors; HBA – number of hydrogen bond acceptors; MW – molecular weight; MlogP – logarithm of the partition coefficient (octanol/water) estimated according to the Moriguchi's approach; AlogP – logarithm of the partition coefficient (octanol/water) estimated according to the Ghose-Crippen's approach; nAT – number of atoms; MR – molar refractivity; RBN – number of rotatable bonds; PSA – polar surface area considering only nitrogen and oxygen atoms.

16.4 Conclusion

Malaria is an emerging and unresolved public health problem that can be halted by using the most advanced *in silico* approaches in drug discovery. In this context, computational tools are essential for prioritizing the search for

novel, potent, and safe antimalarial agents. The predictive models reported to date should go beyond the conventional purpose of screening large chemical libraries. Special attention must be paid to the physicochemical and structural information that can be gathered from the *in silico* models. The ms-QSBER-EL model developed here is capable of integrating antimalarial activity, cytotoxicity, and pharmacokinetic properties, exhibiting a very good accuracy. The combined use of the physicochemical and structural interpretations of the molecular descriptors and the fragment-based analysis permitted the guided design of virtually new molecules as potent and safe antimalarial chemicals, which are now available to the scientific community as potential leads for future biological testing. This work unveils new horizons for the creation of focused libraries for drug discovery against malaria, with a wide range of applications to other tropical diseases.

Acknowledgements

A. Speck-Planche acknowledges all the experimental scientists working in the fields of malaria, toxicological sciences, and pharmacokinetics, whose works have served as inspiration for the compilation of the database that led to the creation of the present ms-QSBER-EL model.

References

1. A. Lavecchia, *Drug Discovery Today*, 2015, **20**, 318.
2. Y. C. Lo, S. E. Rensi, W. Torng and R. B. Altman, *Drug Discovery Today*, 2018, **23**, 1538.
3. A. N. Lima, E. A. Philot, G. H. Trossini, L. P. Scott, V. G. Maltarollo and K. M. Honorio, *Expert Opin. Drug Discovery*, 2016, **11**, 225.
4. L. Zhang, J. Tan, D. Han and H. Zhu, *Drug Discovery Today*, 2017, **22**, 1680.
5. E. Putin, A. Asadulaev, Y. Ivanenkov, V. Aladinskiy, B. Sanchez-Lengeling, A. Aspuru-Guzik and A. Zhavoronkov, *J. Chem. Inf. Model.*, 2018, **58**, 1194.

6. R. Gomez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernandez-Lobato, B. Sanchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2018, **4**, 268.
7. M. H. S. Segler, T. Kogej, C. Tyrchan and M. P. Waller, *ACS Cent. Sci.*, 2018, **4**, 120.
8. A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper and A. Zhavoronkov, *Mol. Pharm.*, 2017, **14**, 3098.
9. W. Jahnke and D. A. Erlanson, *Fragment-based Approaches in Drug Discovery*, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, Germany, 2006.
10. A. Speck-Planche and M. N. D. S. Cordeiro, *Expert Opin. Drug Discovery*, 2015, **10**, 245.
11. A. Speck-Planche and M. N. D. S. Cordeiro, in *Bladder Cancer: Risk Factors, Emerging Treatment Strategies and Challenges*, ed. S. Haggerty, Nova Science Publishers, Inc., New York, **ch. 4**, 2014, pp. 71.
12. V. V. Kleandrova, J. M. Ruso, A. Speck-Planche and M. N. D. S. Cordeiro, *ACS Comb. Sci.*, 2016, **18**, 490.
13. A. Speck-Planche and M. N. D. S. Cordeiro, *Mini-Rev. Med. Chem.*, 2015, **15**, 194.
14. A. Speck-Planche and M. N. D. S. Cordeiro, *Comb. Chem. High Throughput Screening*, 2015, **18**, 305.
15. A. Speck-Planche and M. N. D. S. Cordeiro, *Future Med. Chem.*, 2014, **6**, 2013.
16. A. Speck-Planche and M. N. D. S. Cordeiro, *Curr. Drug Metab.*, 2014, **15**, 429.
17. A. Speck-Planche and M. N. D. S. Cordeiro, *ACS Comb. Sci.*, 2014, **16**, 78.
18. D. M. Herrera-Ibata, A. Pazos, R. A. Orbegozo-Medina, F. J. Romero-Duran and H. Gonzalez-Diaz, *BioSystems*, 2015, **132–133**, 20.
19. D. M. Herrera-Ibata, R. A. Orbegozo-Medina and H. Gonzalez-Diaz, *Curr. Bioinf.*, 2015, **10**, 639.
20. D. M. Herrera-Ibata, A. Pazos, R. A. Orbegozo-Medina and H. Gonzalez-Diaz, *Chemom. Intell. Lab. Syst.*, 2014, **138**, 161.
21. H. Gonzalez-Diaz, D. M. Herrera-Ibata, A. Duardo-Sanchez, C. R. Munteanu, R. A. Orbegozo-Medina and A. Pazos, *J. Chem. Inf. Model.*,

2014, **54**, 744.
22. E. Vasquez-Dominguez, V. D. Armijos-Jaramillo, E. Tejera and H. Gonzalez-Diaz, *Mol. Pharm.*, 2019, **16**, 4200.
23. R. Concu, V. V. Kleandrova, A. Speck-Planche and M. Cordeiro, *Nanotoxicology*, 2017, **11**, 891.
24. A. Speck-Planche, V. V. Kleandrova, F. Luan and M. N. D. S. Cordeiro, *Nanomedicine*, 2015, **10**, 193.
25. F. Luan, V. V. Kleandrova, H. Gonzalez-Diaz, J. M. Ruso, A. Melo, A. Speck-Planche and M. N. D. S. Cordeiro, *Nanoscale*, 2014, **6**, 10623.
26. V. V. Kleandrova, F. Luan, H. Gonzalez-Diaz, J. M. Ruso, A. Speck-Planche and M. N. D. S. Cordeiro, *Environ. Sci. Technol.*, 2014, **48**, 14686.
27. V. V. Kleandrova, F. Luan, H. Gonzalez-Diaz, J. M. Ruso, A. Melo, A. Speck-Planche and M. N. D. S. Cordeiro, *Environ. Int.*, 2014, **73C**, 288.
28. S. G. Martinez-Arzate, E. Tenorio-Borroto, A. Barbabosa Pliego, H. M. Diaz-Albiter, J. C. Vazquez-Chagoyan and H. Gonzalez-Diaz, *J. Proteome Res.*, 2017, **16**, 4093.
29. E. Tenorio-Borroto, N. Castanedo, X. Garcia-Mera, K. Rivadeneira, J. C. Vazquez Chagoyan, A. Barbabosa Pliego, C. R. Munteanu and H. Gonzalez-Diaz, *Chem. Res. Toxicol.*, 2019, **32**, 1811.
30. E. Tenorio-Borroto, F. R. Ramirez, A. Speck-Planche, M. N. D. S. Cordeiro, F. Luan and H. Gonzalez-Diaz, *Curr. Drug Metab.*, 2014, **15**, 414.
31. E. Tenorio-Borroto, C. G. Penuelas-Rivas, J. C. Vasquez-Chagoyan, N. Castanedo, F. J. Prado-Prado, X. Garcia-Mera and H. Gonzalez-Diaz, *Eur. J. Med. Chem.*, 2014, **72**, 206.
32. F. J. Romero-Duran, N. Alonso, M. Yanez, O. Caamano, X. Garcia-Mera and H. Gonzalez-Diaz, *Neuropharmacology*, 2016, **103**, 270.
33. F. J. Romero Duran, N. Alonso, O. Caamano, X. Garcia-Mera, M. Yanez, F. J. Prado-Prado and H. Gonzalez-Diaz, *Int. J. Mol. Sci.*, 2014, **15**, 17035.
34. F. Luan, M. N. D. S. Cordeiro, N. Alonso, X. Garcia-Mera, O. Caamano, F. J. Romero-Duran, M. Yanez and H. Gonzalez-Diaz, *Bioorg. Med. Chem.*, 2013, **21**, 1870.
35. J. Ferreira da Costa, D. Silva, O. Caamano, J. M. Brea, M. I. Loza, C. R. Munteanu, A. Pazos, X. Garcia-Mera and H. Gonzalez-Diaz, *ACS*

Chem. Neurosci., 2018, **9**, 2572.
36. A. Speck-Planche, *Future Med. Chem.*, 2018, **10**, 2021.
37. A. Speck-Planche and M. N. Dias Soeiro Cordeiro, *ACS Comb. Sci.*, 2017, **19**, 501.
38. WHO, 2017, https://www.who.int/malaria/publications/world.
39. C. Wongsrichanalai, A. L. Pickard, W. H. Wernsdorfer and S. R. Meshnick, *Lancet Infect. Dis.*, 2002, **2**, 209.
40. W. R. Taylor and N. J. White, *Drug Saf.*, 2004, **27**, 25.
41. A. Speck-Planche and M. N. D. S. Cordeiro, *Curr. Top. Med. Chem.*, 2013, **13**, 1656.
42. A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani and J. P. Overington, *Nucleic Acids Res.*, 2012, **40**, D1100.
43. N. Y. Mok and R. Brenk, *J. Chem. Inf. Model.*, 2011, **51**, 2449.
44. J. Overington, *J. Comput.-Aided Mol. Des.*, 2009, **23**, 195.
45. A. Speck-Planche, V. V. Kleandrova and M. N. D. S. Cordeiro, *Bioorg. Med. Chem.*, 2013, **21**, 2727.
46. Y. Marrero-Ponce, R. Medina-Marrero, F. Torrens, Y. Martinez, V. Romero-Zaldivar and E. A. Castro, *Bioorg. Med. Chem.*, 2005, **13**, 2881.
47. J. R. Valdes-Martini, Y. Marrero-Ponce, C. R. Garcia-Jacas, K. Martinez-Mayorga, S. J. Barigye, Y. S. Vaz d'Almeida, H. Pham-The, F. Perez-Gimenez and C. A. Morell, *J. Cheminf.*, 2017, **9**, 35.
48. R. Todeschini and V. Consonni, *Molecular Descriptors for Chemoinformatics*, WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, 2009.
49. ChemAxon10030, Budapest, Hungary, v19.18.0, https://www.chemaxon.com, 1998-2019.
50. J. R. Valdés-Martini, C. R. García-Jacas, Y. Marrero-Ponce, Y. Silveira Vaz ‘d Almeida and C. Morell, Villa Clara, Cuba, v1.0, http://tomocomd.com/, 2012.
51. A. Speck-Planche, V. V. Kleandrova and M. N. D. S. Cordeiro, *Eur. J. Pharm. Sci.*, 2013, **48**, 812.
52. H. Bediaga, S. Arrasate and H. Gonzalez-Diaz, *ACS Comb. Sci.*, 2018, **20**, 621.

53. TIBCO-Software-Inc., Palo Alto, California, USA, v13.5.0.17, http://tibco.com, 2018.
54. B. W. Matthews, *Biochim. Biophys. Acta*, 1975, **405**, 442.
55. K. Pearson, *Proc. R. Soc. London*, 1895, **58**, 240.
56. Z.-H. Zhou, J. Wu and W. Tang, *Artif. Intell.*, 2002, **137**, 239.
57. F. Sahigara, K. Mansouri, D. Ballabio, A. Mauri, V. Consonni and R. Todeschini, *Molecules*, 2012, **17**, 4791.
58. A. Speck-Planche and V. V. Kleandrova, *Curr. Top. Med. Chem.*, 2012, **12**, 1734.
59. I. I. Baskin, M. I. Skvortsova, I. V. Stankevich and N. S. Zefirov, *J. Chem. Inf. Comput. Sci.*, 1995, **35**, 527.
60. A. Speck-Planche and M. T. Scotti, *Mol. Diversity*, 2019, **23**, 555.
61. A. Speck-Planche, *ACS Omega*, 2019, **4**, 3122.
62. V. V. Kleandrova and A. Speck-Planche, in *Multi-Scale Approaches in Drug Discovery*, ed. A. Speck-Planche, Elsevier, 2017, pp. 55.
63. A. K. Ghose, V. N. Viswanadhan and J. J. Wendoloski, *J. Phys. Chem. A*, 1998, **102**, 3762.
64. A. Speck-Planche, *ACS Omega*, 2018, **3**, 14704.
65. A. Speck-Planche and M. N. D. S. Cordeiro, in *Multi-Scale Approaches in Drug Discovery*, ed. A. Speck-Planche, Elsevier, 2017, pp. 127.
66. A. Speck-Planche and M. Cordeiro, *Mol. Diversity*, 2017, **21**, 511.
67. A. Speck-Planche and M. N. D. S. Cordeiro, *Med. Chem. Res.*, 2017, **26**, 2345.
68. J. J. Irwin and B. K. Shoichet, *J. Chem. Inf. Model.*, 2005, **45**, 177.
69. A. J. Williams, in *Collaborative Computational Technologies for Biomedical Research*, ed. S. Ekins, M. A. Z. Hupcey and A. J. Williams, John Wiley & Sons, Inc., Hoboken, NJ, ch. 22, 2011, pp. 363.
70. C. A. Lipinski, F. Lombardo, B. W. Dominy and P. J. Feeney, *Adv. Drug Delivery Rev.*, 2001, **46**, 3.
71. D. F. Veber, S. R. Johnson, H. Y. Cheng, B. R. Smith, K. W. Ward and K. D. Kopple, *J. Med. Chem.*, 2002, **45**, 2615.
72. Alvascience-Srl.10069, v1.0.14, https://www.alvascience.com/, 2019.

---

† Electronic supplementary information (ESI) available. See DOI 10.1039/9781839160233.

CHAPTER 17

Machine Learning for Core-loss Spectrum

T. MIZOGUCHI*[a] AND S. KIYOHARA[a,b]

[a] The University of Tokyo, Institute of Industrial Science, Komaba, Tokyo 153-8505, Japan
[b] Laboratory for Materials and Structures, Institute of Innovative Research, Tokyo Institute of Technology, Yokohama 226-8503, Japan
*Email: teru@iis.u-tokyo.ac.jp

17.1 Introduction

Unveiling the atomistic configurations, chemical bonding, and vibrational behaviors of molecules and atoms, which are commonly correlated to the functions of a material, is the most important task of materials research. Spectroscopic techniques, such as diffraction, reflection, emission, and absorption, have been used to identify such information because the spectral features are correlated to the material information.

Among existing spectroscopic techniques, core-loss spectroscopy using electrons or X-rays, namely, electron energy-loss near-edge structure (ELNES) and X-ray absorption near-edge structure (XANES), offer atomic-scale spatial resolution,[1] nanosecond-level time resolution,[2] and high sensitivity[3] and can be considered the "ultimate analysis" in materials science.[4] ELNES and XANES commonly originate from an electron

transition from a core orbital to unoccupied states.[5,6] As their electron transitions commonly follow an electron dipole transition rule, their spectral features reflect the partial density of states (PDOS) in unoccupied bands. Thus, ELNES and XANES features provide information on the local coordination and chemical bonding of selected elements in the illuminated area.

ELNES and XANES are used as complementary tools. XANES has the advantage of sensitivity, *i.e.*, it can be observed from an element present at ppm-levels,[3,7,8] while the main advantage of ELNES is its spatial resolution, *i.e.*, it can be observed from a single atomic column (or even a single atom or buried dopant).[1,9–11] The recent emergence of monochromator systems has improved the energy resolution of EELS, which facilitates the identification of vibrational spectra using TEM/STEM systems.[12–15] Thus, ELNES/XANES have been used for the characterization of a wide variety of materials systems, such as semiconductors,[16–19] ceramics,[20–22] batteries,[23,24] electrical devices,[25–27] and capacitor materials.[28–31]

Instrument development has accompanied advancements in the theoretical simulation of ELNES/XANES. The most important aspect of ELNES/XANES simulation is the introduction of a core hole, which is generated in the core orbital during the electron transition process.[32,33] Depending on the position of the core state and excited electron, ELNES/XANES simulation can be categorized into three methods: one-particle method,[34–37] two-particle method,[38–41] and multi-particle method.[42–45] The fundamentals of ELNES/XANES simulation using the one-, two-, and multi-particle methods have been reviewed elsewhere.[46–49] In addition to spectrum simulation, theoretical tools for unveiling the relationships between spectral features and material properties have also been proposed.[50,51] By combining experimental and theoretical simulations, one can investigate the valence state,[52] local coordination,[37] liquid/gas molecular vibration,[53–55] lattice vibration,[56,57] and van der Waals interaction.[58]

Recent years have witnessed a rapid increase in spectral data. ELNES/XANES experiments often involve time-resolved and/or spatially resolved observations. For instance, spatially resolved ELNES allows

spectral imaging with a resolution of several thousand points in an objective area. In XANES, pico- or nano-second time-resolved observations are possible, and time-resolved experiments for tracing chemical reactions have been performed. Through such spatially/temporally resolved ELNES/XANES observations, up to tens of thousands of spectra can be observed in an experimental session. Individual interpretation through theoretical calculations is infeasible because each ELNES/XANES calculation not only requires extensive experimental and theoretical knowledge but also involves many computations.

Recently, approaches for gaining new insights from big data, namely data-driven approaches, have attracted considerable attention in materials science.[59–63] Analysis based on machine learning (ML) can overcome the limitations of manual multi-dimensional big data analysis. ML is extensively used for materials simulations, characterizations, and discoveries. Such data-driven approaches have facilitated the discovery of desired structures with minimum computation,[64,65] new materials with high efficiency,[66–69] and information that could not be previously determined through experiments[70,71] or simulations.[72,73] ML has also been applied to experimental observations.[74–76] It allows the interpretation of many more spectra compared with manual analysis, as well as the prediction of a scalar value or discrete values in a simple spectrum, such as a chemical shift or peaks in NMR spectra.[74–76] In addition, it has been applied to X-ray absorption fine structure (XAFS) and extended XAFS (EXAFS) spectra.[71,77]

We have reported the applications of ML for materials characterizations. In particular, we have used ML to investigate crystalline interfaces[64,65,78–82] and ELNES/XANES spectra.[83,84]

This chapter reviews our data-driven approaches for the ELNES/XANES spectrum. First, we review spectrum interpretation and prediction by combining unsupervised hierarchical clustering with supervised decision tree classification.[83] In this method, a “two-tree” comprising a dendrogram obtained by hierarchical clustering and another tree for decision tree classification is used to interpret and predict ELNES/XANES spectra. In addition, we use ML to directly quantify the structures and properties of materials from the spectra.[84] A feedforward neural network is used to

predict the local structures, chemical bonding, and transition energy from only the spectral features. Furthermore, the effect of noise on the prediction accuracy is discussed. Finally, data augmentation is used to construct a prediction model that is robust against noise.

17.2 Methodology

17.2.1 Spectrum Database Construction

This section mainly reviews two tasks: spectral interpretation/prediction and property prediction. To avoid accidental errors and noise, we first constructed the spectrum database *via* simulation (the experimental spectrum and the effect of noise are discussed later). CASTEP[85,86] code was used for the ELNES/XANES calculation, which is based on a first-principles plane-wave-basis pseudopotential method. The oxygen K-edge of oxide materials was simulated. Generalized gradient approximation (GGA) of the Perdew–Burke–Ernzerhof (PBE) functional[87] was selected to approximate the exchange–correlation functional, and the cutoff energy was set to 500 eV. To introduce core-hole effects, an excited pseudopotential was generated and applied to the excited oxygen atom in the supercell. To minimize interactions among excited atoms under periodic boundary conditions, sufficiently large supercells (>10 Å) were used in all the cases. The calculated spectrum was obtained by calculating the transition probability based on the Fermi's golden rule. The theoretical transition energy was also simulated similarly to a previous study.[86] The details of the calculated spectra are described below.

17.2.2 Spectral Data Clustering

Hierarchical clustering[88] was used to categorize the spectral data. Initially, each spectrum was assigned to its own cluster. Then, the two most similar clusters were combined into a single cluster. The clustering process was repeated until a single cluster was obtained. The similarity was estimated based on the cosine distance between a pair of spectra, which is often used for measuring spectral similarity.[89,90]

The clustering linkage scheme for measuring the cluster similarity was the “completed” method for the “two-tree” method. In this study, as we directly compared the spectral features, the onsets of all the spectra were aligned to each other. The double differential of the spectrum was used to estimate the position of the onset.

17.2.3 Decision Tree for Material Information

A decision tree visualizes classification or regression results using a tree structure. By repeatedly dividing the data into two or more subsets, the decision tree is composed of certain subsets with labels, which are the same as the labels in the training data. As this approach is a type of supervised learning, training labels are necessary. The labels obtained by spectral clustering were used as the training data. The classification and regression tree (CART) algorithm[91,92] was used for training.

17.2.4 Feedforward Neural Network

Property prediction from the spectral features was performed using a feedforward neural network. A feedforward neural network is the simplest type of neural network. It was selected for this study because it can accommodate high-dimensional input data, such as spectra, and incorporate interactions between them in a non-linear manner. We used backpropagation, based on the Adam scheme,[93] to optimize all the learning parameters in the network, thereby minimizing the mean absolute errors between the model outputs and the training targets. A rectified linear unit (ReLU) was used as the activation function, and the dropout rate was fixed at 0.5 in the hidden layers not linked with the output layer. The hyperparameters, *i.e.*, the number of hidden layers and the regularization parameters, were tuned *via* five-fold cross-validation using the validation datasets.

17.3 Results and Discussion

17.3.1 Overview of the Data-driven Prediction and Interpretation Method

The proposed data-driven spectral analysis is schematically illustrated in Figure 17.1(a)–(c). First, a spectral database, in which each spectrum has a one-to-one correspondence with its material information, was constructed as shown in Figure 17.1(a).

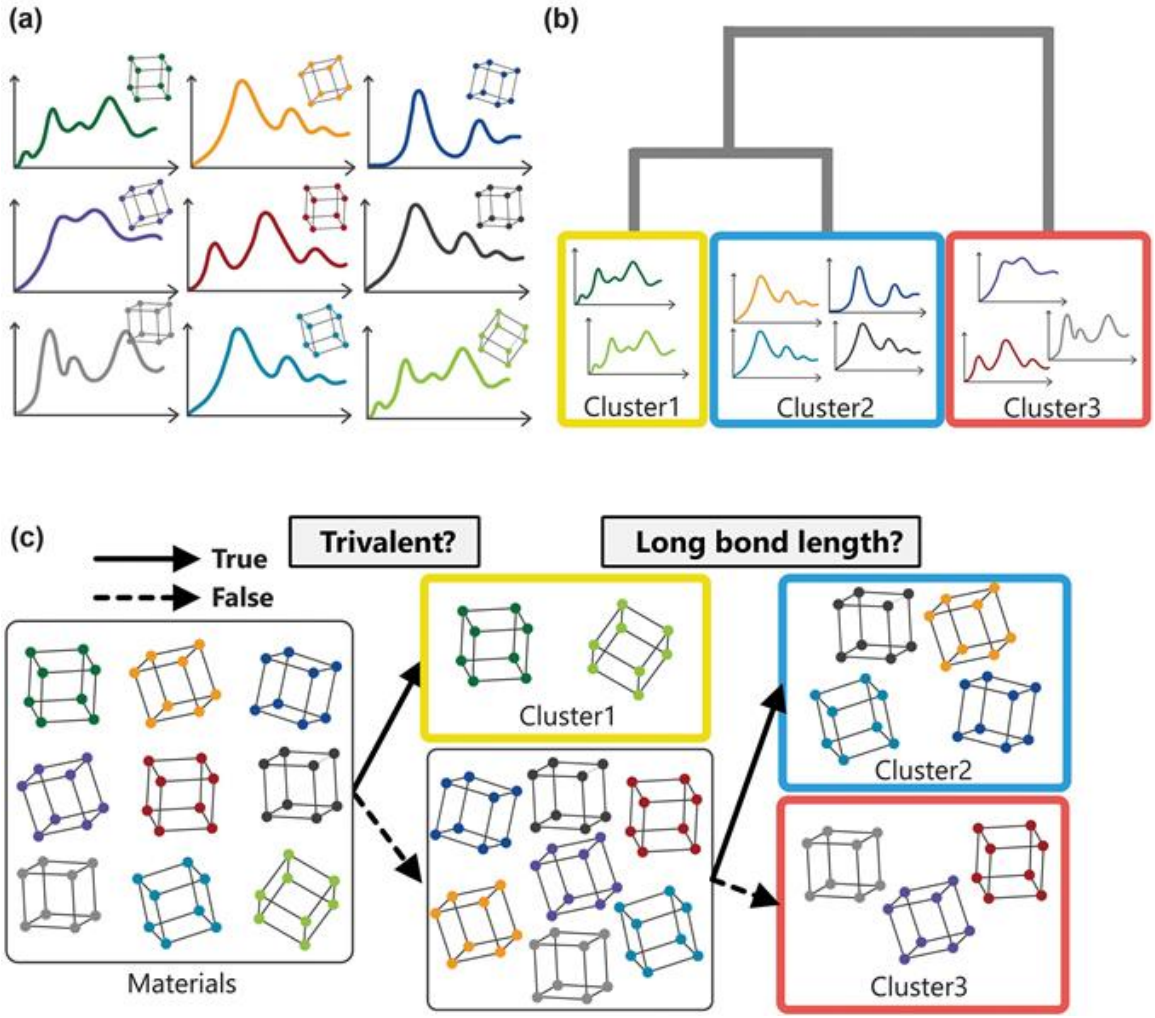


Figure 17.1 Strategy for data-driven prediction and interpretation of spectra. Schematics of (a) the spectral database and structures, which have one-to-one correspondence, (b) spectral clustering, and (c) structural classification with the spectral groups as the training data. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

As described above, we constructed the spectrum database *via* theoretical simulation. After creating the database, the included spectra are divided into groups according to their “similarity” using hierarchical cluster analysis, which divides spectral data into tree-shaped clusters called “dendrogram”. A dendrogram is schematically shown in Figure 17.1(b); the vertical axis corresponds to the “dissimilarity” of the spectra.

In the final step, the materials are classified into some groups by a decision tree based on the material information, such as the bond length, coordination number, the group in the periodic table, and valency. In the example shown in Figure 17.1(c), the objective materials are classified using the material information such as “trivalent”: true/false and “long bond length”: true/false. The most important aspect of our method is the classification of the materials into spectral groups, which are used as the training labels for classification. As the decision tree for materials is based on the spectral groups, it establishes a correlation between the material information and the spectral features. In the decision tree for material information (see Figure 17.1(c)), the branch points provide the “characteristic” descriptors for classifying the spectra, as discussed later.

Our method uses two trees: the dendrogram (see Figure 17.1(b)) and the decision tree (see Figure 17.1(c)). The two trees, related to spectrum classification and material information, are correlated with each other because the decision tree is constructed based on spectrum classification. After the two-tree is constructed, the ELNES/XANES spectra can be predicted and interpreted. The strategy for the prediction and interpretation of the spectra is schematically shown in Figure 17.2(a) and (b), respectively.

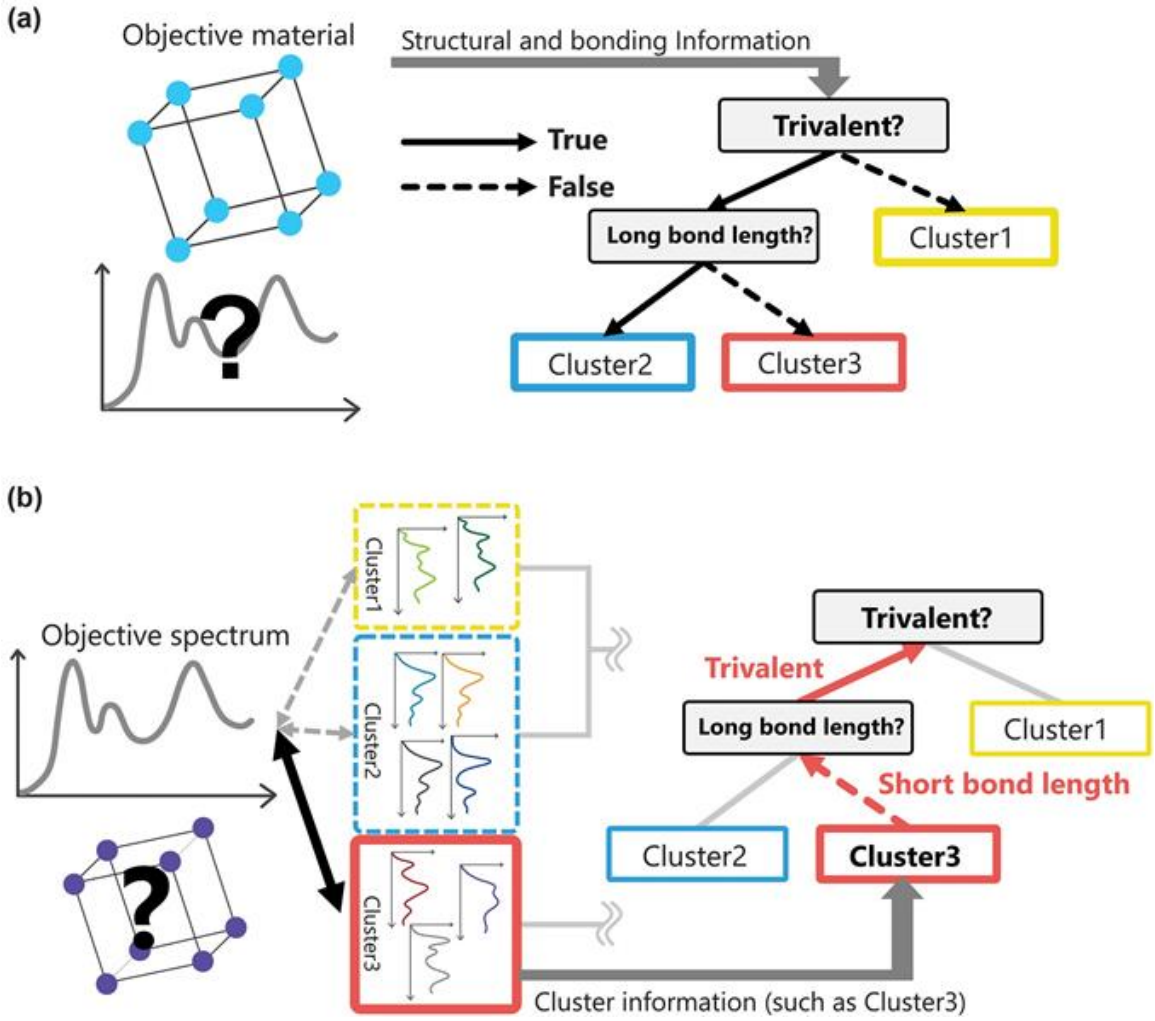


Figure 17.2 Conceptual schemes of (a) "downstream" spectral prediction and (b) "upstream" spectral interpretation. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

Consider a situation in which the atomic configuration of an objective material is known, but its ELNES/XANES spectral profiles are unknown. Such situations often occur because we generally predict the spectral profile prior to making observations to validate the experiment and sample conditions. For the prediction, we apply geometric and element information, such as the bond length, angles, and valence state, to the material information decision tree (right side of Figure 17.2(a)).

Hence, the decision tree becomes a true/false diagram (flowchart), *e.g.*, the true/false diagrams for trivalence and the bond length, as shown in

Figure 17.2(a). Moving down the true/false diagram using the material (geometric and elemental) information, we arrive at any one of the labels, clusters 1, 2, or 3, as shown in Figure 17.2(a). As the decision tree for the material information is correlated to that for the spectrum, the objective spectrum should be similar to that in the cluster. This "downstream" method corresponds to the "prediction" of the unknown spectrum of a known structure.

Next, we describe the interpretation using the two-tree. First, we define the "interpretation" of ELNES/XANES. ELNES/XANES spectra reflect the atomic and electronic structures; thus, the "interpretation of the ELNES/XANES spectrum" corresponds to the determination of the relationship between the spectrum and the atomic and electronic structures.

Consider a situation in which a spectrum of an unknown area/material is observed. In this case, we need to interpret the spectrum, *i.e.*, the relationship between the spectral profile and the material information, such as the bond length and coordination number, needs to be determined. For such interpretation, we again use the two-tree; however, we move up the decision tree as shown in Figure 17.2(b).

The strategy for the interpretation is schematically shown in Figure 17.2(b). For the ELNES/XANES spectrum of an unknown area/material, the cluster in the spectrum tree that is the most similar to the observed spectrum is first determined. By measuring the dissimilarity between the objective spectrum and all the other spectra in the database, we select the most resembling spectrum to be the most similar to the objective spectrum. In the example shown in Figure 17.2(b), assume that the observed spectrum is the most similar to a spectrum in Cluster 3 (red box). Then, by moving up the material information tree starting from Cluster 3, geometric and elemental information, such as the bond length and valency, can be obtained from the branch of the decision tree. Then, the spectrum is interpreted using such atomic and electronic structure information. In the case of Figure 17.2(b), we can determine that the observed spectrum is obtained from a material that has a "short bond length" and is "trivalent".

The branching points in a decision tree play a critical role in the interpretation, *i.e.*, when the most similar spectral cluster has been determined, the objective material can be expected to have a structure similar to that of the materials in the cluster. However, the "characteristic"

information of the corresponding cluster cannot be obtained. The branching points in the decision tree provide the "characteristic" information of the materials in the corresponding cluster.

In the next section, we describe the use of this two-tree method for the prediction and interpretation of oxygen K-edge (O-K edge) ELNES/XANES. The O-K edges are selected because they can be easily calculated using the one-particle method based on density functional theory (DFT) under the GGA framework, and they basically provide all the information for the conduction band.[46,47,50]

17.3.2 Demonstration Using the O-K Edges of Mono-metal Oxides

We calculated 39 O-K edges of oxide materials, including 14 of mono-metal oxides (listed in Table 17.1) and 25 of polymorphous SiO_2 (listed in Table 17.2). The 14 mono-metal oxides were selected because they do not have "complex" electronic structures, such as partially occupied 3d-orbitals, or magnetic properties. SiO_2 was selected because it has several polymorphs and wide applications. The 14 O-K edge spectra of the mono-metal oxides were used to demonstrate our method. The actual interpretation and prediction were performed using the 25 O-K edge spectra of polymorphous SiO_2.

Table 17.1 List of oxides and their crystal structures. TiO_2 has two types of polymorphs.

Metal oxide	Crystal structure
Li_2O	Antifluorite
BeO	Rock salt
Na_2O	Antifluorite
MgO	Rock salt
Al_2O_3	Corundum
SiO_2	β-cristobalite
CaO	Rock salt
TiO_2	Rutile, anatase

Ga_2O_3	β-form
Y_2O_3	Bixbite
ZrO_2	Fluorite
In_2O_3	Corundum
SnO_2	Rutile

Table 17.2 List of SiO_2 polymorphs, their space groups, and names. The second number in the label represents the non-equivalent oxygen sites in the unit cells of the respective polymorphs.

Label	Space group	Name
Polymorph 1	$F4_1/d\bar{3}2/m$	β-Cristobalite
Polymorph 2-1, 2	$P6_3/m2/m2/c$	Tridymite
Polymorph 3-1, 2, 3, 4	P6/m2/c2/c	Zeolite
Polymorph 4	$P4_2/n\bar{3}2/m$	—
Polymorph 5-1, 2	R32	—
Polymorph 6-1, 2	F2/d2/d2/d	—
Polymorph 7-1, 2, 3	C12/c1	—
Polymorph 8-1, 2, 3, 4, 5	C12/c1	Coesite
Polymorph 9	$P3_121$	α-Quartz
Polymorph 10	$P3_221$	—
Polymorph 11	$I\bar{4}2d$	—
Polymorph 12	$P4_12_12$	α-Cristobalite
Polymorph 13	$P6_222$	β-Quartz

First, we applied our approach to the O-K edges of the 14 mono-metal oxides. All the calculated spectra are shown in Figure 17.3. The upper part of Figure 17.3 shows the calculated O-K edges of Al_2O_3, BeO, CaO, Ga_2O_3, In_2O_3, Li_2O, MgO, Na_2O, SiO_2, SnO_2, TiO_2 (anatase), TiO_2 (rutile), Y_2O_3, and ZrO_2. The cluster analysis result for the spectra forms a dendrogram, as shown in Figure 17.4. Starting with no links at the bottom of the dendrogram, similar spectra and/or clusters gradually merge, moving upward in the dendrogram. By cutting the dendrogram horizontally at a

certain level, we can obtain a set of clusters. The selection of the cutting level is arbitrary; we selected the level indicated by the dashed line in the dendrogram, resulting in four clusters, Cluster 1 to Cluster 4 (see Figure 17.4). These four clusters have certain characteristic features; for example, Cluster 3 was composed of alkali metal oxides. We attempted to interpret these features using the decision tree.

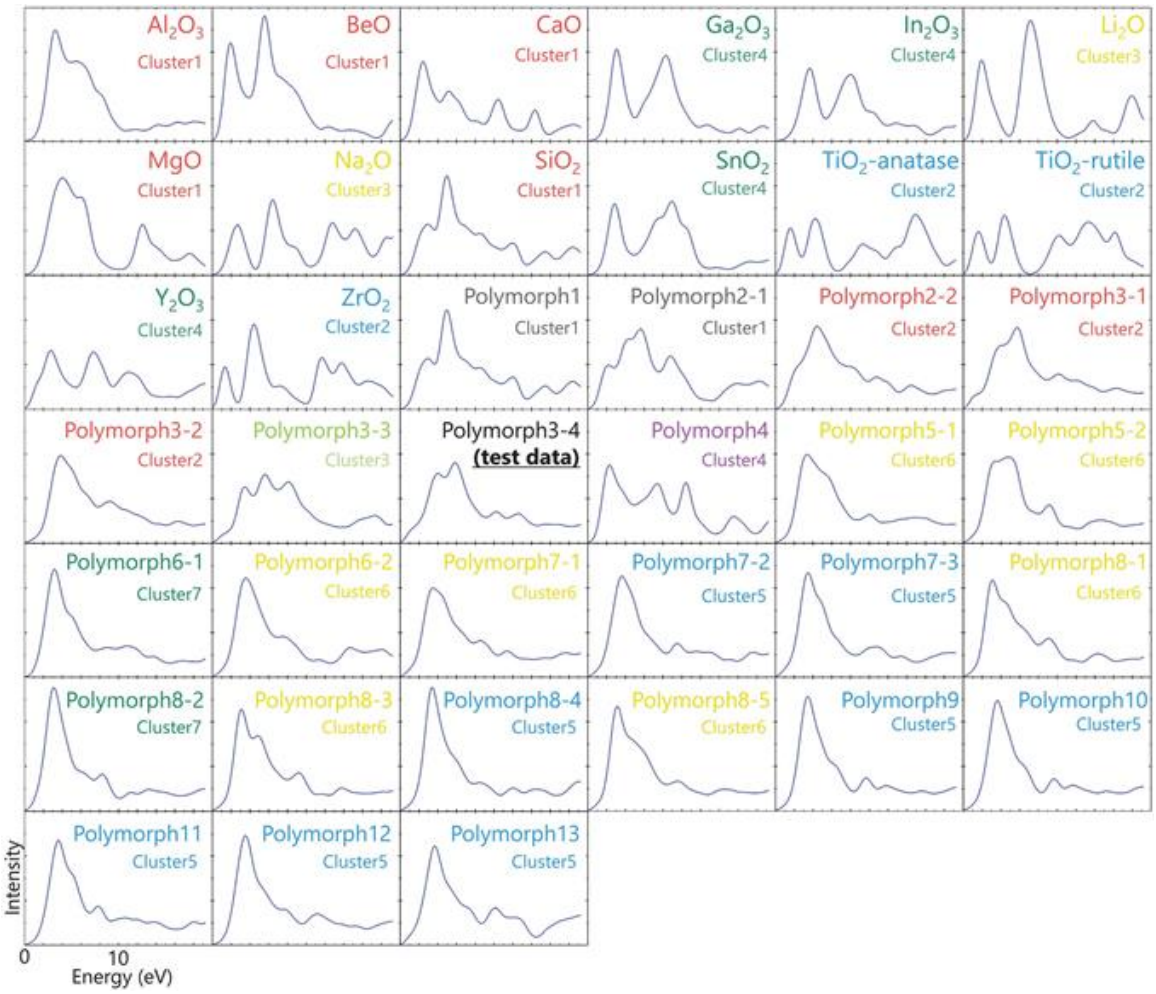


Figure 17.3 Calculated spectra in this study. O-K edge of 14 oxide materials and 25 SiO_2 polymorphs. The label colors correspond to those in the respective dendrograms and decision trees in Figures 17.4 and 17.5. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

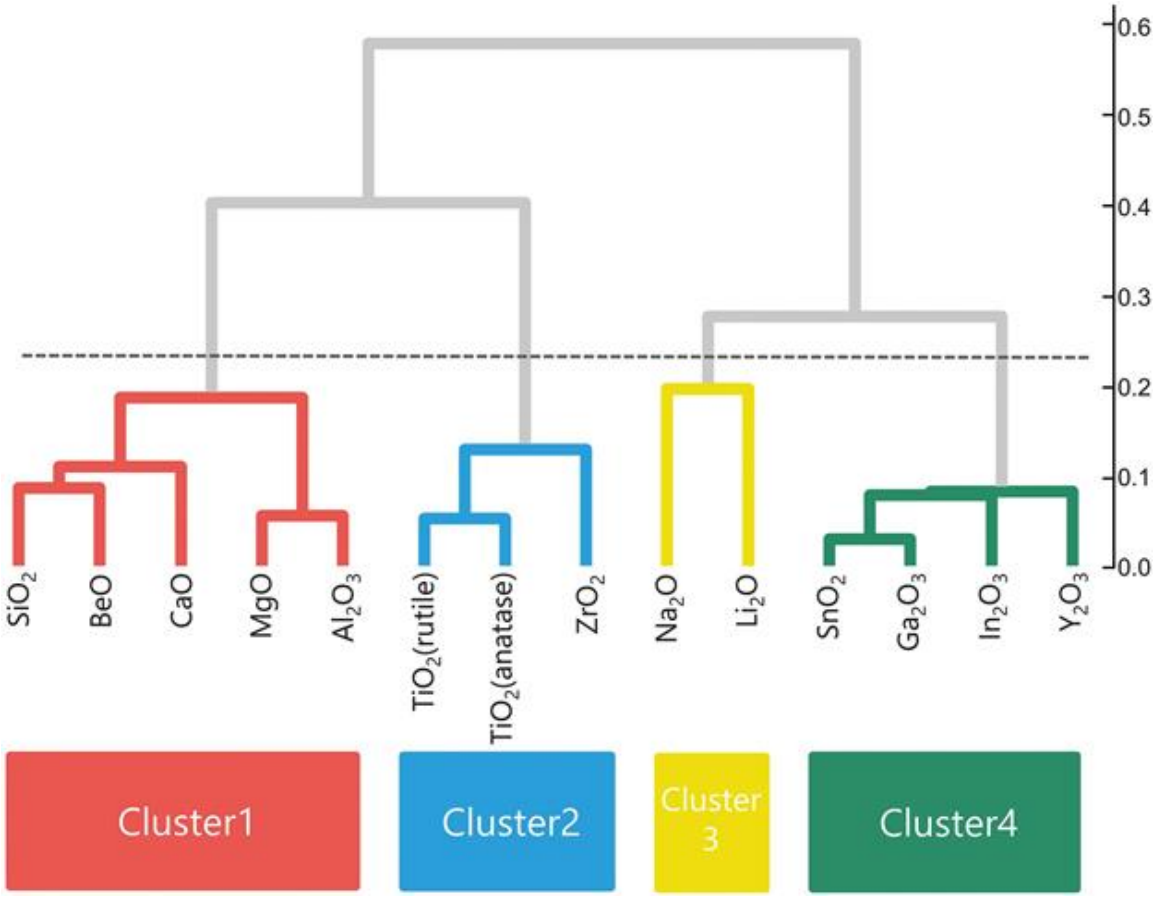


Figure 17.4 Dendrogram based on the spectral similarities of the 14 metal oxide O-K edges. The axis on the right represents the cosine distances. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

Eight parameters were selected as descriptors for the decision tree: (1) valence of the cation, (2) group and (3) period in the periodic table, coordination number of the (4) anion and (5) cation, and the number of (6) s, (7) p, and (8) d electrons of the cation in the valence and semi-core states. Figure 17.5 shows the constructed decision tree based on supervised learning; the decision tree divided the 14 oxides into two subsets according to whether they initially had d electrons and whether the "1st element or not" divided a subset and the "4th element or not" divided the other. The two constructed trees, shown in Figures 17.4 and 17.5, were used for interpretation.

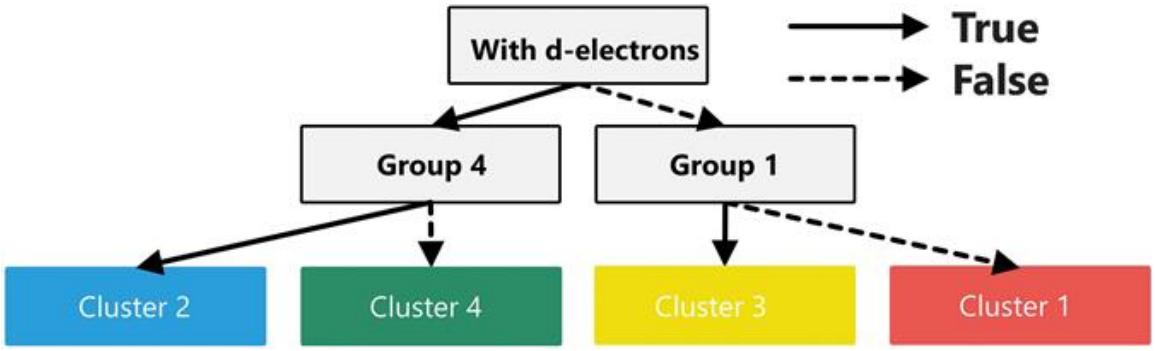


Figure 17.5 Decision tree based on the training labels in Figure 17.4. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

For interpreting the spectra, we moved up the decision tree, as mentioned above. This provides information on each cluster; for example, Cluster 3 was "with 1st elements" and "without d electrons", which were previously considered from the dendrogram.

For the demonstration, we used the elemental information of the 14 oxides. However, the relationship between the spectral features and certain geometric information, such as the bond length and coordination number, is often important for the interpretation of ELNES/XANES spectra.

To perform the actual interpretation and prediction of the ELNES spectra, we applied this method to the O-K edge spectra of SiO_2, as described in the next section.

17.3.3 Interpretation and Prediction of the O-K Edges of Polymorphous SiO_2

Next, we considered 13 polymorphs of SiO_2, the space groups of which are listed in Table 17.2. Some polymorphs have several oxygen sites, namely 2-1, 2; 3-1–4; 5-1, 2; 6-1, 2; 7-1–3; 8-1–5. Hence, all were calculated separately. Thus, a total of 25 O-K edge spectra were obtained, as shown in Figure 17.3. Because the composition is fixed to SiO_2, the atomic configuration is different and the spectral features can be interpreted using geometric information such as the Si–O bond length and the Si–O–Si bond angle. Excluding randomly selected test data, for polymorph 3-4 (middle

panel of Figure 17.3), 24 spectra were used for hierarchical clustering and creating the decision tree.

The spectral features of these O-K edges are shown in Figure 17.3, which clearly exhibit a variety of spectral profiles depending on their local structures. The cluster dendrogram for these 24 spectra is shown in Figure 17.6.

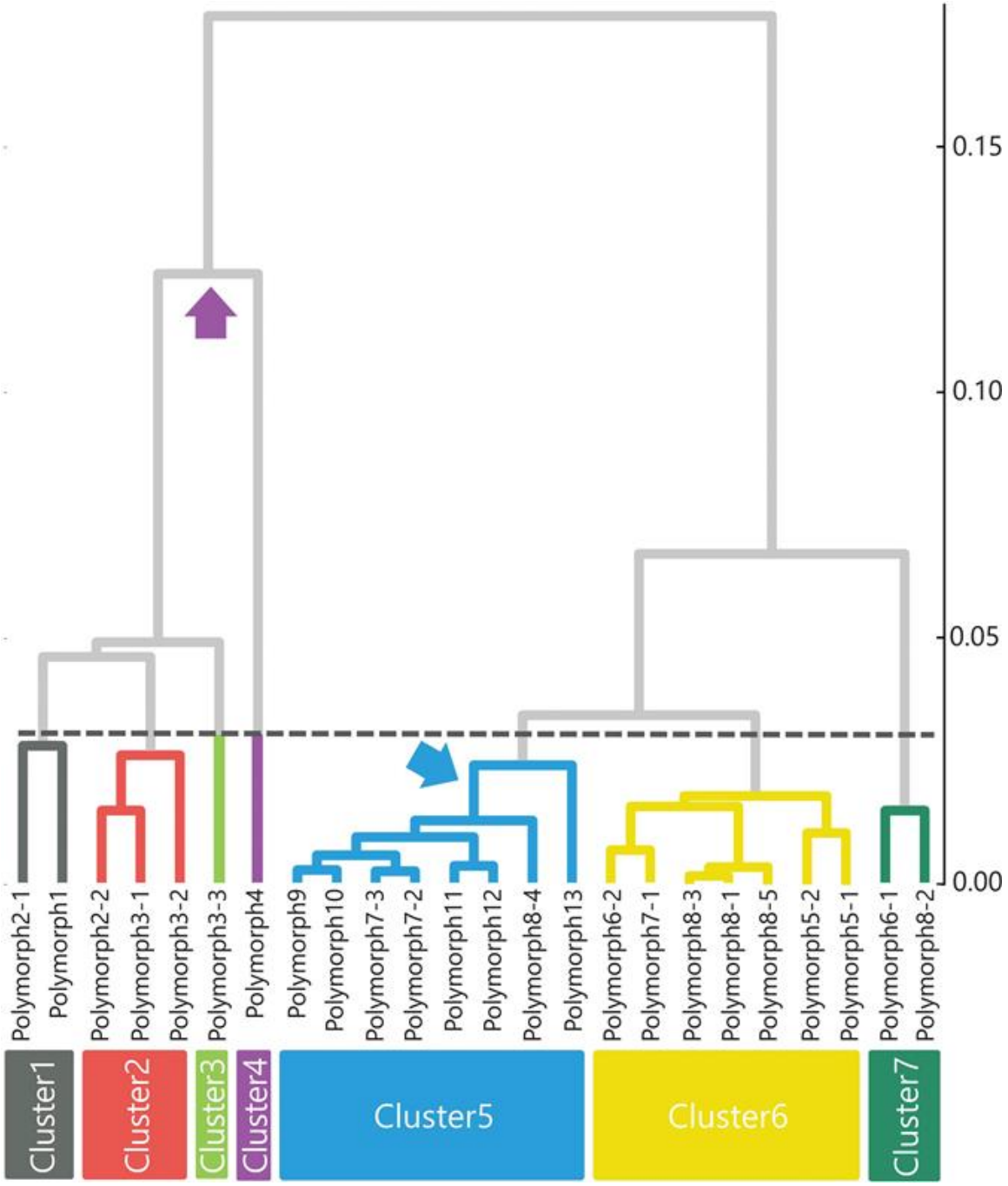


Figure 17.6 Dendrogram based on the spectral similarities of the 24 O-K edges of the SiO_2 polymorphs. The axis on the right represents the cosine distances. The top (purple in online) and lower (blue in online) arrows are explained in the main text. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

The threshold level in Figure 17.6 was set not arbitrarily but in a data-driven manner as described below, and seven clusters were generated. Cluster 4 (purple group), *i.e.*, polymorph 4, is separated from the other spectra at a higher level (as indicated by the upper (purple in online) arrow), indicating that its spectrum is highly dissimilar to those of the other polymorphs. In fact, the spectral features of polymorph 4 differ considerably from those of the others (see Figure 17.3). This result is supported by the fact that the crystal structure of polymorph 4 has higher symmetry than those of the other polymorphs. Polymorphs 7-2, 7-3, 8-4, 9, 10, 11, 12, and 13 formed the large Cluster 5 (blue in online), as their spectral features are very similar to each other (see Figure 17.3).

Next, we focus on Cluster 5 (blue in online and indicated by the lower (blue in online) arrow in Figure 17.6). The spectra in this group commonly include a large peak at the spectrum threshold, followed by small peaks, as shown in Figure 17.3 (spectra with blue labels). Their features are visually very similar to those of the spectra in Cluster 6 (yellow in online) and Cluster 7 (green in online). However, our approach indicated that these three groups have some imperceptible differences. A detailed comparison revealed a small distinction, wherein the main peaks of the blue group were slightly sharper than those of the other two groups. The sharper profile of Cluster 5 is ascribed to the more symmetric structures of the included materials. Actually, most of the materials in Cluster 5 have a single oxygen site, whereas those in Clusters 6 and 7 have multiple oxygen sites in their unit cells. Moreover, the spectra in Cluster 7 (green cluster in online) have a sharp first peak, as with the spectra in Cluster 5, but the position of the first peak is slightly different (slightly toward the lower energy side) from that in Cluster 5. These small spectral differences may be imperceptible to the human eye; however, the proposed clustering method has the potential to discern such small differences.

Next, we constructed the decision tree based on the structural information of these polymorphs. To this end, we selected 32 geometric features as descriptors. These 32 descriptors are listed in Table 17.3.

Table 17.3 List of descriptors for the SiO_2 polymorphs. O* indicates that the oxygen has a core hole.

32 Descriptors for SiO_2 polymorphous
Number of bonds (1.0–1.5, 2.0–2.5, 2.5–3.0, 3.0–3.5, 3.5–4.0, 4.0–4.5, 4.5–5.0, 5.0–5.5, 5.5–6.0 Å)
Shorter Si–O* bond length
Average Si–O* bond length
Si-O*–Si angle
Longer Si–O bond length
Shorter Si–O bond length
Average Si–O bond length
Volume of larger/smaller tetrahedron
Average O–O–O angle in larger/smaller tetrahedron
Average O–Si–O angle in larger/smaller tetrahedron

As mentioned previously, the decision tree was constructed *via* supervised learning. Figure 17.7 shows the constructed decision tree based on the threshold level indicated by the dashed line in Figure 17.6. To determine the threshold, we evaluated the accuracy rate for the training data at each threshold. Decision trees with thresholds higher than the current line had a prediction accuracy of 100% for the training dataset.

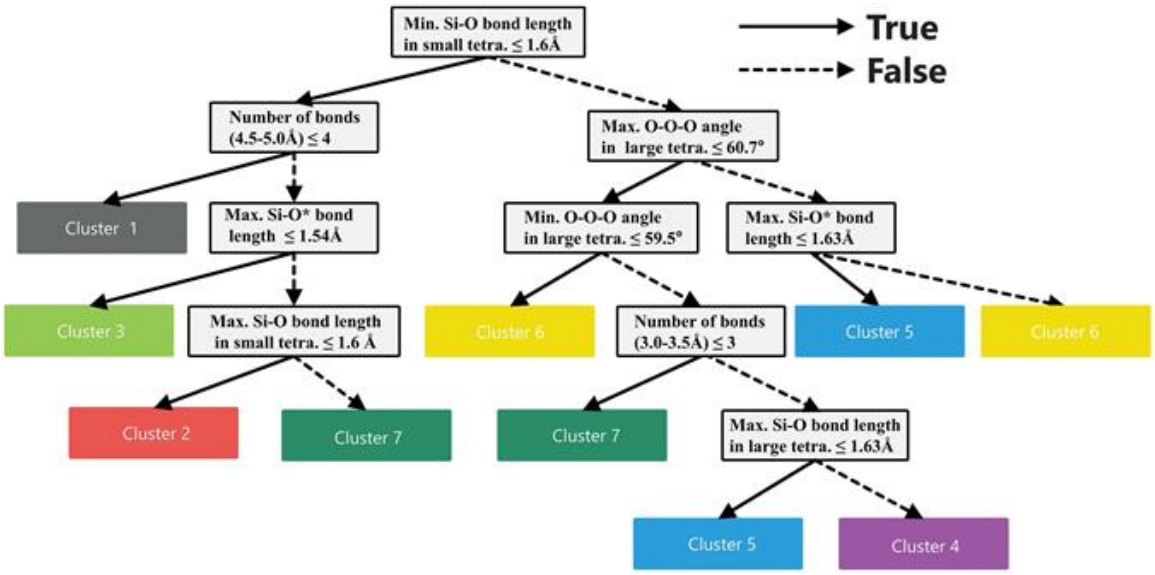


Figure 17.7 Decision tree based on the training labels in Figure 17.6. Each gray rectangle at the branching point represents a division rule. The solid and dashed lines indicate "true"

and "false," respectively. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

To evaluate the constructed decision tree, the test data, namely polymorph 3-4, was used for prediction and interpretation. First, we attempted to predict the spectrum of the test data (polymorph 3-4). In this case, the geometric information of polymorph 3-4 is known, as summarized in Table 17.4. As mentioned above, we move down the decision tree (see Figure 17.7) for prediction, using the geometric information of the polymorph 3-4 site; the decision tree starts from "minimal Si–O bond length in small tetrahedron≤1.6 Å". The site for polymorph 3-4 has a minimal Si–O bond length in the small tetrahedron of 1.53 Å; thus, it is "true". The next true/false decision is the "number of bonds between 4.5–5.0 Å≤4". For polymorph 3-4, this value is 15; thus, it is "false". Furthermore, as per the information of polymorph 3-4, presented in Table 17.4, Cluster 2 (red group) is reached.

Table 17.4 List of geometric characteristics of polymorphs 3–4.

Descriptor	Polymorph 3-4
Minimal Si–O bond length in small tetra.	1.53 Å
Number of bonds (4.5–5.0 Å)	15
Maximum Si–O* bond length	1.60 Å
Maximum Si–O bond length in small tetra.	1.59 Å

Based on the geometric information of the site, the proposed method suggests that the spectral features of polymorph 3-4 are similar to those of Cluster 2, which consists of polymorph 2-2, 3-1, and 3-2. The actual spectrum for polymorph 3-4 is shown in Figure 17.8(a) along with the spectrum for polymorph 3-1, which is the most similar to polymorph3-4 in terms of the cosine distance (see Figure 17.8(b)). Our method predicted that the test data belonged to Cluster 2, and the spectral features of polymorph 3-4 are indeed very similar to those of polymorph 3-1, indicating that this method can predict the spectral features correctly.

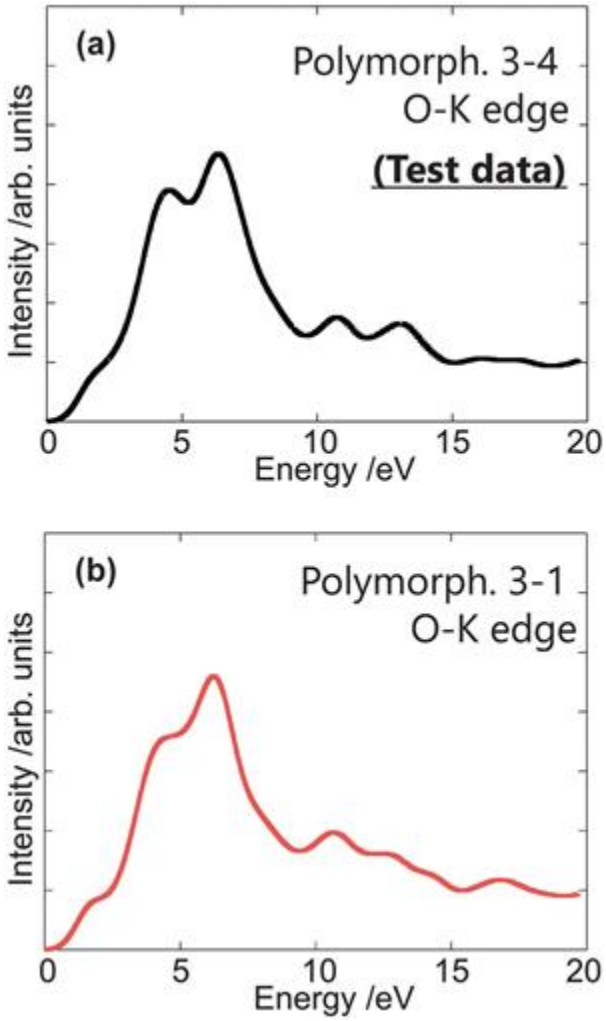


Figure 17.8 Calculated O-K edge spectra of (a) polymorph 3-4 and (b) polymorph 3-1. Adapted from ref. 83, https://doi.org/10.1038/s41598-018-30994-6, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

Next, our method is used for interpreting the spectrum. We considered a different situation in which we know the spectral features but not the material information of polymorph 3-4. First, we measured the cosine distance between the objective spectrum (in this case, the spectrum of polymorph 3-4) and all the other spectra in the database. In this case, the cosine distance to polymorph 3-1 is the shortest, indicating that this spectrum is the most similar to that of polymorph 3-4. Through this process, the class of the objective spectrum can be determined. In this test case, the

objective spectrum is categorized as Cluster 2 because polymorph 3-1 belongs to Cluster 2.

The spectrum is interpreted using the decision tree in Figure 17.7. As the objective spectrum is the most similar to that of polymorph 3-1, the material information of the objective material is expected to be similar to that of polymorph 3-1 and the other polymorphs in Cluster 2. However, as mentioned above, the "characteristic" features of the materials in Cluster 2 cannot be determined. To obtain the "characteristic" feature, we need to travel up the decision tree in Figure 17.7.

We can obtain the initial information, "maximum Si–O bond length in the small tetrahedron ≤1.6 Å", and the next branch points, "max. Si–O* bond length ≥1.54 Å" and "number of bonds between 4.5 and 5.0 Å≥4", from the decision tree (see Figure 17.7). This information is in agreement with the geometric information of polymorph 3-4 (as summarized in Table 17.4). Thus, we can obtain the "characteristic" features of the materials in Cluster 2 from the branching points of the decision tree, and the spectrum can be interpreted using these "characteristic" features.

In this section, we proposed a data-driven approach for predicting and interpreting ELNES/XANES spectra. Our method is based on hierarchical clustering and a decision tree. The calculated O-K edges of 14 metal oxides and 25 SiO_2 polymorphs were used to demonstrate the proposed method. Our method was effective in predicting the spectral features from the material information.

17.3.4 Machine Learning to Quantify Structures and Properties from ELNES/XANES

Although ELNES/XANES is effective for investigating local atomic and electronic structures, "direct" determination of the material properties using core-loss spectroscopy is more attractive for atomic-resolution function measurements. However, the relationship between the core-loss features and the material functions is ambiguous; hence, extracting it requires theoretical expertise and calculations. Therefore, only a few studies have realized quantification of properties using core-loss spectroscopy.[21,50,94,95] Determining the direct relationship between the core-loss features and the

material properties could facilitate a nano-scale understanding of the material properties.

Here, we employed a feedforward neural network to quantify the structures and properties from the ELNES feature.[84] The schematic diagram of the constructed neural network for predicting properties from the spectrum is shown in Figure 17.9.

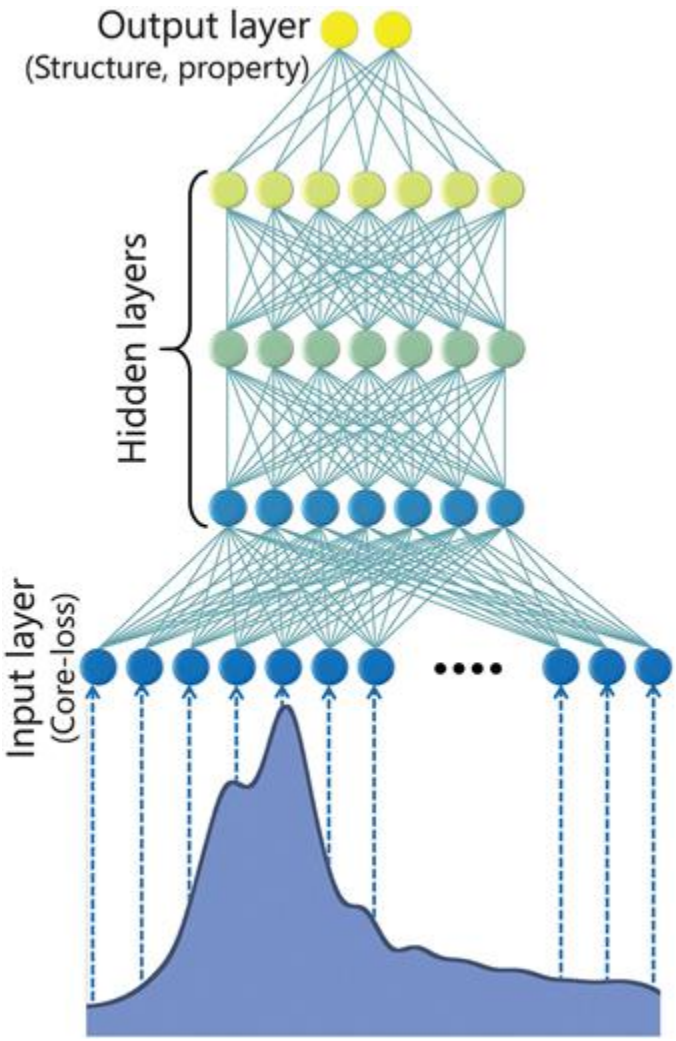


Figure 17.9 Schematic of a feedforward neural network model. The input layer (located in the lower position in this figure) accepts intensities of ELNES/XANES spectra at each energy and transfers that information toward the output layer (located in the highest position in this figure). Adapted from ref. 84, https://doi.org/10.1088/2515-7639/ab0b68, under the terms of the CC BY 3.0 licence, https://creativecommons.org/licenses/by/3.0/.

The input data consisted of the intensities of the core-loss spectrum, which were in the range of −1 to 15 eV in increments of 0.1 eV, *i.e.* 160 dimensional inputs, and the output data were the objective properties. Thus, the theoretical transition energy was not considered in the input data. The properties of individual atomic sites, including local geometrical and bonding properties, such as bond length, bond angles, Voronoi volume, bond overlap population, and Mulliken charge, were calculated by the CASTEP code.[85,86] In addition, the transition energy of the respective spectra was selected as the output. Thus, prediction of the transition energy was attempted using only the spectral feature. The average bond length, average bond angle, and Voronoi volume are geometrical features; the first two provide short-range information and the third provides short- and middle-range information. The bonding properties, bond overlap population, Mulliken charge, and excitation energy are related to the valence and core states. Although prediction of the excitation energy is not necessary because it is observed with the experimental spectrum, prediction of the excitation energy based solely on the spectral features was attempted to demonstrate the ability of the present method to predict a core-electron-related property.

Considering these six properties as the objective variables, regression analysis was conducted *via* a neural network. The best hyperparameters were determined by grid search. The correct and predicted values of the training and test data are plotted in Figure 17.10(a)–(f).

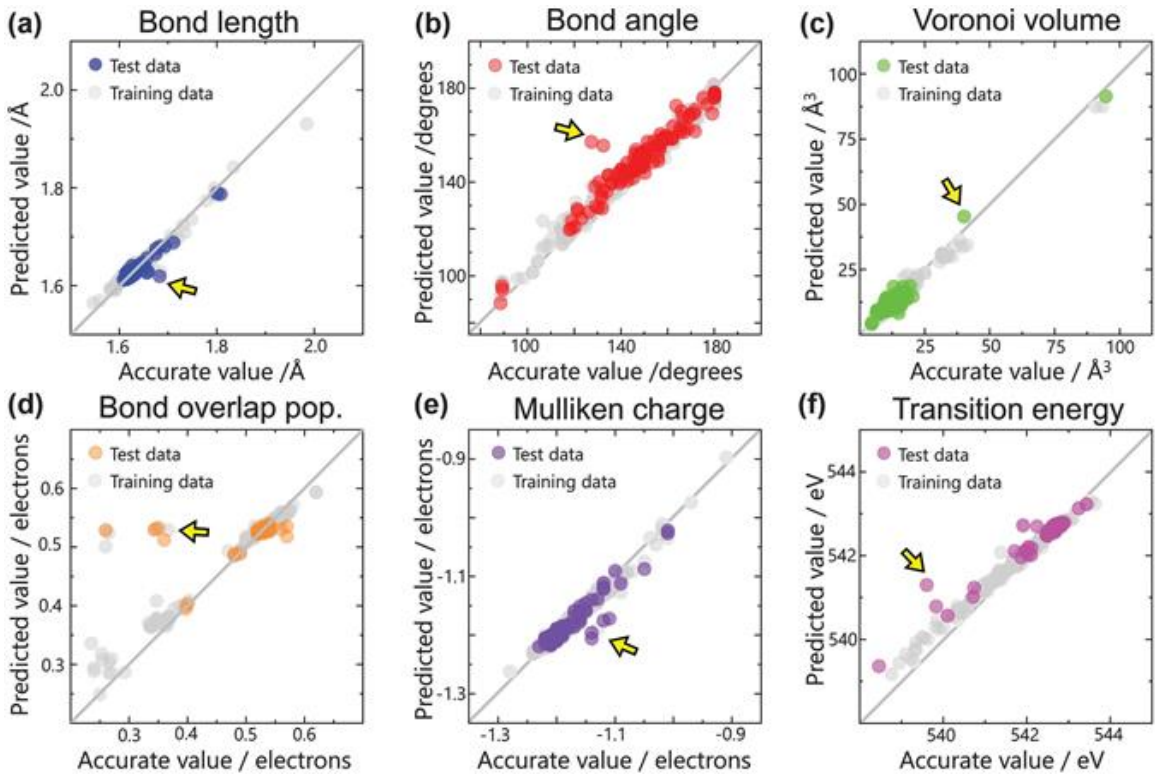


Figure 17.10 Accurate and predicted values of six types of properties: (a) average bond length, (b) average bond angle, (c) Voronoi volume, (d) bond overlap population, (e) Mulliken charge, and (f) transition energy. The gray and dark gray (colored in online) dots represent training and testing data, respectively. The gray diagonal lines indicate that the predicted values are equal to the actual ones. The arrows indicate the spectra with large errors. Adapted from ref. 84, https://doi.org/10.1088/2515-7639/ab0b68, under the terms of the CC BY 3.0 licence, https://creativecommons.org/licenses/by/3.0/.

The gray and colored circles represent the training and testing data, respectively. The circles on the diagonal gray line imply that the predicted values are equal to the actual values. As most of the colored circles are on the gray line in Figure 17.10(a)–(f), our neural network model can predict these local properties accurately, indicating that the information about both geometrical and bonding properties is implicitly contained in the spectral features of the core-loss spectrum. Moreover, the prediction model works well not only for bond lengths and bond angles but also for the Voronoi volume. Core-loss is often believed to reflect very localized information (such as bond lengths and angles); however, the present result implies that middle-range information (such as Voronoi volume) is also included implicitly, and it can be extracted by machine learning.

In addition to the geometrical properties, the same argument can apply to bonding properties. The neural network can correctly predict all the bonding properties (see Figure 17.10(d)–(f)). Note that the present method can correctly predict the excitation energy using "only" the spectral features.

In core-loss spectra, the spectral features and excitation energy are often discussed separately; for instance, the excitation energy, *i.e.*, the chemical shift, is related to the oxidization state, whereas the spectral profile reflects the more detailed and fine structure of the partial density of states (PDOS) of the conduction band. However, the present results indicate that the spectral features themselves contain information on the excitation energy, and the excitation energy can be predicted solely by the spectral features. Furthermore, the present method can correctly predict the valence band information in ground states, *i.e.*, the Mulliken charge and bond overlap population. The core-loss spectrum is known to reflect the PDOS of the conduction band in the excited state and it has no direct relationship with the valence states; however, the present results indicate that spectral features definitely contain information on the valence states. By using a neural network, geometrical and bonding information was extracted from the spectral features of the core-loss spectrum.

A detailed inspection of Figure 17.10 reveals some circles located away from the diagonal lines (denoted by arrows), indicating that the prediction with the neural network fails for some materials/sites. On careful examination of these circles, *i.e.* the spectra with large errors, it was found that the spectra with large errors for one property tend to have similar large errors for other properties. Specifically, 13 spectra have errors ranked in the worst ten with more than two properties, and 6 spectra have errors ranked with more than four properties. The spectra with large errors are indicated by arrows in Figure 17.10(a)–(f).

A detailed analysis showed two causes of misprediction. First, no similar spectrum was included in the training data. Second, when similar spectra were included in the training data, the corresponding properties differed from each other. Both these causes can be attributed to unique electronic structures, which cannot be learned from the training data.[84]

17.3.5 Quantification of Structures and Properties from Noisy Spectra

We showed that ELNES/XANES can be used to quantify the structures and properties based on a neural network. However, these investigations were performed using a spectral dataset constructed by theoretical simulation. Specifically, noise was not considered even though an experimental spectrum must contain noise. To apply our method to the experimental spectra, we must investigate the effect of noise on the prediction accuracy.

Figure 17.11 shows the effect of noise on the prediction accuracy. Noise was applied to the original calculated spectra, which do not include noise, based on the Poisson distribution, and the noise intensity was changed and eight types of spectra with different noise levels were generated (black lines from top to bottom in Figure 17.11(a)). Figure 17.11(a) shows the spectra without noise (blue line in online) and with noise (black lines in online). As the distribution of noise differs randomly, 10 spectra with each noise level were generated.

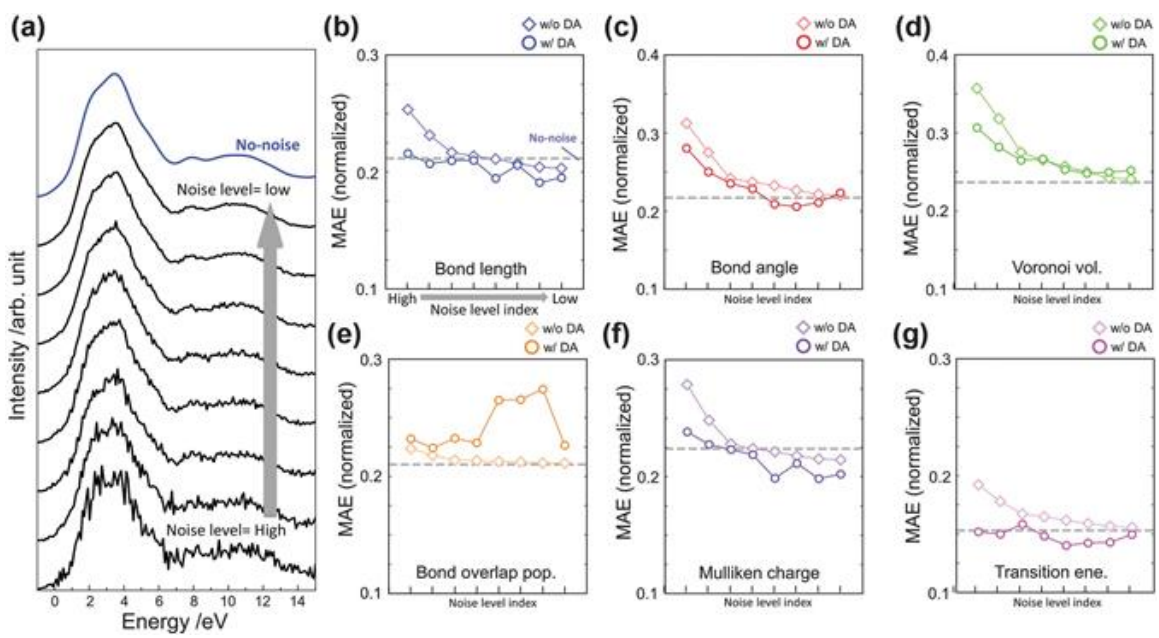


Figure 17.11 (a) The top (blue) and lower black spectra correspond to no-noise and noise-added spectra, respectively. Eight different noise levels were generated based on a Poisson distribution (bottom–top correspond to high–low noise levels). (b)–(g) Prediction accuracy measured by mean absolute error (MAE) of each property with different

noise levels. The square and circle symbols correspond to the prediction accuracies without and with data augmentation (DA).

First, the spectra with noise were employed for the prediction model constructed by the spectra without noise. The square symbols in Figure 17.11 (b)–(g) show the prediction accuracies estimated by the mean absolute error (MAE) of the respective properties. The gray dashed lines in these figures represent the prediction accuracies using the spectra without noise. The square symbols are basically located at a higher MAE than the gray dashed lines, indicating that the prediction model constructed by the spectra without noise is not robust against noise.

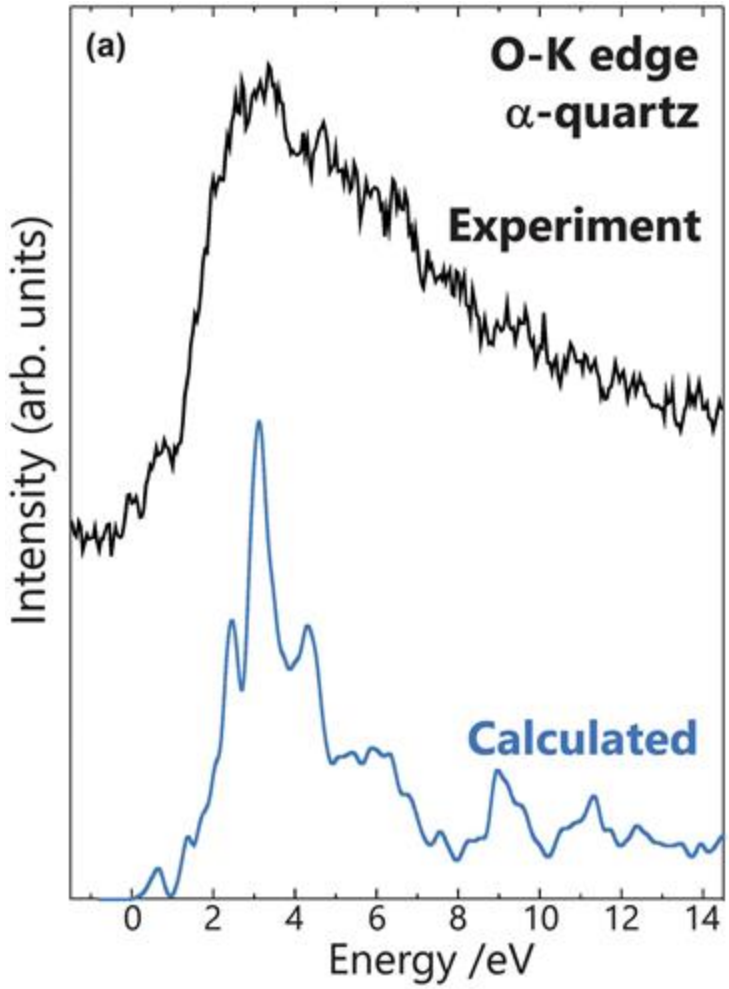
To construct a more robust prediction model against the noise, a data processing method, data augmentation (DA), was adopted. DA is used to increase the number of training data and create a robust prediction model. In the case of data augmentation of image data, noise-added and/or rotated images were included in the training data, which is effective in constructing more versatile prediction models. In the present case, DA was performed using noise-added spectra. In addition to the no-noise spectra, the noise-added spectra were used as the training data. As mentioned above, the spectra with eight noise levels were separately generated. As the noise distribution is random, 10 spectra were separately generated for the respective noise levels, and the prediction models were constructed for the respective noise levels. Thus, the number of the training data for the respective noise levels becomes 10 times greater than that of previous models without DA.

The prediction accuracy with DA is shown in Figure 17.11(b)–(g) by circles. Clearly, the MAE with DA (circle symbols) is usually lower than that without DA (square symbols) even for spectra with large noise, indicating that DA is highly effective in constructing a prediction model that is robust against noise.

In terms of the bond length, bond angle, Mulliken charge, and transition energy, the prediction accuracy becomes higher (lower MAE) than that for the no-noise spectra (gray dashed lines) (see Figure 17.11(b)(c)(f)(g)). This higher accuracy may be due to the larger number of training data than that without DA. Furthermore, the prediction accuracy for the bond overlap population was not improved by DA (see Figure 17.11(e)). Although this

worse prediction was not clear, we can expect that the information on the bond overlap population in the spectrum is small and is hidden by noise. Thus, high-quality spectra with small noise are indispensable for quantifying the bond overlap population by ELNES/XANES.

Finally, we applied our model to an experimental spectrum. We measured the O-K edge of a α-quartz sample. We used an aberration-corrected STEM (JEM-ARM200F, JEOL Ltd.) equipped with a monochromator. As the O-K edge is broadened by the electron–hole lifetime, by approximately 0.1 eV, the energy resolution of the instrument can be ignored. We retrained the model by the database broaden by Gaussian with 0.1 eV and predicted the properties of the experimental spectrum. Figure 17.12(a) and (b) show the theoretical and experimental spectra and both of the properties.



(b)

	Theoretical	Experiment
Ave. bond length(Å)	1.630	1.620
Ave. bond angle (degrees)	147	143
Voronoi volume(Å^3)	16.4	17.4
Bond overlap pop. (elec.)	0.53	0.53
Mulliken charge (elec.)	-1.18	-1.18
Excitation energy (eV)	542.9	542.4

Figure 17.12 (a) Experimental and theoretical O-K edge spectra of α-quartz. (b) Theoretical and predicted values from the experimental spectrum. Adapted from ref. 84, https://doi.org/10.1088/2515-7639/ab0b68, under the terms of the CC BY 3.0 licence, https://creativecommons.org/licenses/by/3.0/.

The experimental spectrum has a profile similar to that of the calculated spectrum; however, it includes some noise. Nevertheless, the predicted properties of the experimental spectrum are extremely similar to the accurate values. Therefore, we can conclude that the model constructed using the simulated spectra is considerably robust against noise and it can predict properties accurately even from the experimental spectrum.

17.4 Conclusions

In this study, we reviewed “data-driven” approaches for ELNES/XANES spectra. First, we applied unsupervised hierarchical clustering and a supervised decision tree to the spectra. A two-tree comprising a dendrogram and a decision tree was constructed, and it was used to interpret and predict the spectra. The calculated oxygen K-edges of 14 metal oxides and 25 SiO_2 polymorphs were used to demonstrate the proposed method. With this method, the ELNES/XANES spectra can be interpreted in accordance with material information, such as chemical, elemental, and geometric information. Furthermore, our method is effective in predicting the spectral features from the material information.

Further, hidden information was extracted from core-loss spectra *via* a feedforward neural network. Thus, three types of geometrical properties, namely average bond length, average bond angle, and Voronoi volume, were successfully predicted. This implies that the spectral features reflect not only short-range information but also relatively middle-range information. Furthermore, our method could simultaneously predict three types of bonding properties, namely bond overlap population, Mulliken charge, and transition energy. Although these chemical bonding properties mainly originate from the valence band, the present method can correctly predict this bonding information, indicating that the core-loss spectra, which reflect the conduction band, contain information pertaining to the valence bands in the ground states.

In addition, the effect of noise on the prediction accuracy was investigated. We showed that data augmentation can be used to construct a prediction model that is robust against noise. We also confirmed that our

method can correctly predict the material properties from the spectral features in most cases, and it can be applied to an experimental spectrum with some noise.

We emphasize that the proposed machine learning method is not spectroscopy-dependent and should thus work well in terms of interpreting and predicting any spectral data as well as diffraction, emission, and experimental data. A substantial database of over 500 000–800 000 calculated spectra was recently constructed.[82,96] Combining this database or others similar to this method could facilitate versatile and accurate predictions and interpretations. In summary, we believe that our method can pave the way for data-driven spectral interpretation and prediction and local measurements of material properties.

Acknowledgements

The experiments reported here were performed in collaboration with Dr Tomohiro Miyata, Tohoku University; Dr Masashi Tsubaki, AIST; and Prof. Koji Tsuda and Mr Kunyen Liao, University of Tokyo. The EELS measurements were supported by Kyoto University Nano Technology Hub under the “Nanotechnology Platform Project” sponsored by the Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan. This study was also supported by JST-PRESTO (JPM-JPR16NB 16814592), MEXT, under Grant Nos. 17H06094, 19H00818, and 19H05787 as well as the special fund of the Institute of Industrial Science, University of Tokyo (Tenkai5504850104).

References

1. K. Kimoto, T. Asaka, T. Nagai, M. Saito, Y. Matsui and K. Ishizuka, *Nature*, 2007, **450**, 702–704.
2. B. Barwick, S. P. Hyun, O. H. Kwon, J. S. Baskin and A. H. Zewail, *Science*, 2008, **322**, 1227–1231.
3. I. Tanaka, T. Mizoguchi, M. Matsui, S. Yoshioka, H. Adachi, T. Yamamoto, T. Okajima, M. Umesaki, W. Y. Ching, Y. Inoue, M.

Mizuno, H. Araki and Y. Shirai, *Nat. Mater.*, 2003, **2**, 541–545.
4. L. M. Brown, *Nature*, 1993, **366**, 721.
5. R. F. Egerton, *Electron Energy-loss Spectroscopy in the Electron Microscope*, Springer, Boston, MA, 2011.
6. J. Stöhr, *NEXAFS Spectroscopy*, Springer, Berlin, Heidelberg, **vol. 25**, 1992.
7. T. Mizoguchi, M. Sakurai, A. Nakamura, K. Matsunaga, I. Tanaka, T. Yamamoto and Y. Ikuhara, *Phys. Rev. B*, 2004, **70**, 153101.
8. K. Matsunaga, H. Murata, T. Mizoguchi and A. Nakahira, *Acta Biomater.*, 2010, **6**, 2289–2293.
9. K. Suenaga and M. Koshino, *Nature*, 2010, **468**, 1088–1090.
10. M. Varela, S. D. Findlay, A. R. Lupini, H. M. Christen, A. Y. Borisevich, N. Dellby, O. L. Krivanek, P. D. Nellist, M. P. Oxley, L. J. Allen and S. J. Pennycook, *Phys. Rev. Lett.*, 2004, **92**, 095502.
11. T. Tohei, T. Mizoguchi, H. Hiramatsu, Y. Kamihara, H. Hosono and Y. Ikuhara, *Appl. Phys. Lett.*, 2009, **95**, 193107.
12. T. Miyata, M. Fukuyama, A. Hibara, E. Okunishi, M. Mukai and T. Mizoguchi, *Microscopy*, 2014, **63**, 377–382.
13. O. L. Krivanek, T. C. Lovejoy, N. Dellby, T. Aoki, R. W. Carpenter, P. Rez, E. Soignard, J. Zhu, P. E. Batson, M. J. Lagos, R. F. Egerton and P. A. Crozier, *Nature*, 2014, **514**, 209–212.
14. J. C. Idrobo, A. R. Lupini, T. Feng, R. R. Unocic, F. S. Walden, D. S. Gardiner, T. C. Lovejoy, N. Dellby, S. T. Pantelides and O. L. Krivanek, *Phys. Rev. Lett.*, 2018, **120**, 95901.
15. J. A. Hachtel, J. Huang, I. Popovs, S. Jansone-Popova, J. K. Keum, J. Jakowski, T. C. Lovejoy, N. Dellby, O. L. Krivanek and J. C. Idrobo, *Science*, 2019, **363**, 525–528.
16. T. Mizoguchi, Y. Sato, J. P. Buban, K. Matsunaga, T. Yamamoto and Y. Ikuhara, *Appl. Phys. Lett.*, 2005, **87**, 241920.
17. M. Kunisu, I. Tanaka, T. Yamamoto, T. Suga and T. Mizoguchi, *J. Phys. Condens. Matter*, 2004, **16**, 3801–3806.
18. T. Suga, S. Kameyama, S. Yoshioka, T. Yamamoto, I. Tanaka and T. Mizoguchi, *Appl. Phys. Lett.*, 2005, **86**, 163113.
19. S.-Y. Choi, S.-D. Kim, M. Choi, H.-S. Lee, J. Ryu, N. Shibata, T. Mizoguchi, E. Tochigi, T. Yamamoto, S.-J. L. Kang and Y. Ikuhara, *Nano Lett.*, 2015, **15**, 4129–4134.

20. K. Tatsumi, T. Mizoguchi, S. Yoshioka, T. Yamamoto, T. Suga, T. Sekine and I. Tanaka, *Phys. Rev. B*, 2005, **71**, 033202.
21. T. Mizoguchi, T. Sasaki, S. Tanaka, K. Matsunaga, T. Yamamoto, M. Kohyama and Y. Ikuhara, *Phys. Rev. B*, 2006, **74**, 235408.
22. K. Nakazawa, T. Miyata, S. Amma and T. Mizoguchi, *Scr. Mater*, 2018, **154**, 197–201.
23. K. Kubobuchi, M. Mogi, H. Ikeno, I. Tanaka, H. Imai and T. Mizoguchi, *Appl. Phys. Lett.*, 2014, **104**, 053906.
24. K. Kubobuchi, M. Mogi, M. Matsumoto, T. Baba, C. Yogi, C. Sato, T. Yamamoto, T. Mizoguchi and H. Imai, *J. Appl. Phys.*, 2016, **120**, 142125.
25. T. Sasaki, T. Mizoguchi, K. Matsunaga, S. Tanaka, T. Yamamoto, M. Kohyama and Y. Ikuhara, *Appl. Surf. Sci.*, 2005, **241**, 87–90.
26. T. Mizoguchi, J. P. Buban, K. Matsunaga, T. Yamamoto and Y. Ikuhara, *Ultramicroscopy*, 2006, **106**, 92–104.
27. T. Mizoguchi, M. Varela, J. P. Buban, T. Yamamoto and Y. Ikuhara, *Phys. Rev. B*, 2008, **77**, 024504.
28. T. Miyata and T. Mizoguchi, *Ultramicroscopy*, 2017, **178**, 81–87.
29. Y. Sugimori, T. Miyata, H. Hashiguchi, E. Okunishi and T. Mizoguchi, *RSC Adv.*, 2019, **9**, 10520–10527.
30. T. Miyata, F. Uesugi and T. Mizoguchi, *Sci. Adv.*, 2017, **3**, e1701546.
31. T. Miyata and T. Mizoguchi, *Microscopy*, 2016, **65**, i33.
32. T. Mizoguchi, I. Tanaka, M. Yoshiya, F. Oba, K. Ogasawara and H. Adachi, *Phys. Rev. B*, 2000, **61**, 2180–2187.
33. I. Tanaka, H. Araki, M. Yoshiya, T. Mizoguchi, K. Ogasawara and H. Adachi, *Phys. Rev. B*, 1999, **60**, 4944–4951.
34. T. Mizoguchi, I. Tanaka, S. Yoshioka, M. Kunisu, T. Yamamoto and W. Y. Ching, *Phys. Rev. B*, 2004, **70**, 045103.
35. T. Yamamoto, T. Mizoguchi and I. Tanaka, *Phys. Rev. B*, 2005, **71**, 245113.
36. T. Mizoguchi, I. Tanaka, M. Kunisu, M. Yoshiya, H. Adachi and W. Y. Ching, *Micron*, 2003, **34**, 249–254.
37. T. Mizoguchi, M. Yoshiya, J. Li, F. Oba, I. Tanaka and H. Adachi, *Ultramicroscopy*, 2001, **86**, 363–370.
38. K. Tomita, T. Miyata, W. Olovsson and T. Mizoguchi, *J. Phys. Chem. C*, 2016, **120**, 9036–9042.

39. W. Olovsson, I. Tanaka, T. Mizoguchi, P. Puschnig and C. Ambrosch-Draxl, *Phys. Rev. B*, 2009, **79**, 041102.
40. W. Olovsson, I. Tanaka, T. Mizoguchi, G. Radtke, P. Puschnig and C. Ambrosch-Draxl, *Phys. Rev. B*, 2011, **83**, 195206.
41. K. Tomita, T. Miyata, W. Olovsson and T. Mizoguchi, *Ultramicroscopy*, 2017, **178**, 105–111.
42. H. Ikeno, T. Mizoguchi and I. Tanaka, *Phys. Rev. B*, 2011, **83**, 155107.
43. H. Ikeno, T. Mizoguchi, Y. Koyama, Y. Kumagai and I. Tanaka, *Ultramicroscopy*, 2006, **106**, 970–975.
44. S. Ootsuki, H. Ikeno, Y. Umeda, Y. Yonezawa, H. Moriwake, A. Kuwabara, O. Kido, S. Ueda, I. Tanaka, Y. Fujikawa and T. Mizoguchi, *Microscopy*, 2014, **63**, 249–254.
45. S. Ootsuki, H. Ikeno, Y. Umeda, H. Moriwake, A. Kuwabara, O. Kido, S. Ueda, I. Tanaka, Y. Fujikawa and T. Mizoguchi, *Appl. Phys. Lett.*, 2011, **99**, 233109.
46. T. Mizoguchi, W. Olovsson, H. Ikeno and I. Tanaka, *Micron*, 2010, **41**, 695–709.
47. H. Ikeno and T. Mizoguchi, *J. Electron Microsc.*, 2017, **66**, 305–327.
48. I. Tanaka, T. Mizoguchi and T. Yamamoto, *J. Am. Ceram. Soc*, 2005, **88**, 2013–2029.
49. I. Tanaka and T. Mizoguchi, *J. Phys. Condens. matter*, 2009, **21**, 104201.
50. T. Mizoguchi, *J. Phys. Condens. Matter*, 2009, **21**, 104215.
51. T. Mizoguchi, K. Tatsumi and I. Tanaka, *Ultramicroscopy*, 2006, **106**, 1120–1128.
52. S. Nishida, S. Kobayashi, A. Kumamoto, H. Ikeno, T. Mizoguchi, I. Tanaka, Y. Ikuhara and T. Yamamoto, *J. Appl. Phys.*, 2013, 054906.
53. Y. Matsui and T. Mizoguchi, *Chem. Phys. Lett.*, 2016, **649**, 92–96.
54. Y. Matsui and T. Mizoguchi, *Chem. Phys. Lett.*, 2016, **649**, 92–96.
55. H. Katsukura, T. Miyata, M. Shirai, H. Matsumoto and T. Mizoguchi, *Sci. Rep.*, 2017, **7**, 16434.
56. W. Olovsson, T. Mizoguchi, M. Magnuson, S. Kontur, O. Hellman, I. Tanaka and C. Draxl, *J. Phys. Chem. C*, 2019, **123**, 9688–9692.
57. T. Mizoguchi, T. Miyata and W. Olovsson, *Ultramicroscopy*, 2017, **180**, 93–103.

58. H. Katsukura, T. Miyata, K. Tomita and T. Mizoguchi, *Ultramicroscopy*, 2017, **178**, 88–95.
59. A. Seko, K. Toyoura, S. Muto, T. Mizoguchi and S. Broderick, *MRS Bull.*, 2018, **43**, 690–695.
60. C. Draxl and M. Scheffler, *J. Phys. Mater.*, 2019, **2**, 036001.
61. G. R. Schleder, A. C. M. Padilha, C. M. Acosta, M. Costa and A. Fazzio, *J. Phys. Mater.*, 2019, **2**, 032001.
62. A. Jain, K. A. Persson and G. Ceder, *APL Mater.*, 2016, **4**, 053102.
63. *Nanoinformatics*, ed. I. Tanaka, Springer, Singapore, Singapore, 2018.
64. S. Kiyohara, H. Oda, T. Miyata and T. Mizoguchi, *Sci. Adv.*, 2016, **2**, e1600746.
65. S. Kiyohara, H. Oda, K. Tsuda and T. Mizoguchi, *Jpn. J. Appl. Phys.*, 2016, **55**, 045502.
66. A. Seko, A. Togo, H. Hayashi, K. Tsuda, L. Chaput and I. Tanaka, *Phys. Rev. Lett.*, 2015, **115**, 1–5.
67. D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue and T. Lookman, *Nat. Commun.*, 2016, **7**, 11241.
68. P. V. Balachandran, D. Xue, J. Theiler, J. Hogden and T. Lookman, *Sci. Rep.*, 2016, **6**, 19660.
69. G. Pilania, C. Wang, X. Jiang, S. Rajasekaran and R. Ramprasad, *Sci. Rep.*, 2013, **3**, 2810.
70. M. Shiga, K. Tatsumi, S. Muto, K. Tsuda, Y. Yamamoto, T. Mori and T. Tanji, *Ultramicroscopy*, 2016, **170**, 43–59.
71. J. Timoshenko, D. Lu, Y. Lin and A. I. Frenkel, *J. Phys. Chem. Lett.*, 2017, **8**, 5091–5098.
72. T. Lam Pham, H. Kino, K. Terakura, T. Miyake, K. Tsuda, I. Takigawa and H. Chi Dam, *Sci. Technol. Adv. Mater.*, 2017, **18**, 756–765.
73. P. V. Balachandran, J. Theiler, J. M. Rondinelli and T. Lookman, *Sci. Rep.*, 2015, **5**, 13285.
74. V. Kvasniĉka, *J. Math. Chem.*, 1991, **6**, 63–76.
75. L. S. Anker and P. C. Jurs, *Anal. Chem.*, 1992, **64**, 1157–1164.
76. J. Cuny, Y. Xie, C. J. Pickard and A. A. Hassanali, *J. Chem. Theory Comput.*, 2016, **12**, 765–773.
77. J. Timoshenko, A. Anspoks, A. Cintins, A. Kuzmin, J. Purans and A. I. Frenkel, *Phys. Rev. Lett.*, 2018, **120**, 225502.

78. S. Kikuchi, H. Oda, S. Kiyohara and T. Mizoguchi, *Phys. B Condens. Matter*, 2018, **532**, 24–28.
79. H. Oda, S. Kiyohara, K. Tsuda and T. Mizoguchi, *J. Phys. Soc. Japan*, 2017, **86**, 123601.
80. S. Kiyohara and T. Mizoguchi, *Phys. B Condens. Matter*, 2018, **532**, 9–14.
81. S. Kiyohara and T. Mizoguchi, *J. Chem. Phys.*, 2018, **148**, 241741.
82. H. Oda, S. Kiyohara and T. Mizoguchi, *J. Phys. Mater.*, 2019, **2**, 034005.
83. S. Kiyohara, T. Miyata, K. Tsuda and T. Mizoguchi, *Sci. Rep.*, 2018, **8**, 13548.
84. S. Kiyohara, M. Tsubaki, K. Liao and T. Mizoguchi, *J. Phys. Mater.*, 2019, **2**, 024003.
85. S. J. Clark, M. D. Segall, C. J. Pickard, P. J. Hasnip, M. J. Probert, K. Refson and M. C. Payne, *Z. Krist*, 2005, **220**, 567–570.
86. T. Mizoguchi, I. Tanaka, S.-P. Gao and C. J. Pickard, *J. Phys. Condens. Matter*, 2009, **21**, 104204.
87. J. P. Perdew, K. Burke and M. Ernzerhof, *Phys. Rev. Lett.*, 1996, **77**, 3865–3868.
88. O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*, Springer US, Boston, MA, 2010.
89. S. Kim and X. Zhang, *Comput. Math. Methods Med.*, 2013, **2013**, 509761.
90. D. L. Tabb, M. J. MacCoss, C. C. Wu, S. D. Anderson and J. R. Yates, *Anal. Chem.*, 2003, **75**, 2470–2477.
91. L. Breiman, J. Friedman, C. J. Stone and R. A. Olshen, *Classification and Regression Trees*, Wadsworth, Belmont, CA, 1984, p. 358.
92. C.-H. Yeh, *Chemom. Intell. Lab. Syst.*, 1991, **12**, 95–96.
93. D. Kingma and J. Ba, *arXiv:1412.6980 [cs.LG]*, 2014, 1–15.
94. L. Bocher, E. Popova, M. Nolan, A. Gloter, E. Chikoidze, K. March, B. Warot-Fonrose, B. Berini, O. Stéphan, N. Keller and Y. Dumont, *Phys. Rev. Lett.*, 2013, **111**, 167202.
95. M. N. Grisolia, J. Varignon, G. Sanchez-Santolino, A. Arora, S. Valencia, M. Varela, R. Abrudan, E. Weschke, E. Schierle, J. E. Rault, J. P. Rueff, A. Barthélémy, J. Santamaria and M. Bibes, *Nat. Phys.*, 2016, **12**, 25.

96. C. Zheng, K. Mathew, C. Chen, Y. Chen, H. Tang, A. Dozier, J. J. Kas, F. D. Vila, J. J. Rehr, L. F. J. Piper, K. A. Persson and S. P. Ong, *npj Comput. Mater.*, 2018, **4**, 1–9.

CHAPTER 18

Autonomous Science: Big Data Tools for Small Data Problems in Chemistry

ANDREAS C. GEIGER, ZIYI CAO, ZHENGTIAN SONG, JAMES R. W. ULCICKAS AND GARTH J. SIMPSON*

[a] Purdue University, Department of Chemistry, 560 Oval Drive, West Lafayette, IN 47907, USA
*Email: gsimpson@purdue.edu

18.1 Introduction

Arguably, the greatest opportunities to capitalize on machine learning advances lie at the interface between data science and measurement science; algorithms can inform the choice of measurements, which in turn can serve as inputs to algorithms. In this framework, two general classes of data science challenges emerge. In the first, the ever-increasing volume of data produced by high throughput (HT) instrumentation generates databases that can subsequently be mined.[1–5] Alternatively, many chemical measurements are limited by instrumental, kinetic, or sample constraints, restricting the number of measurements that can be practically made to inform decisions. This latter case is the primary focus of the present work, in which "big data" tools are brought to bear to optimize the information content produced by the fewest possible experimental measurements.

In contrast, most machine learning methods have focused on analysis and training using large volumes of preexisting data, including those generated by HT chemical instrumentation. HT screening has been applied to cell-based microfluidics,[6–10] protein engineering,[11–15] metabolomics,[16–20] proteomics,[21–25] pharmacotoxicology,[26–28] nanomaterials synthesis,[29] and efforts in small-molecule synthetic optimization.[30,31] Techniques like these have enabled researchers to collect "big data" in ever shorter periods of time, which can then be mined for pattern-analysis and data-driven decision-making.

In turn, machine learning methods have been applied extensively in mining of chemistry databases, spanning a range of applications. Algorithms have been demonstrated with the capability of predicting chemical reactivities,[32,33] predicting compound properties[34–36] and determining chemical structures.[37] Zhuo and coworkers developed a machine-learning model that accurately predicted the band gap of inorganic solids solely from composition.[38] The model used support vector classification to separate metals from nonmetals and support vector regression to predict the band gap. The results from this method were closer to experimentally reported values than density functional theory calculations. Coley and coworkers used a graph-convolutional neural network to predict the major product of a reaction.[39] The network accurately predicted the major product 85% of the time, a significant improvement from previous methods. Turcani and coworkers predicted shape persistence and cavity size in porous organic cages using a random forest algorithm.[40] Their model achieved a 93% accuracy for the prediction of shape persistence. Advances like these have made it possible for chemists to leverage their "big data" to make impactful discoveries.

One obvious limitation on "big data" tools is the requirement that the data must be big. Many time-consuming experiments are incommensurate with conventional machine learning methods because of the intrinsic sparsity of the available data. An illustrative example is the National Ignition Facility at Lawrence Livermore National Laboratory, which houses the world's largest and most energetic laser.[41–43] The laser (formally a combination of 192 lasers) is only fired about 400 times per year ($\sim 10^{-5}$ Hz repetition rate). The capacity for such experiments to benefit

from machine learning tools lies in the ability to optimally select conditions for the next experiments based on the limited availability of preceding information. Such cases are arguably best described as “little data” problems, presenting both challenges and opportunities for integration into the emerging “big data” infrastructure.

These challenges are not new. Rather than random “shotgun” sampling, an experienced bench chemist working the laboratory with limited data availability will generally use the outcomes of the preceding set of measurements to inform the next selection of conditions (or more commonly, to know which conditions to avoid). Rationale for selection of the next experiment is typically made by combining the observations to date with the experience of the investigator in working with similar compounds in similar scenarios. Porting this decision-making process over to machine learning platforms involves methods capable of both capturing the experience of the bench chemist from general knowledge of related molecular systems and approaches for optimally selecting the most informative next set of experiments.

When coupled with experimental automation, this algorithm-driven independent design of experiments enables autonomous science, in which machine learning or AI algorithms are trained to design experiments, interpret results, and execute additional measurements through hardware automation. In principle, such systems operate independently in all phases of the scientific method. As shown in Figure 18.1, hypothesis-making, experimental design, data collection, data analysis, and hypothesis-testing are performed iteratively with all previous experiments informing the design of future experiments until chemical space is sufficiently mapped to provide an answer.

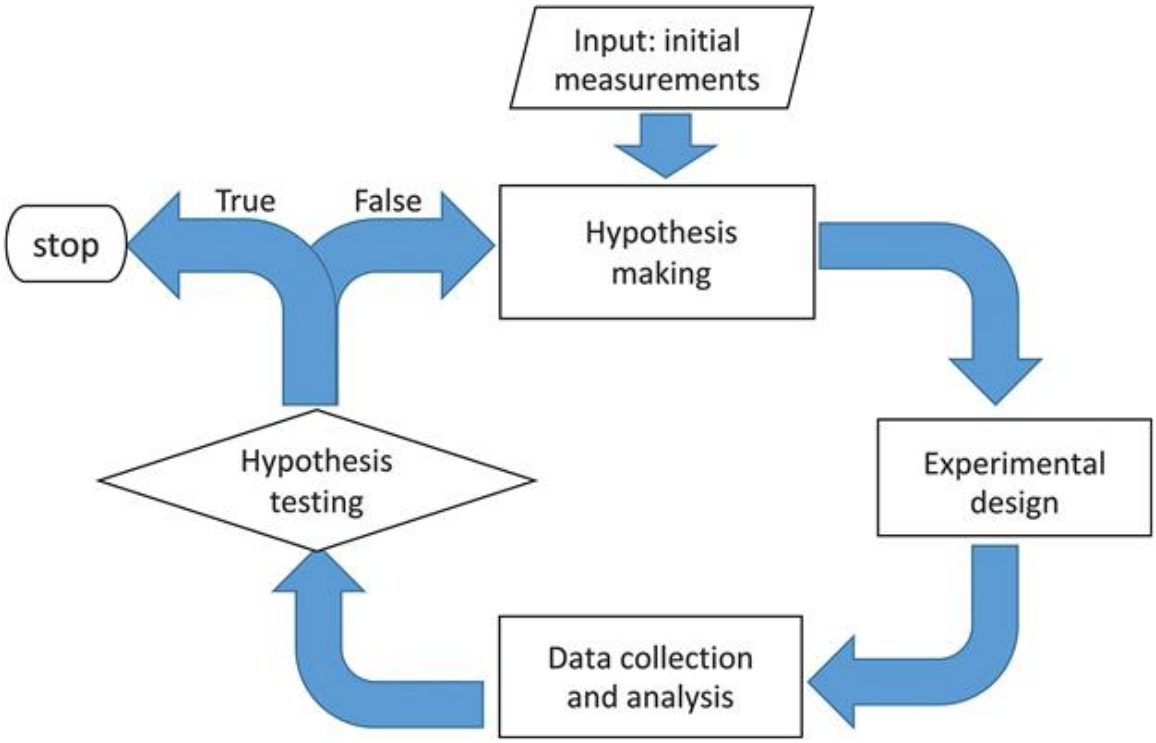


Figure 18.1 Diagram of the scientific method. An autonomous laboratory can independently operate in all phases of the scientific method. Adapted from ref. 54, https://doi.org//10.1107/s160057751601612x, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

Autonomous instrumentation should be distinguished from automated instrumentation. Many modern laboratories have increased experimental throughput by fully automating processes that were previously done by hand.[44–48] The emerging revolution in chemical measurement science is in automating the hypothesis-making and experimental design steps in the scientific method.[49–51] By delegating these tasks to algorithms, the loop can be closed to reduce human intervention in the scientific method.

The rest of this chapter will delve into topics that are relevant for the development of machine learning algorithms and autonomous instrumentation for optimized collection in “little data” situations, in which the number of experiments is limited by time, sample, or expense. An overview of autonomous experimental design and execution will detail strategies for implementing experiments with algorithms producing autonomous scientific decisions. One architecture for autonomous

chemistry will be reviewed in detail, building upon the concept of sparsely sampling chemical space, to dynamically sampling chemical space, and finally a supervised learning approach to dynamic sampling as applied to chemical imaging. Furthermore, a detailed discussion is presented on the potential for adversarial attacks against algorithms that are used in a chemical context. The discussion reveals that dimension reduction techniques are potentially susceptible to adversarial attacks and provides relevant context on the general robustness of machine-learning-based classifiers for autonomous science. Some suggestions and methods to improve general stability *versus* adversarial attacks and spurious misclassification are discussed.

18.2 Autonomous Design of Experiments

Autonomous science requires both the ability to act independently (automation) and to make independent decisions based upon evolving knowledge (intelligence). The focus of this chapter is squarely on the latter, as automation is a largely solved problem on many modern instruments supporting HT analyses. An autonomous system is an “expert” that can incorporate new information to revise a governing hypothesis and guide improvements. This definition can be broken down into two key constituent functions: the acquisition of new information, and the selection of future experiments. The focus here largely rests upon autonomous systems designed to reduce the experimental burden associated with mapping chemical space. Initial discussions will focus on a prerequisite capacity for functioning autonomous systems: the ability to extrapolate anticipated experimental outcomes from previously sampled data. Fixed pattern sparse sampling strategies will serve to illustrate some approaches that can be undertaken to achieve this goal. The discussion of sparse sampling will be followed by a survey of dynamic sampling approaches, which aim to reconstruct chemical space with the smallest number of measurements. These methods differ from fixed pattern sparse sampling strategies in that real-time experimental observations can be leveraged to inform future experiments on the fly. Various algorithms for dynamic sampling will be

discussed, first in the intuitively simple context of chemical imaging, and later in the more abstract context of chemical space (*e.g.*, protein concentrations in live cell cultures). This section will conclude with a brief discussion of modularized autonomous laboratories with multiple autonomous instruments, which utilize advanced software that integrates various modules (*e.g.*, communication, databases, robotics, characterization, learning, and analysis) to map highly complex feature spaces.

18.2.1 Sparse Sampling Strategies for "Small Data" Challenges

In cases where single experiments can be relatively costly, building a map of observables as a function of experimental parameters can be accelerated by using sparse sampling strategies. Chemical measurements tend to be highly correlated, providing opportunities for sparse sampling. Using chemical imaging as an example, the composition in a given pixel is likely to be highly dependent on the composition in adjacent pixels. As such, reasonably good estimates of the composition distribution can be made from sampling a subset of the total number of pixels and "inpainting" the unsampled locations. The presence of correlations between measurements is a fundamental requirement of autonomous science; if every measurement were completely independent, no rationale would be available for selecting any particular set of conditions. Learning the underlying patterns connecting chemical measurements is a critical first step in developing methods for autonomous science.

Applications involving chemical imaging can leverage advances in image recognition to assist in predicting spatial correlations in chemical composition. Using these tools, architectures for generating complete representations of chemical space from a limited number of sampling events have had great success in novel sampling strategies to sparsely collect data from point-probe microscopes (*e.g.*, Raman microscopy,[52] scanning electron microscopy (SEM),[53] synchrotron X-ray diffraction[54]). In point-wise image acquisitions, sparse sampling offers the potential to dramatically reduce the number of measurements. Sparse sampling methods have been implemented to minimize sample damage for beam-sensitive materials, reducing total time for data acquisition and minimizing sample

exposure to potential structural change from, *e.g.*, X-ray or electron dose.[55,56]

There are two main components of a sparse sampling strategy: the sparse sampling pattern and the "inpainting" algorithm. Figure 18.2 illustrates the general strategy in sparse sampling. The pattern used determines which experiments will be performed; in many cases, the most intuitive pattern is random, which generates a wide distribution of measurements to inform the "inpainting" algorithm and leaves minimal artifacts from the sampling pattern. However, random sampling can increase measurement time significantly compared to continuous sampling patterns. For example, in beam scanning fluorescence microscopy, the time required to re-position the laser beam at a random pixel (milliseconds) is far greater than the time required to measure a pixel (nanoseconds). In this case, a continuous sampling pattern (*e.g.* Lissajous trajectories) takes full advantage of the sampling speed while also sampling from a wide distribution of positions in the field of view. The optimal sampling pattern for a sparse sampling strategy will be specific to each technique and dependent on the slowest step in the measurement.

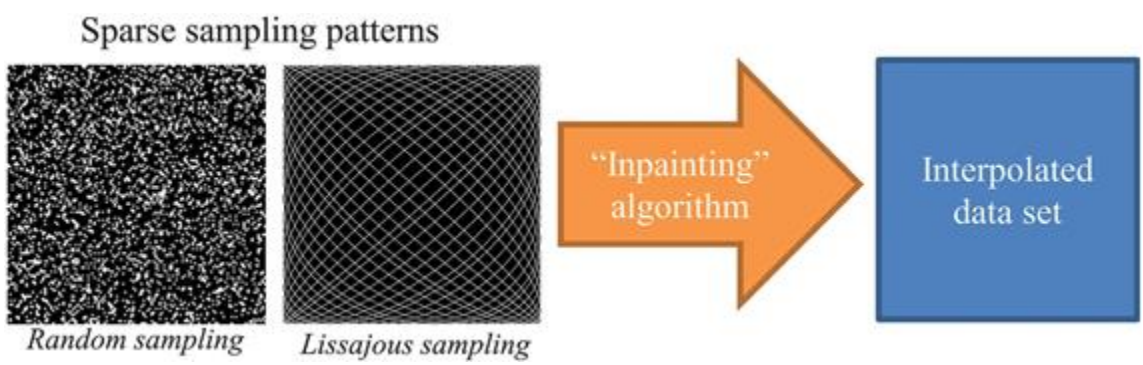


Figure 18.2 Sparse sampling combines fixed or random sampling patterns with "inpainting" algorithms. This reduces total measurement time by "inpainting" unsampled conditions and locations with predictions rather than performing measurements at every condition and location. Sparse sampling strategies rely on correlations between proximal data points.

In previous studies, sparse sampling patterns such as random sampling, low-discrepancy sampling, and Lissajous trajectories were employed to accelerate data collection. Simpson and coworkers developed a beam-

scanning optical microscope based on Lissajous trajectory imaging, which achieved up to kHz frame-rate optical imaging on multiple simultaneous data acquisition channels.[57,58] They used two fast-scan resonant mirrors to direct the optical beam on a circuitous trajectory through the field of view and used an "inpainting" algorithm to interpolate unsampled pixels.

Once sparsely sampled data are collected, it is the job of the "inpainting" algorithm to predict the values of unsampled conditions and locations. If only local correlations between measurements are considered, a smoothing function can be used for interpolation. Garcia and coworkers developed an "inpainting" algorithm that performs a smoothing function to fill in missing data points from a data set of arbitrary dimensions.[59,60] The algorithm predicts missing values using a penalized least squares regression. The cost function to be minimized is shown in eqn (18.1), where $\boldsymbol{y}$ is the sparsely sampled data set, $\hat{\boldsymbol{y}}$ is the smoothed, interpolated data set, $\| \|$ denotes the Euclidean norm, s is a parameter that controls the degree of smoothing, and P is a penalty term that represents the degree of roughness in the smoothed data set, defined as $\mathbf{P}(\hat{\boldsymbol{y}})=\|\boldsymbol{D}\hat{\boldsymbol{y}}\|^2$ where $\boldsymbol{D}$ is a tridiagonal matrix, which describes the distance between sampled pixels.

$$\mathrm{F}(\hat{y}) = \|\hat{y} - y\|^2 + s\mathrm{P}(\hat{y}) \tag{18.1}$$

The minimization of $\mathbf{F}(\hat{\boldsymbol{y}})$ gives the linear system in eqn (18.2), where $\boldsymbol{I}$ is the identity matrix.

$$(I + sD^{\mathrm{T}}D)\hat{y} = y \tag{18.2}$$

To account for missing data points, a diagonal matrix $\boldsymbol{W}$ is added to eqn (18.2) in the place of $\boldsymbol{I}$, where the diagonal elements W_{ii} are equal to 0 when y_i is missing and 1 when y_i is present.

$$(W + sD^{\mathrm{T}}D)\hat{y} = y \tag{18.3}$$

Eqn (18.2) and (18.3) can be solved by left-matrix division. When sparsely sampled data are evenly spaced, a discrete cosine transform can be used to simplify and speed up the solution.

Other algorithms go beyond smoothing and look for non-local correlations in data sets. Bouman and coworkers have demonstrated model-based iterative reconstruction (MBIR) approaches for accurately performing in-painting.[61] In MBIR, forward models provide a mechanism for assigning the unsampled pixels, based in part on training using similar images.

Patterned sparse-sampling approaches provide a key step toward autonomous decision-making in imaging by enabling image reconstruction from subsampled measurement sets; however, these methods are limited in the sense that they do not necessarily sample the optimal pixels for an accurate reconstruction of the system. Reconstruction algorithms employed in fixed-pattern or random sparse sampling techniques provide a starting point for extension to autonomous laboratory systems, which aim to reconstruct descriptions of chemical space with maximal accuracy under the constraint of minimal sampling.[59] Such methods, referred to here as dynamic sampling methods, enable more rapid reconstruction of chemical space by deciding the optimal positions to sample within chemical space.[62–64]

18.2.2 Simplex Approaches for Autonomous Design of Experiments

Early work in autonomous experimental design focused on both chemical synthesis and optimizing chemical measurements through the simplex architecture of algorithm design. The simplex is a specific type of dynamic sampling strategy, where some probe of chemical space is mapped to a geometric representation with an arbitrary number of vertices and sides, a simplex. Beginning along one vertex of the polytope, which can be thought of as a discretized chemical space, a simplex algorithm is used to find the maximum value within the feasible region *via* walking along the edges of the shape in the direction of increasing values of the objective function.

The modified simplex algorithm was described in detail by Nelder and Mead as an algorithm for function minimization.[65] A function with n variables without constraints is minimized first by sampling n+1 points within the parameter space. Each sampled point is the vertex of the geometric simplex, giving rise to the method's name. To minimize the objective function, the output of each parameter set is first computed. The

vertex with the highest valued objective function (*i.e.*, the worst performing set of parameters) is then reflected about the hyperplane of the remaining vertices, across the centroid of the hyperplane. Depending on whether the new point is an improvement over the previous vertex, the operations of expansion or contraction can be utilized, which shift the sampled position according to the local curvature of the objective function about the reflection. When iterated, this procedure will generate a simplex that converges about the minimum of the objective function, *i.e.*, the volume of the simplex will contract about the minimum point. Figure 18.3 illustrates the procedure for 2-dimensional optimization.

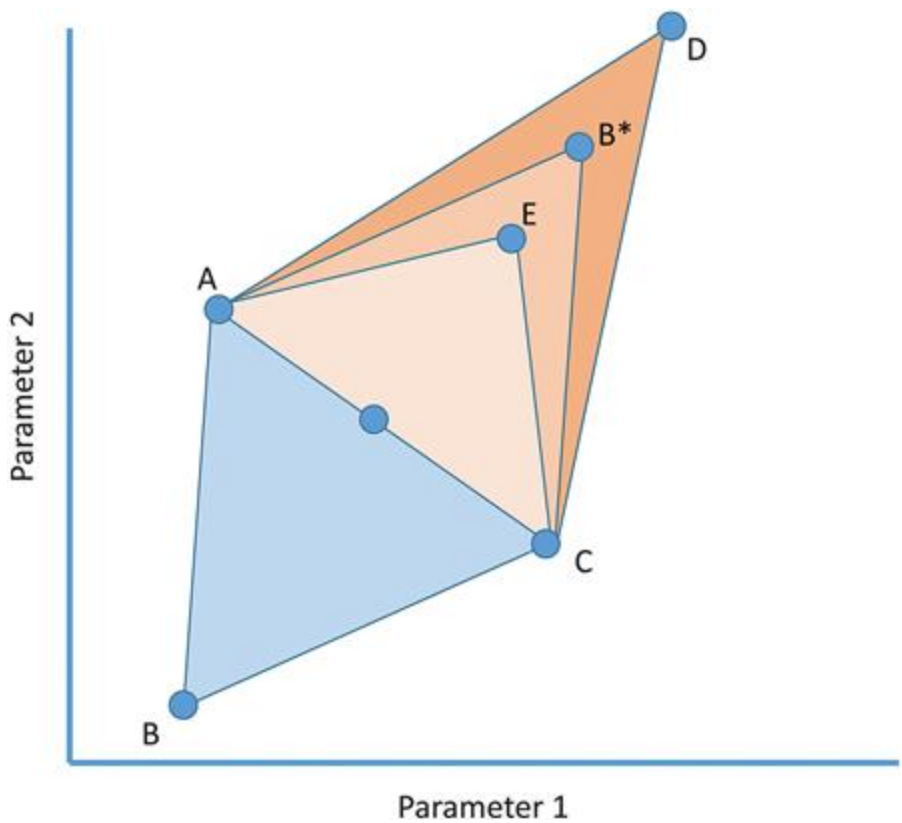


Figure 18.3 A 2-dimensional simplex optimization step. The initial simplex is defined by the vertices ABC. The objective function is computed for each set of parameters, and B is found to be the worst input. The reflection B* is generated about the centroid of the line segment AC. The algorithm can then either expand the simplex, accepting a new vertex D in place of B*, or contract the simplex, accepting E in place of B*, according to local curvature.

Beginning in the 1970s, simplex algorithms were applied to chemical automation, including the genesis of closed-loop automated synthetic platforms.[64,66,67] Other work utilized simplex algorithms to generate H_2O_2 complexed with Ti(IV) in the presence of EDTA; automated injection of reagents was followed by measurement of the absorbance spectrum, with wait times added to assess the stability of the product.[68] The optimization of aerosol delivery in a flame spectrophotometer by Denton and coworkers serves as a simple example to illustrate how the simplex procedure can be utilized for automated analysis.[66] In that work, the physical position of the aerosol spray nozzle is the optimization space, with stepper motors controlling *x* and *y* position on orthogonal axes. The aim was to generate the maximum possible signal detected at the calcium emission line of 422.7 nm. Using Figure 18.3 as a template, consider the scenario where three unique sets of coordinates are used to measure calcium emission, represented by the vertices A, B, and C. Vertex B is found to result in the lowest amount of signal, so a new vertex is generated by reflecting across the line AC. The outcome of measurement at the new vertex B* determines whether the simplex will be kept at this new location, expanded, or contracted. The mathematical details of this procedure are described fully in ref. 65. These initial simplex-based approaches to autonomous chemical experimentation use relatively simple algorithms, however they serve to illustrate the general architecture of dynamic sampling approaches to autonomous experimentation, with one key difference. The simplex approach aims to optimize a single function, returning only the optimal value. In contrast, the dynamic sampling approaches described below reconstruct a full objective function using various interpolation and reconstruction algorithms. By mapping the full space of the objective function, dynamic sampling approaches can address more complex chemical problems, such as chemical classification.

18.2.3 Interpolation Algorithms for Dynamic Sampling

Dynamic sampling is a critical component in autonomous instruments, as it provides real-time feedback about the most beneficial experiment to be conducted with respect to the targeted goal. Dynamic sampling can enable

instruments to advance in performance beyond automated, HT systems by increasing the value of each experiment.

Notingher and coworkers developed a dynamic sampling algorithm for Raman microscopy to determine the most informative sampling location by assigning a score to each unmeasured pixel in real time.[69,70] The score was calculated at each unsampled pixel as the difference between the values predicted by two different interpolation algorithms, a cubic spline, and a Kriging interpolation. The pixel location where the reconstruction algorithms differed the most was treated as the most informative pixel and then selected for subsequent measurement. Due to the integration of interpolation algorithms, this method showed an improvement in efficiency over random sampling. For biological tissue imaging, this method achieved up to 30 times reduction in the sampling time compared to raster-scanning, with sufficient spectral signal-to-noise ratio to identify individual tissue structures.

One strategy for further improving the performance of dynamic sampling is to use an algorithm that is trained prior to analysis. This approach, a supervised learning approach for dynamic sampling (SLADS) has been developed to improve the accuracy of dynamic sampling algorithms through training data that is specific to the sample under study.

18.2.4 A Supervised Learning Approach for Dynamic Sampling

The SLADS algorithm, developed by Bouman and coworkers, represents an illustrative case-study for understanding the general architecture of autonomous laboratory workflows.[62–64] SLADS uses a machine-learning-based algorithm to select the next most information-rich location to sample during real-time measurements. However, in contrast to other dynamic sampling algorithms, SLADS uses training data with a known ground truth to train the algorithm to calculate the expected reduction in distortion (ERD) for each sampling location. Leveraging this training, SLADS is capable of dramatically reducing the number of measurements required to obtain a reconstructed image without a significant loss in image quality. The theoretical framework of SLADS is demonstrated by the flow chart in Figure 18.4 and eqn (18.4)–(18.6).

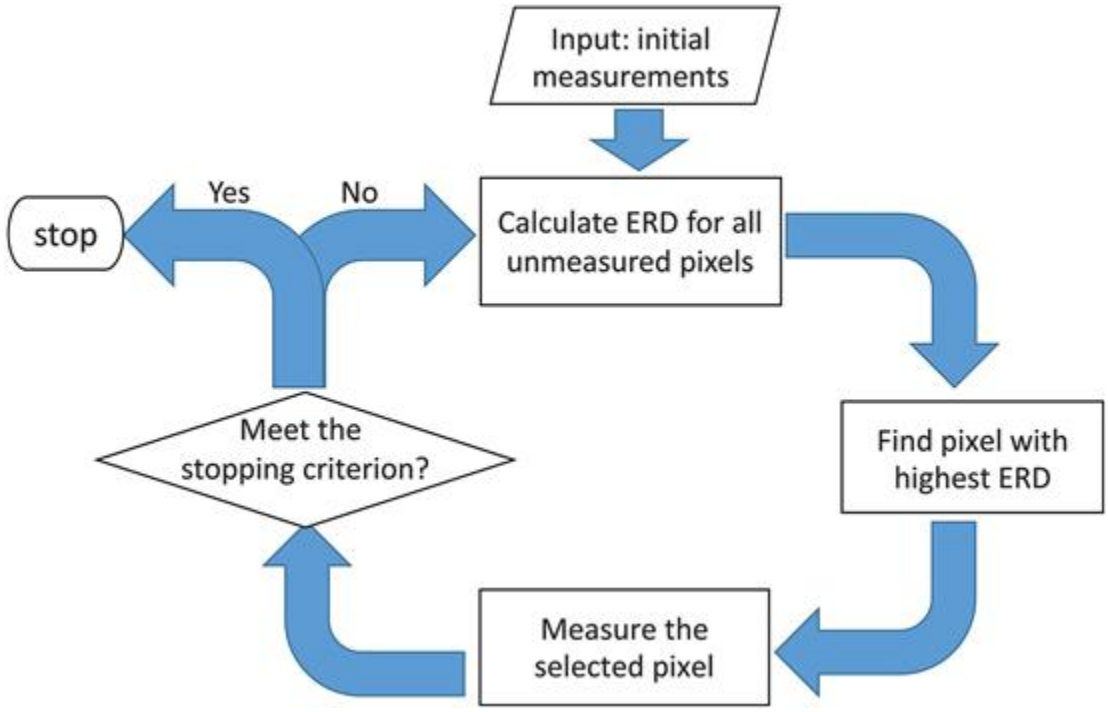


Figure 18.4 The SLADS algorithm samples the unmeasured pixel with the highest ERD until the stopping criterion is met. Adapted from ref. 54, https://doi.org//10.1107/s160057751601612x, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

The SLADS algorithm represents a 2D image as a matrix $\boldsymbol{X}$. In eqn (18.4), $\boldsymbol{X}_r$ is a single pixel of the image at location $r \in \Omega$. The location of each of the k measured pixels are encoded within $\boldsymbol{S}=\{s^{(1)}, s^{(2)}, \ldots s^{(k)}\}$. Intuitively, $\boldsymbol{Y}^{(k)}$ can be interpreted as storing all known measured information about the image.

$$\boldsymbol{Y}^{(k)} = \begin{pmatrix} s^{(1)}, X_{s^{(1)}} \\ \vdots \\ s^{(k)}, X_{s^{(k)}} \end{pmatrix}. \tag{18.4}$$

$\boldsymbol{Y}^{(k)}$, as a $k \times 2$ matrix of sampling location and measured values, can then be used to reconstruct an image $\hat{\boldsymbol{X}}^{(k)}$ which is the best estimate of the ground truth image $\boldsymbol{X}$. SLADS is then employed to find the next sampling location, which maximizes the ERD, $E[\boldsymbol{R}^{(k;s)}|\boldsymbol{Y}^{(k)}]$ given in eqn (18.5),

where $\boldsymbol{R}$ is the reduction in distortion. Maximizing the ERD results in reconstructing an image with the minimum amount of error relative to the true image.

$$s^{(k+1)} = \arg \max_{s \in \{\Omega \setminus S\}} \left\{ E\left[\boldsymbol{R}^{(k;s)} \mid \boldsymbol{Y}^{(k)} \right] \right\} \tag{18.5}$$

Eqn (18.6) shows the calculation of $\boldsymbol{R}$ resulting from measuring pixel s. $\hat{\boldsymbol{X}}^{(k)}$ is the reconstruction result made by $\boldsymbol{Y}^{(k)}$, while $\hat{\boldsymbol{X}}^{(k;s)}$ is the reconstruction made from $\boldsymbol{Y}^{(k)}$ and X_s, the next sampled location. $D(\boldsymbol{X}, \hat{\boldsymbol{X}}^{(k)})$ represents the distortion between the ground truth image $\boldsymbol{X}$ and the reconstructed image $\hat{\boldsymbol{X}}^{(k)}$ with k measurements. In short, selection of an optimal new sampling location X_s will maximize $\boldsymbol{R}$, reproducing the true image with minimal distortion.

$$R^{(k;s)} = D(X, \hat{X}^{(k)}) - D(X, \hat{X}^{(k;s)}). \tag{18.6}$$

As the ground truth image $\boldsymbol{X}$ is not available during image acquisition, the ERD needs to be calculated from $\boldsymbol{Y}^{(k)}$. A function relating $\boldsymbol{Y}^{(k)}$ to the ERD, $f^s_\theta(\boldsymbol{Y})$, is learned through supervised learning with a training set of known ground truth. The overall SLADS architecture thus updates the sampled locations stored in $\boldsymbol{Y}^{(k)}$, at each step, and the optimal next pixel location $s^{(k+1)}$ is determined by a supervised learning algorithm trained to maximize the ERD.

18.2.5 SLADS for Raman Hyperspectral Imaging and X-Ray Diffraction Imaging

In recent studies, SLADS has shown broad application in point-wise imaging techniques such as confocal Raman microscopy, synchrotron X-ray diffraction imaging, and SEM. Simpson and coworkers implemented SLADS into confocal spontaneous Raman imaging, which was an early use of a machine learning approach for sparse sampling Raman imaging.[52] Integration of SLADS into the feedback for beam positioning enabled fully autonomous control over the selection of location and data acquisition. By using this approach, chemical images of pharmaceutical materials were

acquired with >99.5% accuracy from 15.8% sampling, representing a ∼6-fold reduction in measurement time compared to conventional raster scanning methods.

The SLADS algorithm was designed to select optimal sampling locations within images, while each pixel in a Raman image included a Raman spectrum with thousands of elements. As such, utilization of SLADS with Raman was enabled by preprocessing and classification of acquired Raman spectra to identify the chemical composition of the sample measured at specific locations. SLADS was then able to select the next pixel based on the greatest uncertainty in its predicted class. Classification was conducted by combining two supervised learning algorithms: linear discriminant analysis (LDA) and support vector machine (SVM).[71,72] LDA was performed for initial dimensional reduction, followed by SVM for classification by constructing optimal hyperplanes in data space to separate different clusters of data points. The SLADS algorithm was iterated until the maximum ERD of the reconstructed image converged below a threshold.

Simulated random sampling experiments were performed to compare the performance and efficiency of SLADS with other alternative sparse sampling methods. Figure 18.5D shows the image reconstruction error for random sampling and SLADS; the error achieved with random sampling was 4.65% with a 15% sampling rate, which was 20 times higher than the error with SLADS. Most incorrectly classified pixels obtained from random sampling were located at the boundaries of the sample particles in different classes. From an image analysis perspective, these boundaries can be considered to contain high spatial frequency information compared to other positions in the image, which means the boundaries are the most ambiguous and informative positions for the measurements. Random sampling does not adjust its measurement density according to the different spatial frequency at different positions, which significantly reduces its efficiency. SLADS adjusted the measurement density and selectively measured more pixels with higher spatial frequency information, enabling image reconstruction with a much lower error percentage than is achievable *via* random sampling with the same sampling rate.

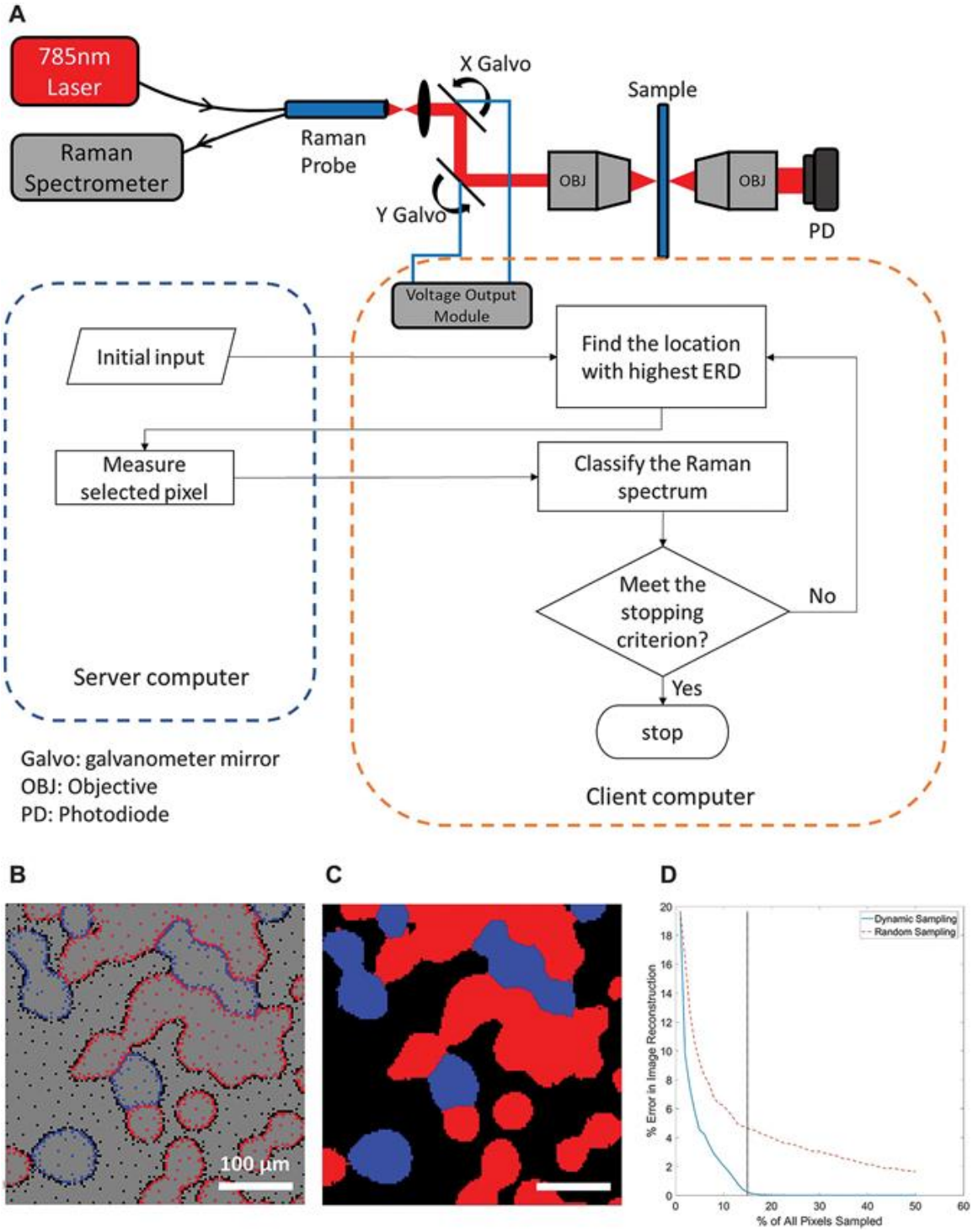


Figure 18.5 Raman hyperspectral imaging with SLADS. (A) Instrument schematic showing a Raman microscope with a dual-galvo scan mirror pair. SLADS experiments were carried out through automated control of the mirror pair to position the laser beam at the pixel with the highest ERD. (B) Dynamically sampled image after classification at

each sampled pixel. (C) Reconstructed image using trained SLADS algorithm. (D) Plot comparing reconstruction error between random sampling and dynamic sampling. Adapted from ref. 52 with permission from American Chemical Society, Copyright 2018.

Simpson and coworkers also implemented the SLADS algorithm with synchrotron X-ray diffraction experiments, which significantly reduced both dose and measurement time due to crystal positioning.[54] Crystal centering is required for macromolecular diffraction at synchrotron facilities, with X-ray diffraction mapping growing in popularity as a mechanism for localization. In X-ray raster scanning, diffraction is used to identify the crystal positions based on the detection of Bragg-like peaks in the scattering patterns. However, this additional X-ray exposure may result in detectable damage to the crystal prior to data collection. SLADS reconstructed the X-ray image with a 31% sampling of the total volume and only 9% of the interior of the crystal, greatly reducing the X-ray dosage on the crystal. When implemented into the beamline at Argonne National Laboratory an acceptable reconstruction was obtained with 3% of the image sampled corresponding to approximately 5% of the crystal. Dynamic sampling is well-matched to problems in which the cost of single-pixel acquisition limits the overall imaging time, consistent with diffraction imaging described herein. The cost here is not only in measurement time but also in sample integrity through X-ray induced damage.

SLADS is a powerful tool to accelerate 2D imaging, but extension to higher dimension remains to be demonstrated. Conceptually, dimensionality can be extended most trivially by incorporating time-domain information or a third spatial dimension. Adding in time-dependent changes in chemical composition adds another layer of complexity to these measurements. However, the curse of dimensionality makes higher dimensional reconstructions more difficult to solve. As dimensionality increases, the fast growth in the volume of measurement space also increases the data sparsity. To obtain a reliable statistical significance, the amount of data should increase exponentially with dimensionality. Furthermore, dimension reduction approaches also become increasingly challenging, either from a need for more training data to achieve the same resolution in projection onto a lower-dimensional space, or by covariance obfuscating dimension reduction. Therefore, a key step in facilitating extrapolation to higher

dimensions is the ability to generate robust dimension reduction techniques and classifiers capable of overcoming these challenges for autonomous decision-making. Consequently, understanding where and how these machine learning methods fail will facilitate the expansion of autonomous measurement methods to higher dimensions, as described in greater detail in Section 18.3 on adversarial attack strategies. However, before strategies for improving classifiers are considered, the following discussion surveys dynamic sampling methods used to map more complex chemical spaces in cellular biology and materials science.

18.2.6 Noniterative Dynamic Sampling for Autonomous Design of Experiments

One example of dynamic sampling applied to a complex chemical space was performed by King and coworkers who built a “Robot Scientist” named “Adam” that could generate and test hypotheses in functional genomics space.[73] Adam was designed to operate almost completely autonomously while measuring growth rates of microbial samples. In this work, the robot was applied to determine the genes that encoded some of the enzymes in a yeast organism, *S. cerevisiae*. Adam used prior knowledge about the functional genomics of *S. cerevisiae* from a database to select the most informative experiments to perform. Adam generated 20 hypotheses about the connections between 13 pairs of genes and enzymes (*e.g.* gene X encodes enzyme Y). After performing the experiments and analyzing the data, Adam confirmed 12 of the 20 hypotheses (with $P<0.05$ for the null hypothesis). Researchers confirmed some of Adam's results by expressing the proteins from the genes that Adam concluded to be encoding the yeast enzymes. Manual enzyme assays with the expressed proteins confirmed three of the 12 conclusions, while review of the scientific literature provided strong evidence for six conclusions and revealed a possible error in one conclusion made by the robot.

The key innovation in the development of Adam was the integration of models, databases, and software to make this autonomous genomics experiment possible. A logical model, designed with metabolites as nodes and enzymes as connections between nodes, was used to encode knowledge about the metabolism of *S. cerevisiae*.[74] Information connecting genes with

the proteins they encode was obtained from a bioinformatics database, which in turn informed the hypothesis generating software used by Adam. As shown in Figure 18.6, the software operated in the following stepwise fashion,: 1) find all reactions in the model of *S. cerevisiae* that are orphans (enzyme is not linked to a known gene), 2) determine which of these reactions would affect cell growth, 3) find the enzyme commission (EC) class of these reactions, 4) find the genes that express enzymes in the same EC class in other organisms, 5) find genes in *S. cerevisiae* with similar sequences to those in other organisms 6) make hypotheses linking genes with orphan enzymes. Additional software was used for experimental design, laboratory automation, and data analysis.

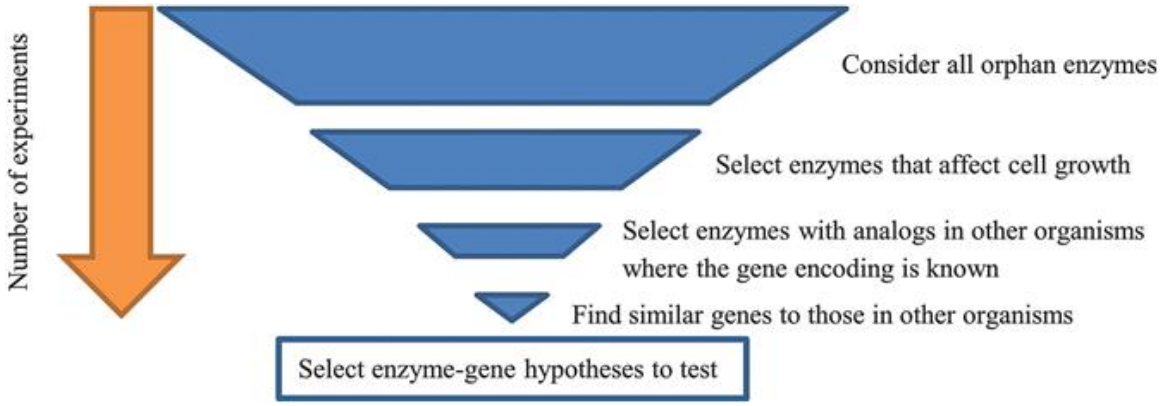


Figure 18.6 Diagram of the dynamic sampling framework used in Adam. At first, all known orphan enzymes in a species are considered. Then, enzymes candidates are selectively eliminated based on a predetermined set of criteria (impact on cell growth, similarity to well-characterized enzymes in other organisms, *etc.*). The most critical experiments are selected based on the most probable hypotheses.

The experimental selection procedure that Adam used differs from other dynamic sampling methods in that it is noniterative. Adam is designed to synthesize knowledge from various databases and models to narrow down the set of possible experiments to the most informative hypotheses. In contrast, more recent work in dynamic sampling has explored the ability to generate *de novo* hypotheses from real-time experimental outcomes, as described elsewhere in this chapter (18.2.3–18.2.5, 18.2.7–18.2.9). The noniterative framework of Adam is well-suited for the functional genomics questions at hand because of the large amount of information available in

databases. Experiments in similar fields with high availability to databases could likewise benefit from a noniterative approach to dynamic sampling.

18.2.7 Active Machine Learning

Innovative work in autonomous biochemical research has been accomplished by Murphy and coworkers in their application of active machine learning to investigate the effects of active compounds on proteins within cells. Active machine learning is simply a strategy for experimental design based on observed data whereby the strategy adapts the experiment dynamically to account for real-time measurements.[75–78] Elsewhere in this chapter, we refer to active machine learning as dynamic sampling. In their first publication, Murphy and coworkers tested their active machine learning algorithms with simulations and experimental results of drug activity on gene expression.[79] The active machine learning process presented in this work is described in Figure 18.7. (A) The experiments are codified by target and condition. (B) Samples are clustered based on the phenotype revealed by experimental measurements. (C) Correlations between targets and conditions are identified based on the phenotypic distributions. (D) The next set of experiments is chosen based on gaps in a predictive model. Steps B–D are repeated until a stopping condition is met.

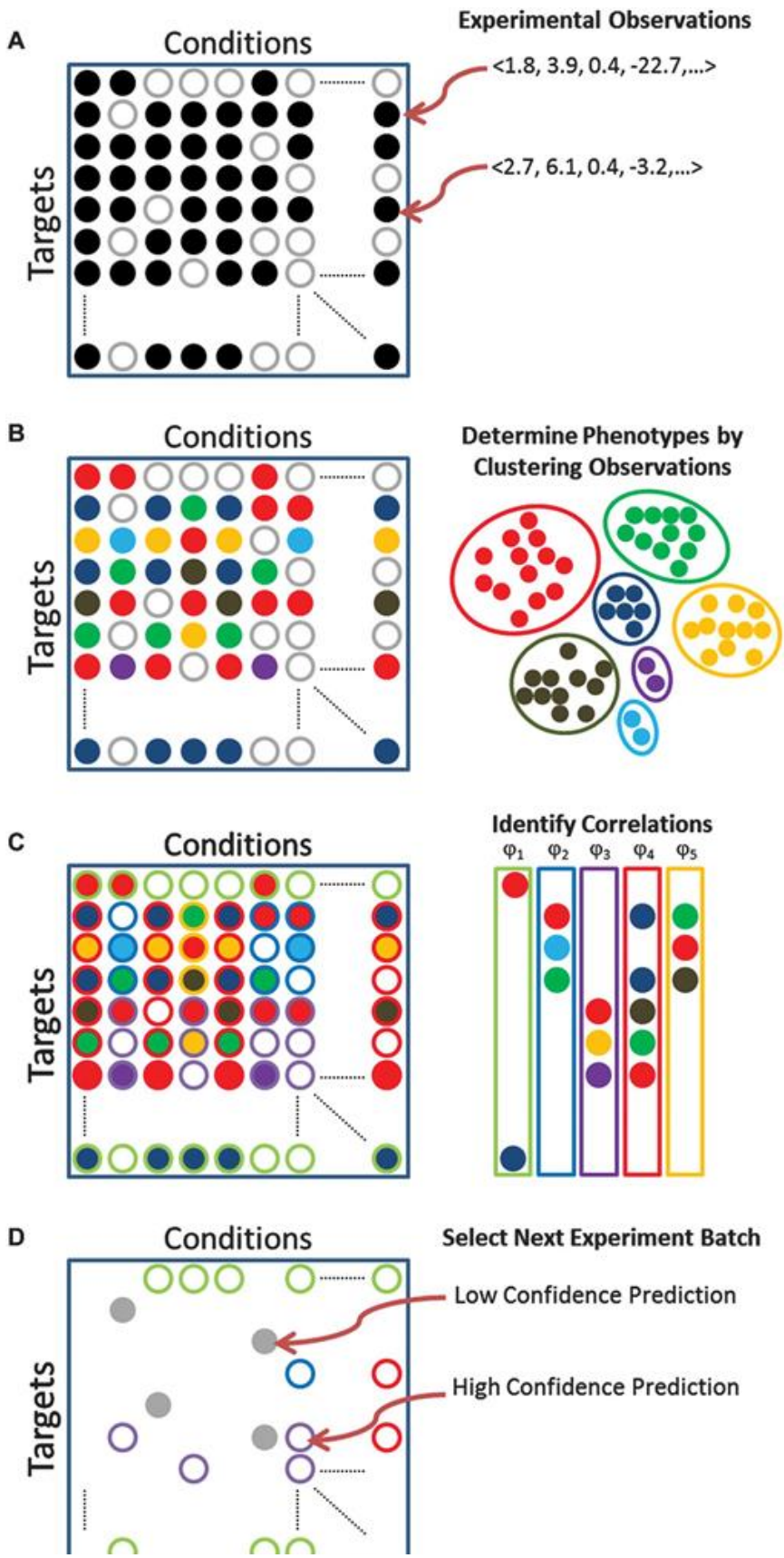

A
Conditions
Targets
Experimental Observations
<1.8, 3.9, 0.4, -22.7,...>
<2.7, 6.1, 0.4, -3.2,...>
B
Conditions
Targets
Determine Phenotypes by Clustering Observations
C
Conditions
Targets
Identify Correlations
φ1 φ2 φ3 φ4 φ5
D
Conditions
Targets
Select Next Experiment Batch
Low Confidence Prediction
High Confidence Prediction

Figure 18.7 The active machine learning process described by Murphy and coworkers. (A) Samples are codified by targets and conditions (B) Samples are clustered into phenotypes based on experimental observations. (C) Correlations are identified between target, condition, and phenotype. (D) The next set of experiments is chosen. Reproduced from ref. 79, https://doi.org/10.1371/journal.pone.0083996, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

A key component of this method, and every dynamic sampling method, is the underlying predictive model that informs future measurements. In this work, the predictive model comprised a set of distributions that described the phenotypic dependence on targets and conditions. The model was constructed using clustering algorithms, which classified observations into phenotype groups. These algorithms also identified correlations between measurements and produced distributions, which described the correlations and collectively made up the predictive model. Greedy Merge and B-Clustering algorithms were used in this work. In this active machine learning method, the next experiment is chosen based on a score given to every unobserved experiment, based on the number of distinct predictions that can be made for each experiment. It is possible that one unobserved experiment could be described by more than one predicted distribution and therefore more than one predicted phenotype. The experiments for which the model had the most predicted phenotypes were assumed to be the most informative experiments to perform and were selected for subsequent analysis.

In their second publication, Murphy and coworkers integrated their active learning algorithm into an instrument to enable autonomous science in a real-time experiment.[80] They investigated the effect of 48 chemical compounds on the subcellular localization of 48 proteins. Experiments were performed with an automated fluorescence microscope, an automated liquid handling robot for transferring cell culture and chemical compounds, and automated image analysis software. 48 different clones expressing different EGFP-tagged proteins were generated and treated with one of 48 different compounds. The active machine learning algorithms guided the instrument in real-time to select the most informative protein-drug pair for the next experiment, based on prior results. Cells were imaged using the

fluorescence microscope six hours after the selected drug was added to the culture media. Subcellular localization of the EGFP-tagged protein was determined using an image analysis method which produced a vector representation for each cell describing the positions of protein and DNA stains relative to each other.[81–84] A predictive model was updated on the fly during data collection until a stopping criterion was met. At the end of the experiment, the autonomous instrument achieved a 92% accuracy with only 29% sampling of all possible experiments.

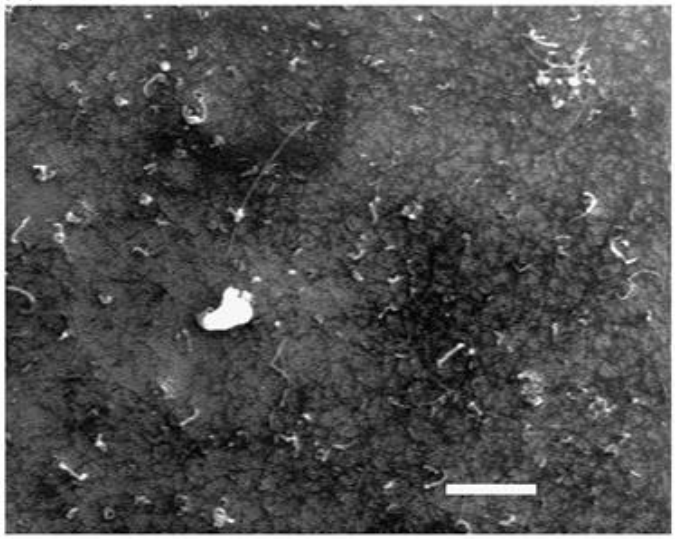
18.2.8 Dynamic Sampling for Materials Synthesis

Research in materials synthesis has also leveraged autonomous chemical measurement, as demonstrated by Maruyama and coworkers from the Air Force Research Laboratory, who developed an Autonomous Research System (ARES).[85] ARES optimized the growth rate of single-walled carbon nanotubes (CNTs) through the combination of an automated growth reactor, rapid *in situ* characterization, and an AI system. The system utilized an AI planner to iteratively propose new experiments in order to optimize CNT growth rate without *a priori* knowledge of the underlying physical processes. CNTs were synthesized in each experiment by a chemical vapor deposition process in which the growth conditions were varied (temperature, pressure, and partial pressures of ethylene, hydrogen, and water vapor). A laser was used to simultaneously heat the sample and as the excitation source for Raman spectroscopy.[86,87] The change in intensity of a characteristic Raman band was used as the readout for CNT growth rate and the experimentally determined growth rate was used as the feedback mechanism for the ARES system in its objective to optimize CNT growth.

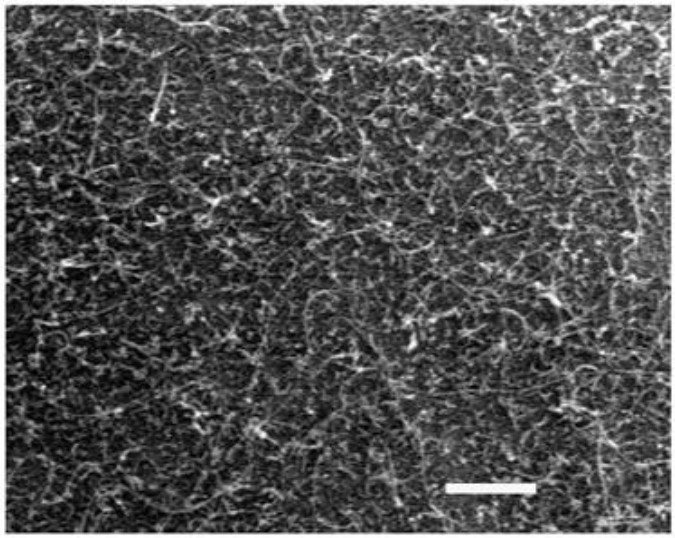
The ARES system used a random forest model to make predictions about the CNT growth rate for untested conditions. A random forest is a machine learning method related to a decision tree.[88–91] A decision tree is an algorithm designed to perform a classification or regression on a data set through a branching series of decision nodes, where the most discriminate feature is used to divide the data set into two subsets.[92–94] A random forest is simply a set of decision trees in which each decision tree is trained using a random subset of the training data and a random set of features to divide the data. The averaged response of all the decision trees in the random

forest is used as the output of the algorithm. In this case, an initial set of 84 user-designed experiments was provided to the random forest model as a training set to enable operation in autonomous mode. The random forest model then predicted experimental conditions that would achieve the user-specified target growth rate. After each experiment, the random forest model was updated to account for the most recent results and the experiment with the closest predicted growth rate to the target was performed. This process was iterated until the difference between the predictions and the experimental measurements of the growth rate converged within the noise floor of the experimental results. Figure 18.8 shows SEM images of CNTs produced by the ARES system. The images show samples with CNT growth rates of a) 500 s^{-1}, b) 3000 s^{-1}, and c) 16 000 s^{-1}. ARES was shown to design, execute, and analyze its own experiments 100 times faster than conventional experiments and represents an early demonstration of an autonomous research system in materials science.

(a)



(b)



(c)

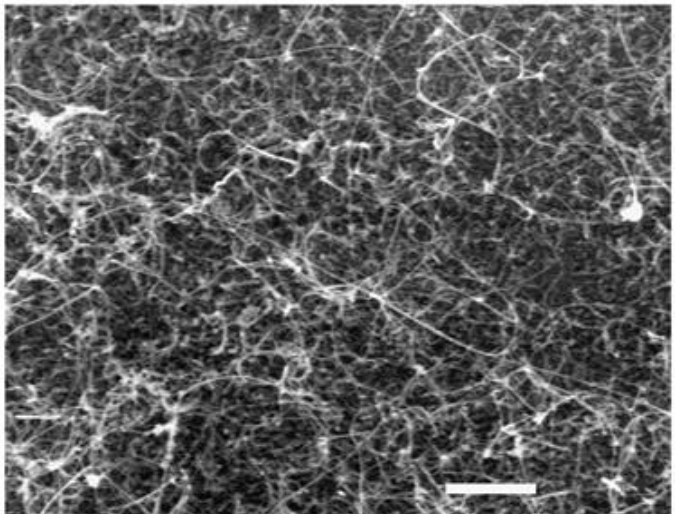


Figure 18.8 SEM images of CNTs produced by the ARES system. Images show CNT growth rates of (a) 500, (b) 3000, and (c) 16 000 (s^{-1}). Scale bar is 500 nm. Reproduced from ref. 85, https://doi.org/10.1038/npjcompumats.2016.31, under the terms of the CC BY 4.0 licence, https://creativecommons.org/licenses/by/4.0/.

18.2.9 Modular Architectures for Autonomous Design of Experiments

The preceding sections have focused on autonomous design of experiments where often only a single autonomous agent governs the process. In each case, a single measurement was conducted, which then provided feedback for the selection of the next best parameter set to sample. If multiple instruments are required to probe the chemical problem of interest, a central hub is needed to control communication between instruments, execution of experiments, evaluation of the objective function, and selection of the next experiments. Figure 18.9 illustrates one possible architecture that could be utilized for an autonomous laboratory.

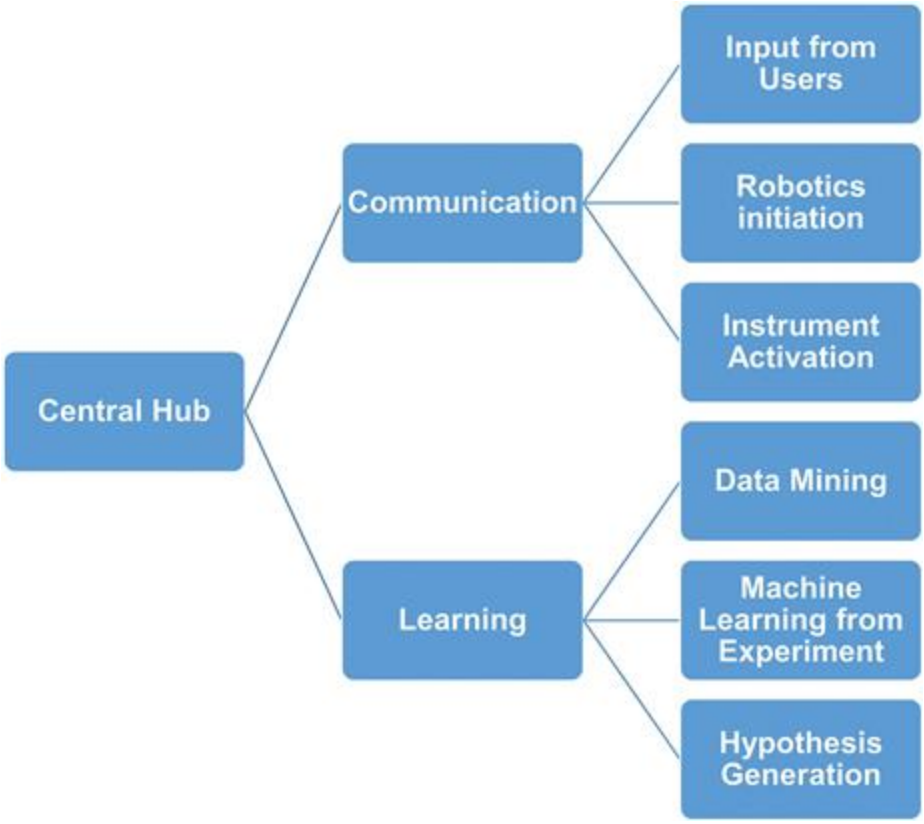


Figure 18.9 Sample modular layout for the autonomous laboratory. Data handling is delineated into two general protocols: communication, where data are transferred from one module to another, and learning, where the hub utilizes available data to generate models. Each module on the far right is considered as a distinct element of the whole architecture, enabling the addition of new modules as necessary to add to existing capabilities.

To illustrate how the central hub might interface with the individual modules, consider a hypothetical scenario where the aim is to cook the most delightful omelet. In the described scenario, each module has both an input and output flow of information. First, the hub might access the internet with its data mining module, providing the input of targeted databases. The data mining module has been outfitted with several different data mining approaches, including an image and video analysis tool, which extracts cooking tutorials in video form, as well as text-learning modules that mine recipes from the internet. These modules individually access and learn key features of the data. The learned features are then passed back to the central hub, which relates the learned features to the "machine learning from

experiment" module. This module can be used to generate a regression model predicting key input variables, such as the number of eggs, the temperature of the cooking vessel, amount of salt to add, time of cooking, *etc*. This module is distinct from the data mining module, as it can follow an active learning/dynamic sampling architecture where subsequent information gathered from experiment is incorporated. The regression model is fed back to the hub, and the hypothesis generation module is then called in to predict the set of parameters for the optimal omelet. With the predicted set of parameters, the hub must then communicate to activate the robotics that control the cracking of eggs, the beating of eggs, the addition of other ingredients, *etc*. Each robotic action is accompanied with an activation of the module, initiating the task, and successful execution of the programmed motion will return completion back to the hub to initiate each subsequent robotic motion. In this example, the beginning of the cooking process corresponds to the activation of the instrument. Once cooking has been completed, the hub receives a completion command. As the deliciousness of an omelet requires human testing, the hub can then prompt the user that the output is ready. Various factors relating to the quality of the omelet such as taste, temperature, mouthfeel, smell *etc*. can be requested as feedback from the user. This feedback, in turn, can be cycled from the communication module through the hub and back to the machine learning module to improve upon the model.

Recent progress on the front of multi-instrument integration in autonomous science has been made by Aspuru-Guzik and coworkers who have developed ChemOS, a software designed for operating autonomous laboratories.[95,96] ChemOS is composed of a set of modules or functions: (i) communication, (ii) databases, (iii) robotics, (iv) characterization, (v) learning, and (vi) analysis. The communication module enables interaction between ChemOS and researchers, while the learning module is designed to propose parameters for new experiments based on previous results. The module provides real-time feedback to suggest more informative experiments to efficiently investigate an application space. ChemOS has been applied to a diverse set of experiments, demonstrating the versatility of the software to solve problems in disparate fields.[95,96] Aspuru-Guzik and coworkers showed that ChemOS can map "color space" by experimenting

with mixing dye solutions, learn “Tequila Sunrise space” to optimize cocktail taste based on feedback from researchers, and calibrate a robotic sampling sequence for direct-inject HPLC analysis. The flexibility of this modular framework for directing autonomous science suggests that it could be applied in a variety of research contexts.

Modular architectures to achieve the autonomous laboratory allow the integration of new experimental probes and models in a dynamic fashion. This is particularly powerful in the regime of “little data”, where individual measurements are costly or time-consuming. In scenarios where database information can be leveraged, the modular framework can enable model generation from previous experiments with continuous optimization afforded by dynamic sampling methods. Considering this, it is critical to note that for many chemical questions there is little existing data to build from, and consequently, machine learning methods can be limited in scope and application. The subsequent section will discuss existing methods in machine learning that have been leveraged to solve similar problems in image analysis, with emphasis on generative adversarial networks (GAN) and an illustrative linear analog.

18.3 Generative Adversarial Methods for Data-limited Training

The preceding section highlights the benefits of integrating pre-trained machine learning tools to aid in optimally reducing the number of samples required to inform a chemical decision and support autonomous design of experiments. In contrast, the central focus of this section moves back one level in the pipeline, focusing on methods to optimally train machine learning and conventional chemical classification tools for these tasks and others when confronted with limited or expensive training data. As one example, the dynamic sampling method described in Section 18.2.5 used Raman spectral analysis to determine composition at each spatial location in an image. The algorithm used LDA for dimension reduction and SVM for identification of classification boundaries, both of which required relatively large training sets to optimize. Similarly, the large number of

parameters available for optimization in an artificial neural network (ANN) generally cannot be stably trained with “small data” inputs, exacerbating further the challenges in leveraging machine learning tools in data-sparse scenarios.

Generative adversarial strategies have had great success in addressing many of the most insidious training challenges in data-limited scenarios. Adversarial approaches support the generation of additional training data that are ideally randomly produced from a similar underlying probability density function (pdf) as the raw inputs. If the data were truly from an identical pdf, they would be indistinguishable from the results obtained from additional experimental measurements. In practice, the utility of the generated inputs hinges on the degree to which the small data inputs capture the underlying features of the pdf and how well those can be extracted algorithmically. The goals of this section are threefold: i) review previous successes using adversarial attacks and general adversarial networks in nonlinear problems (*e.g.*, image and speech recognition) as a foundation for extension to chemical applications, ii) improve the understanding of operations “under the hood” in adversarial attacks by devising analogous effects in more intuitive models based on simpler linear spectral analysis, and iii) lay the foundation for integration of generative adversarial attacks in linear spectroscopic analysis to improve chemical classifiers.

18.3.1 Review of Generative Adversarial Networks in Image Recognition Applications

ANNs are designed for optimal performance when trained with large input databases. The numerous weights within each hidden layer in an ANN collectively comprise a large number of free adjustable parameters, which in turn generally requires large volumes of training data to reliably train. In the absence of sufficient training data, the disparity between dimensionality in parameters and inputs generally results in poor statistical confidence in the assignment of an ANN output from undertraining. In brief, limited data sizes do not provide sufficient constraints to prevent underdetermination in the neural network architecture. In such situations, neural networks trained with limited training data can be prone to erroneous results and/or low confidence in classification of testing data.

Even when training with relatively large data sets, residual noise contributions manifested within the ANN weights invariably remain. In such cases, dimension reduction followed by classification is susceptible to hidden “traps” in classification. The susceptibility of machine learning methods (*e.g.*, convolutional neural networks) to spurious changes in input data has been reported in multitude.[97–101]

Generative adversarial strategies work by first identifying potential weak points in an ANN through targeted attacks, and then fortifying the ANN by training it to recognize these attacks. This process is iterated, with the generated “attack” data serving to further refine and train the ANN as a surrogate for large volumes of genuine experimental input data. This iterative two-step process is arguably most well established in neural networks for image recognition, the first step of which is identifying weak points by producing attacks designed to induce image misclassification.

Prior to considering attack strategies in machine learning methods, it is useful to consider how human perception can be influenced by the way image data are portrayed to examine the interplay between a presented set of data and subsequent interpretation. Figure 18.10 shows an example of an optical illusion (Adelson's checker-shadow), designed specifically to attack human neural networks.[102,103] In Adelson's shadow, a checkerboard pattern is overlaid with a cylinder that appears to cast a shadow across the checkerboard. The shadow itself is used to selectively darken several squares in the checkerboard; inspection of the left image yields a perception that the squares A and B are of distinctly different brightness, when in reality A is precisely the same shade as square B. Optical illusions demonstrate how a designed pattern can attack decisions derived from a human neural network; subtle perturbations to the input image can result in interpretation of the image with incorrect conclusions.

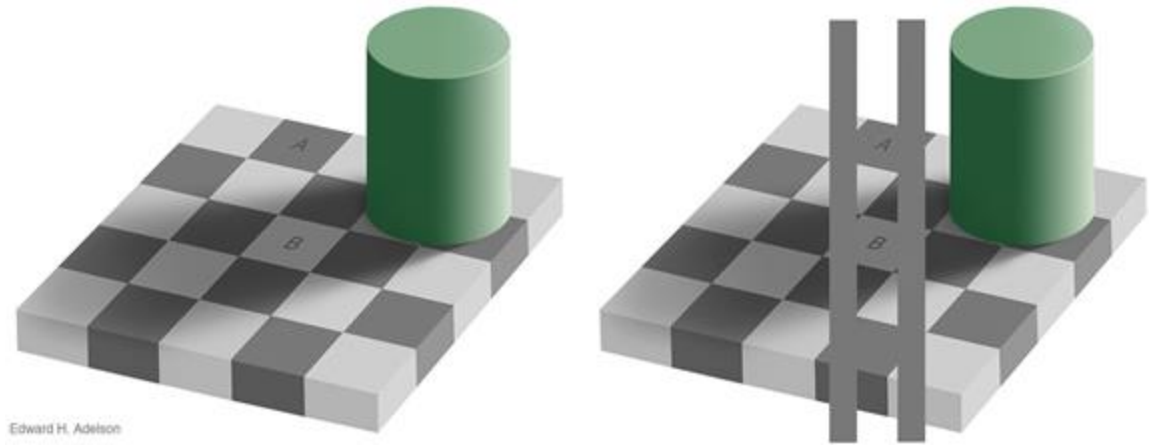


Figure 18.10 An example of visual illusion: Adelson's checker-shadow. In the left image, area B looks brighter than area A. In the right image, two gray bars were added as a reference revealing that area A and area B have the same brightness. Reproduced with permission from E. H. Adelson, In *On seeing stuff: The perception of materials by humans and machines*, Conference on Human Vision and Electronic Imaging VI, San Jose, Ca, Jan 22–25; Spie-Int Soc Optical Engineering: San Jose, Ca, 2001; pp. 1–12.

Adversarial attacks operate in a qualitatively similar manner, but with dramatically different quantitative outcomes in selecting the optimal attacks. Specifically, residual noise within the neural network weightings arising from finite training dataset sizes often serves as a "handle" for launching attacks in neural network classifiers. As such, the malicious perturbation is often engineered to be challenging to identify by naïve human inspection, but with profound impacts on classification by a machine learning algorithm.[104]Figure 18.11 demonstrates an example of adversarial attack in handwriting recognition, in which the effect of the attack can be easily rationalized in hindsight.[105] Changing a few spots in the handwriting input can cause the number 1 to be identified as the number 4. Close examination of this example reveals that the added dots to the image could, in fact, be connected with lines to form a shape of a 4 – the neural network here is weighting those particular pixels with enough importance to outweigh the lack of true connections, generating the full shape of a 4. However, as the dimensionality of the image increases, retrospective rationalization of the perturbations can be increasingly challenging to intuit.

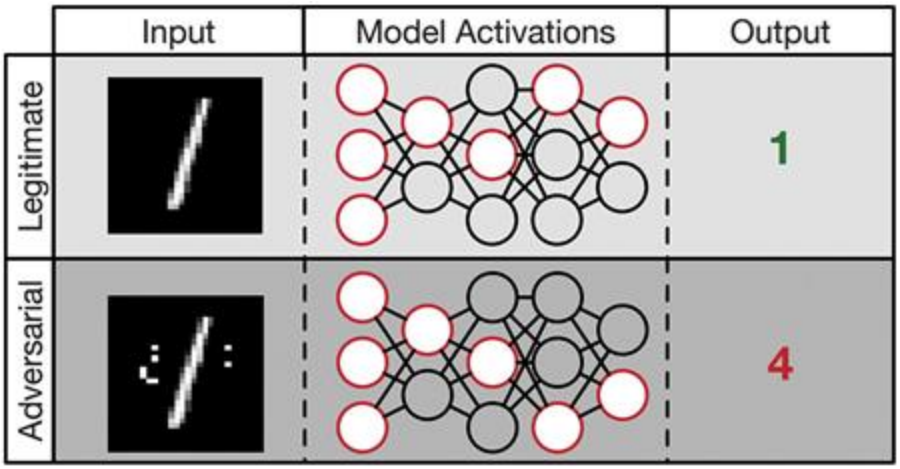


Figure 18.11 Adversarial attack on handwriting recognition. An adversarial sample (bottom row) is produced by slightly altering a legitimate sample (top row) in a way that forces the model to make a wrong prediction, whereas a human would likely correctly classify the sample. Reproduced with permission from N. Papernot, P. McDaniel and I. Goodfellow, Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* 2016.

When discussing adversarial attacks, it is useful to distinguish between attacks on models where the architecture and weights of the trained classifier are known prior to the design of an attack *versus* cases in which they are unknown. Attacks on networks where the attacker has full knowledge of the classification structure and weights are known as white box attacks, in which cases the optimal perturbations can typically be calculated analytically to efficiently fool classifiers for given inputs and targets. The strategies for attack can vary widely, targeting different image aspects. In one of the most common implementations, back-propagation of the neural network allows determination of the steepest gradient toward a classification boundary, corresponding to the smallest perturbation required to induce a given misclassification. In contrast, attacks, where only the classification outcome is accessible to the attacker, are known as black box attacks – such methods rely on optimization approaches to generate a perturbation to the input data to produce the desired classification outcome.

Illustration of a white box attack is shown in Figure 18.12, in which an optimized, subtle perturbation added to an image of a panda bear resulted in the GoogLeNet image classifier confidently misclassifying the image as a gibbon.[106,107] In this attack, the elements of the perturbation are equal to the

sign of the elements of the gradient of the cost function with respect to the input, a method dubbed the fast gradient sign method. Intuitively this approach can be interpreted as identifying the direction of highest classification uncertainty and adding the minimal push to the input data *x*. The scaling factor $\in$=0.007 is the magnitude of the smallest bit of an 8-bit image encoder after conversion to real numbers. Notably, this approach does not point the classifier to a specific outcome, but rather utilizes the full knowledge of the training cost function to minimally push the classifier to an incorrect conclusion.

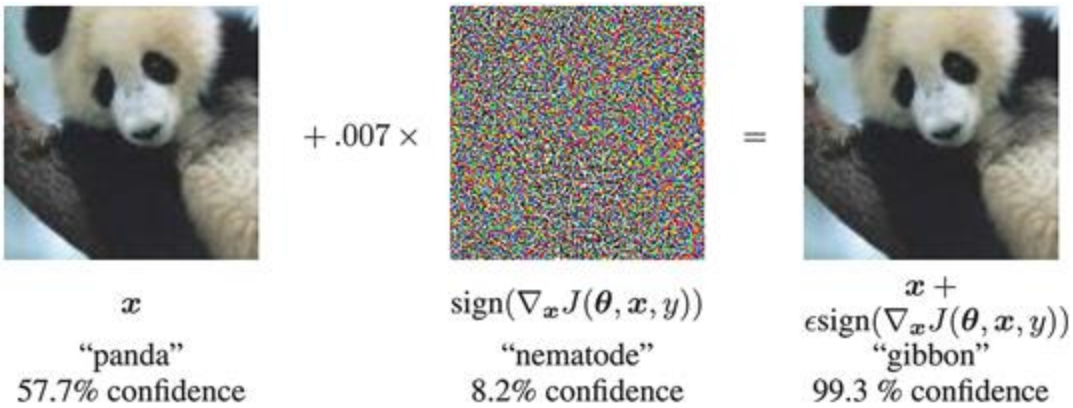


Figure 18.12 A demonstration of fast adversarial example generation applied to GoogLeNet on ImageNet. By adding an imperceptibly small perturbation, GoogLeNet's classification of the image is significantly altered. Reproduced with permission from I. J. Goodfellow, J. Shlens, and C. Szegedy, Explaining and Harnessing Adversarial Examples. *arXiv e-prints* (2014).

In many cases, the analytical solution for optimizing an attack pattern is not known, for which the outcomes of the neural network for given inputs alone serve to inform the optimization of an attack. Such black box attacks add small perturbations to input data, test the classifier, and then iteratively optimize perturbations numerically to achieve a classification confidence subject to a cost function. Kevin Eykholt and coauthors illustrate an example of a black-box attack, in which perturbations were determined to induce misinterpretation of traffic sign images by image recognition algorithms.[108] In this black-box attack, they were able to induce misclassification of a stop sign as a 45 mph speed limit sign through perturbations difficult to rationalize upon human inspection. Without

knowing the neural network behind the traffic sign classification, the attack pattern sampled from a distribution that modeled physical dynamics (in this case, varying distances and angles), and used a mask to project computed perturbations to a shape that resembled graffiti. This attack fooled two convolutional neural networks, LISA and GTSRB-CNN with over 85% confidence.

Despite the success of these adversarial attack strategies in ANNs, the inherent nonlinear nature of many ANN architectures complicates intuitive rationalization of both the intrinsic susceptibilities to attack and origins of the optimized attack patterns. The greatest strengths in ANNs arguably lie in their abilities to exploit underlying patterns through coupled but highly nonlinear transformations. However, the intrinsic nonlinearities associated with the pattern-matching in ANNs greatly complicate meaningful mapping onto more intuitive and tractable linear frameworks more commonly used in spectral analyses. It is challenging enough to intuit the strengths and weaknesses of an ANN in general, while it is more challenging to interpret a numerically generated adversarial attack thereof.

18.3.2 Illustrative Linear Example of Adversarial Attacks: Adversarial Spectroscopy

Many of these training challenges in ANNs under data-limited conditions have direct analogs in classical, linear dimension reduction methods, providing a significantly more intuitive and tractable context for interpreting the origins of these potential numerical instabilities. When used for classification of images, sounds, or spectra, neural networks can be viewed as dimension reduction methods, in which high-dimensional objects in image or spectral space result in low-dimensional classified outputs. As such, linear methods for dimension reduction such as principal component analysis (PCA) and LDA can potentially serve as simple ANN surrogates for interpreting the design and execution of adversarial attacks, which is the primary subject of this section.

As an illustrative example, it is useful to compare dimension reduction/classification by ANNs with conventional LDA for spectral classification. In LDA, supervised inputs are used to identify a new lower-dimensional space designed to maximize the resolution between the

classified inputs. Maximizing the resolution is mathematically analogous to maximizing the Fischer linear discriminant. For a given projection in spectral space $\boldsymbol{w}$, the corresponding value of the Fisher discriminant J is given by the eqn (18.7), in which the matrices $\boldsymbol{S}_w$ and $\boldsymbol{S}_B$ correspond to the within-class and between-class variance/covariance matrices, respectively.

$$J = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w}} \tag{18.7}$$

Mathematically, the direction in spectral space that maximizes the value for J can be found by solving the eigenvector/eigenvalue problem in eqn (18.8).

$$Jw = S_w{}^{-1} S_B w \tag{18.8}$$

Upon extension to multiple classes, the vector $\boldsymbol{w}$ can be replaced by a matrix $\boldsymbol{W}$ to produce sets of eigenvectors corresponding to the reduced dimensional coordinates generated by LDA.

The matrix inversion step in eqn (18.8) provides a source of potential instability in data-limited applications, directly analogous to data-limited training of ANNs. Specifically, the $\boldsymbol{S}_w$ matrix is only nonsingular and formally invertible when the number of training spectra is greater than the length of a given spectrum. When training data are limited and this criterion is not met, direct application of LDA with numerical methods to estimate the matrix inversion operation results in poorly constrained selection of dimensions highly prone to “overfitting”, in which noise contributions within the data contribute significantly or dominate the selection of dimensions to maximally resolve the classified data. Even when the number of training data exceed the dimensionality of the spectral space, the matrix $\boldsymbol{S}_w$ is generally still not sufficiently overdetermined to remove significant contributions from noise in the loading vectors used for dimension reduction. As such, spurious noise contributions can potentially result in misclassification or classification with low statistical confidence. In LDA, this noise-susceptibility is apparent upon inspection of the “loading plots”, which correspond to the directions in spectral space that each spectrum is

projected onto in order to generate the corresponding position in LDA-space. Representative loading plots shown for a simulated training set of Raman spectra just large enough (1424 spectra) to satisfy the nonsingularity requirement are shown in Figure 18.13, clearly indicating significant contributions of residual noise to the classifiers designed to maximize the resolution between classes. As expected, increasing the number of simulated Raman spectra in the training set by about two orders of magnitude (160884 spectra) significantly reduced the noise in the LDA loading plots.

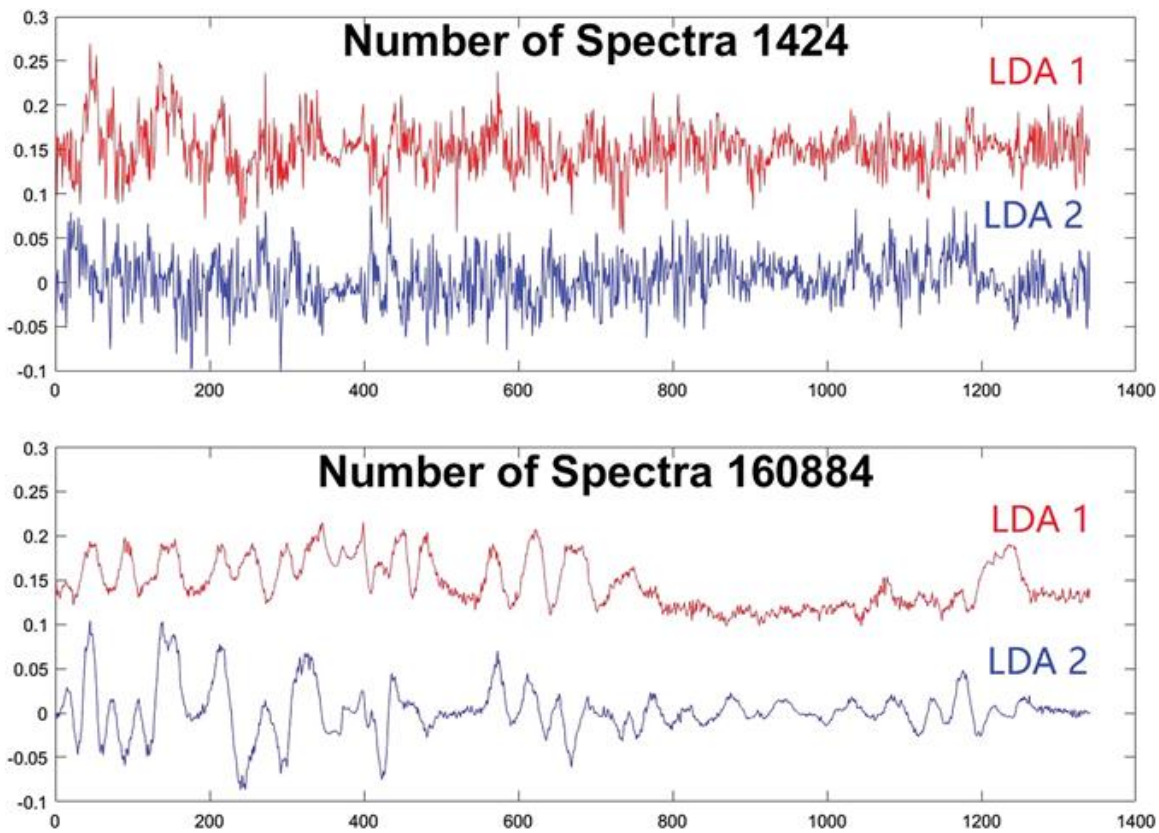


Figure 18.13 Comparison of LDA loading plots with different amounts of simulated training spectra. An original set of 267 experimental Raman spectra are supplemented with simulated spectra from the same statistical distribution. As expected, noise is significantly reduced when performing LDA with a greater number of training spectra.

The benefits of generative adversarial approaches to enable chemical analyses with limited data hinge on the success of the adversarial attacks in identifying points of vulnerability in classifiers. In this illustration of

adversarial spectroscopy, a perturbation to the initial spectrum $\boldsymbol{\delta}$ is added to the initial sample spectrum $\boldsymbol{x_s}$ to produce the perturbed spectrum $\boldsymbol{x}'=\boldsymbol{x}_s+\boldsymbol{\delta}$. The cost function to optimize the attack is given in eqn (18.9) as the sum of two terms. The first term, $\|\boldsymbol{D}(\boldsymbol{x}_s+\boldsymbol{\delta}-\boldsymbol{\mu}_t)\|^2$, minimizes the squared deviation of the perturbed spectra $\boldsymbol{x}_s+\boldsymbol{\delta}$ and the target spectrum $\boldsymbol{\mu}_\mathrm{t}$ in LDA space. The transform matrix $\boldsymbol{D}$ maps the difference in spectral space $\boldsymbol{x}_s+\boldsymbol{\delta}-\boldsymbol{\mu}_t$ to LDA space, driving the input data to the location of the target class mean and inducing misclassification. The second term minimizes the overall magnitude of the perturbation $\|\boldsymbol{\delta}\|^2$, constraining the overall perturbation to a small magnitude and reducing the visual difference between the initial and attacked spectrum. In other words, the first term fools the classifier, while the second term fools the human. The scaling parameter β allows for empirical adjustment of the importance of subtlety; β=1 was used for this study.

$$\hat{\boldsymbol{\delta}} = \arg\min_{\boldsymbol{\delta}} \left[\|\boldsymbol{D}(\boldsymbol{x}_s + \boldsymbol{\delta} - \boldsymbol{\mu}_t)\|^2 + \beta \|\boldsymbol{\delta}\|^2 \right] \tag{18.9}$$

The large apparent contributions of noise to the loading plots in LDA-based dimension reduction suggests that a significant contribution to the chemical classification includes contributions from spurious noise in the training data, creating potential susceptibilities for adversarial attacks. Incorporation of spurious noise within the dimension reduction operation in LDA is directly analogous to “overfitting” in ANNs, in which classification becomes overly dependent upon random features (including spurious noise) within the training data that serve as toeholds for adversarial perturbations. While in principle, this intrinsic susceptibility to attack could be minimized through increases in the size of the training set, practical limitations can complicate such approaches, particularly when data are relatively expensive or time-consuming to obtain.

An example in Figure 18.14 illustrates the vulnerability of a Raman spectral classification informed by LDA. An initial spectrum (top trace in **A**) clearly corresponds to Class I (middle right of **B**), both by its visual similarity to the Class I mean spectrum and by its corresponding position in LDA-space. Addition of a perturbation to the spectrum according to the

cost function in eqn (18.9) results in minor visual changes in the spectrum (middle trace in **A**), but quite dramatic reclassification to Class II (bottom of **B**), with >95% confidence based on the position in LDA-space.

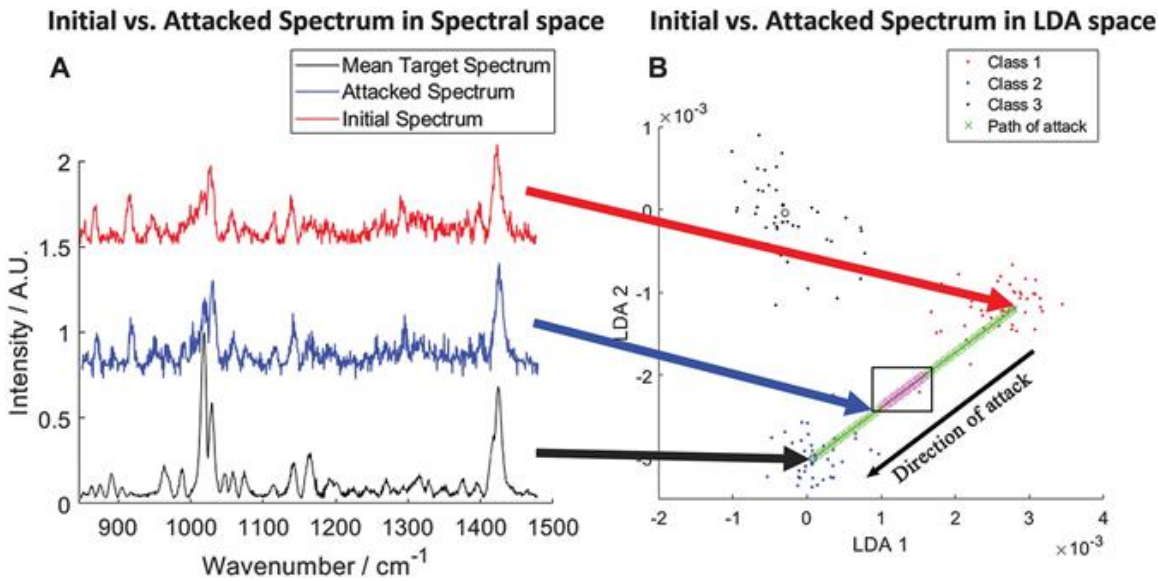


Figure 18.14 Illustration of an adversarial spectroscopic attack (**A**) An initially unperturbed spectrum (top) is attacked, generating the attacked spectrum (middle). The attacked trace is then classified as the target spectrum (bottom). Note the attacked spectrum (middle) does not bear a significant spectral resemblance to the target class (bottom). The classifier is fooled while human inspection will not detect the perturbation. (**B**) Demonstration of an adversarial attack in LDA space. The attack spectrum is moved from class 1 (middle right) to class 2 (bottom left) by adding small amounts of noise to the original spectrum. The boxed region denotes the area of greatest uncertainty for classification.

This susceptibility to adversarial perturbation is an intrinsic consequence of the dimension reduction by projection onto the LDA coordinates; there is an infinite set of possible spectra that correspond to the same position in LDA-space. Correspondingly, innumerable perturbations to any initial spectrum can shift its position to the target location in LDA-space. The definition of the cost function selects the one spectrum among this set that corresponds to the optimal perturbation (in this case, the perturbation with the smallest squared magnitude). For this cost function, the presence of residual noise within the LDA-coordinates preferentially selects many small "nudges" over more obvious large perturbations at a small number of

wavelengths. Most significantly, the perturbations generally do NOT result in spectra with peaks similar to those expected in the target class. The optimized perturbations shown in panel A of Figure 18.14 retained major spectral features of the original class while misclassifying with ≥95% confidence.

The vulnerability to adversarial attack inherent in many spectral classifiers illustrated in this section also presents opportunities for exploiting adversarial strategies to improve chemical classification algorithms. Potential benefits of defensive strategies for improving chemical classification algorithms are the primary subject of the next section.

18.3.3 Illustrative Example of Data Generation: Generative Adversarial Linear Analysis

The primary utility of adversarial spectroscopy is arguably in its potential for improving the general reliability of chemical classifiers. Training data for creating classifiers are necessarily finite resources, with any practical data set potentially susceptible to spurious or malignant misclassification. One promising avenue to address this vulnerability is through generative adversarial strategies in spectrochemical analysis, inspired by parallel successful implementation of GANs in image processing.[109]

A GAN is specifically designed to identify and shore-up potential weak-points in a given classifier through the generation of "fake" inputs (*e.g.*, images), followed by discrimination between real and fake inputs. GANs are typically deep neural networks comprised of two key components: a generator and a discriminator. Random arrays serve as inputs to the generator, which produces an output of fake inputs (*e.g.*, images) for the discriminator. Knowledge of the discriminator is used in training the generator to produce fake inputs through adversarial attacks that will be classified into target categories. Next, a new discriminator is produced capable of additionally distinguishing between real and fake inputs. This new discriminator is then used to train a new generator, and this adversarial back-and-forth is iterated until some convergence criterion is met. Through this process, the final discriminator exhibits substantial improvements against adversarial or spurious perturbations.

The success of GANs can be interpreted within the context of signal averaging. If we could devise a perfect generator, the output would produce spectra or images statistically identical to genuine measurements, which in turn, could be used for noise suppression in the discriminator as illustrated in Figure 18.15. Stated differently, the perfect generator captures all the complex and generally nonlinear relationships associated with the underlying pdf describing the experimental measurements. Just as a Gaussian pdf describing the generation of many thousands of measured inputs can be described by just two parameters (mean and variance), the generated inputs to a discriminating neural network can be fundamentally described by lower-dimensional latent-space parameters. In neural networks, the functional form for the pdf and the nature of the underlying latent parameter-space are generally not known or even analytically derivable, but rather are learned through training of the generator.

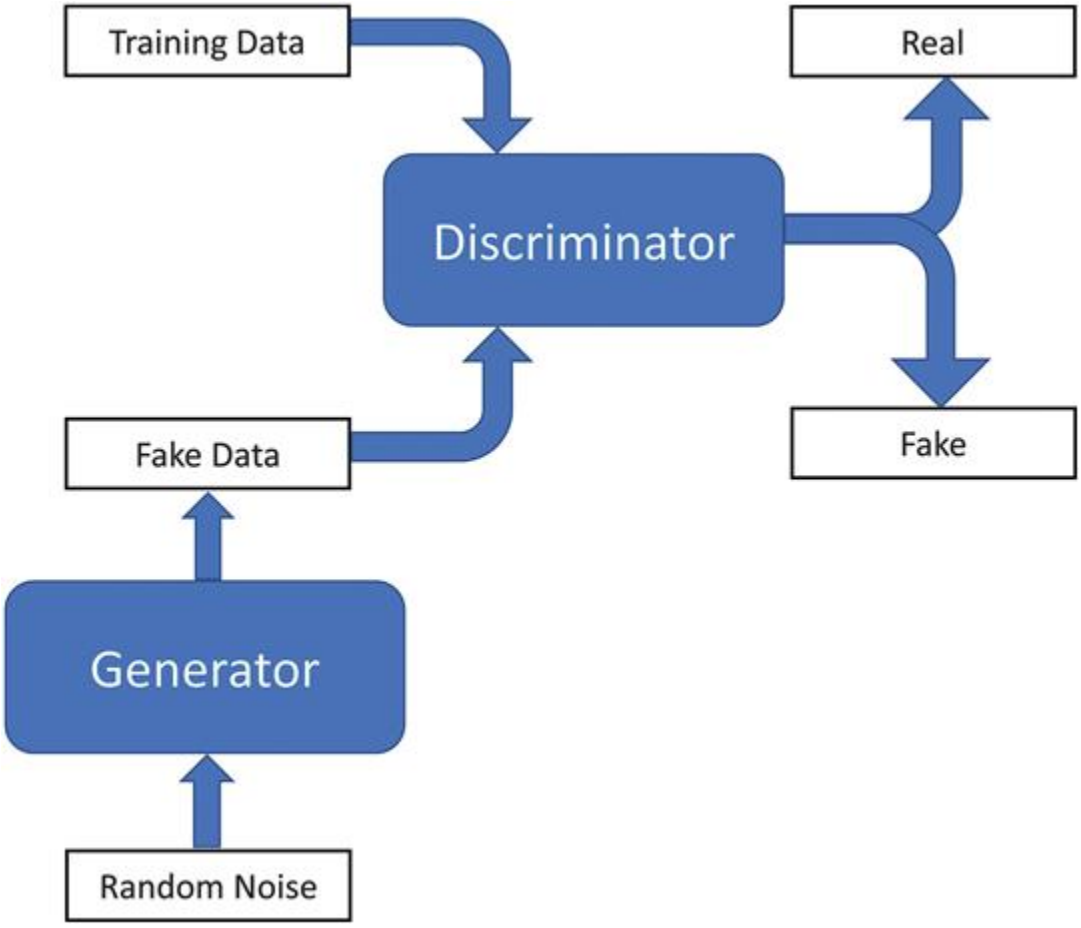


Figure 18.15 Workflow of a GAN. The generator is trained to produce fake data that will "fool" the discriminator while the discriminator is trained to distinguish fake data from real data. By forcing the generator and discriminator to compete, the discriminator will become more robust.

The authors expect adversarial spectroscopic approaches to ultimately result in analogous improvements in chemical classification upon integration into future generative adversarial architectures. A conceptual overview of how such an approach could be implemented is envisioned in Figure 18.16. Using the results described in Figure 18.14 as an example, measured spectral data first undergo dimension reduction for noise suppression and to aid in the definition of classification boundaries (indicated by the ellipses in the left-most plot). Randomly generated initial spectra (*e.g.*, Gaussian distributed random values) can serve as starting points for launching adversarial attacks targeting one of the chemical classes. Projection of these random initial spectra onto the reduced dimensional space is illustrated by the large ellipse in the middle plot. Using the cost function given in eqn (18.9), the attack minimizes the distance to the target in the reduced dimensional space, while simultaneously reducing the overall magnitude of the perturbation in the higher-dimensional spectral space. Next, a new set of loading plots for dimension reduction is determined that includes discrimination between the initial genuine data and the fake spectra generated by performing attacks on random initial inputs. Iteration of this procedure has the potential to suppress noise contributions within the loading plots through mechanisms qualitatively similar to those arising from signal averaging. Furthermore, no explicit assumptions are necessary regarding the underlying pdfs describing the signal or noise in the measurements.

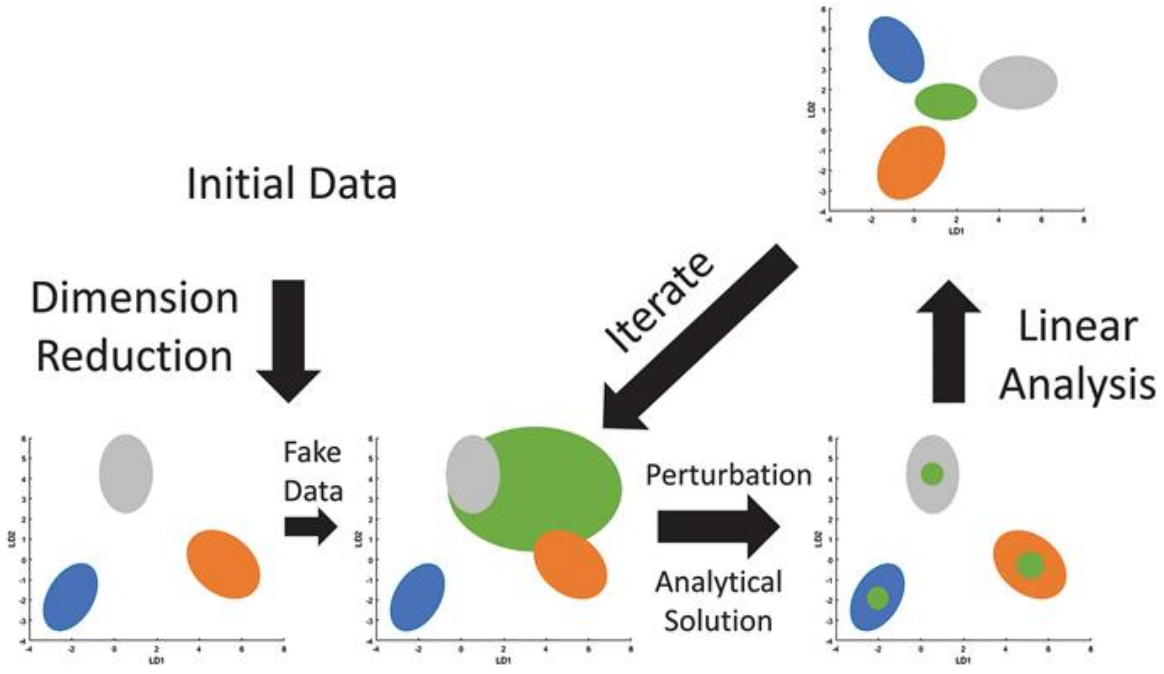


Figure 18.16 The workflow of generative adversarial approaches in spectral classification involves the following conceptual steps: (i) initial data are analyzed by dimension reduction, (ii) "fake" data are generated in spectral space using random inputs and projected into this lower-dimensional space, (iii) each of the fake spectra is perturbed by an adversarial attack to induce classification into one of the target classes, (iv) a new dimension reduction operation is performed to separate the real and fake spectra. Steps ii–iv are then iterated to a desired level of convergence.

18.4 Conclusion

Recent advances in machine learning have opened exciting avenues for leveraging big data tools in small data applications. Individual chemical measurements can be time-consuming and practically difficult, or otherwise costly. To meet these challenges, several strategies were explored for minimizing the total number of measurements required to inform decision-making. In chemical imaging, sparse sampling strategies were shown to leverage intrinsic image compressibility to reliably "inpaint" composition within unsampled pixel locations. Coupling sparse sampling methods with models for predicting the most informative experiments enables dynamic sampling, in which the previous set of measurements dynamically informs

the next sampling locations. Finally, recent advances in GANs for neural network training in data-limited scenarios were mapped to analogous operations in linear classifiers, providing a route for understanding the mechanism of operation underpinning GANs and enabling extension of those concepts to more traditional linear algorithms for chemical classification.

Acknowledgements

The authors gratefully acknowledge funding for the present work from the National Science Foundation through an NSF-GOALI award (No. CHE-1710475) and an NSF-CIF award (No. 1763896).

References

1. X. W. Chen and X. T. Lin, *IEEE Access*, 2014, **2**, 514–525.
2. Q. C. Zhang, L. T. Yang, Z. K. Chen and P. Li, *Inform. Fusion*, 2018, **42**, 146–157.
3. L. Xu, J. S. J. Ren, C. Liu and J. Y. Jia, *Advances in Neural Information Processing Systems 27 (Nips 2014)*, **vol. 27**, 2014, p. 9.
4. F. Seide, G. Li, D. Yu and A. Int Speech Commun, *12th Annual Conference of the International Speech Communication Association 2011 (Interspeech 2011)*, **vol. 1–5**, 2011, p. 444.
5. M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald and E. Muharemagic, *J. Big Data*, 2015, **2**, 1.
6. A. Khademhosseini, J. Yeh, G. Eng, J. Karp, H. Kaji, J. Borenstein, O. C. Farokhzad and R. Langer, *Lab Chip*, 2005, **5**, 1380–1386.
7. Y. Gao, P. Li and D. Pappas, *Biomed. Microdevices*, 2013, **15**, 907–915.
8. E. S. Park, A. C. Brown, M. A. DiFeo, T. H. Barker and H. Lu, *Lab Chip*, 2010, **10**, 571–580.
9. J. Kim, D. Taylor, N. Agrawal, H. Wang, H. Kim, A. Han, K. Rege and A. Jayaraman, *Lab Chip*, 2012, **12**, 1813–1822.

10. N. N. Ye, J. H. Qin, W. W. Shi, X. Liu and B. C. Lin, *Lab Chip*, 2007, **7**, 1696–1704.
11. A. Aharoni, K. Thieme, C. P. C. Chiu, S. Buchini, L. L. Lairson, H. M. Chen, N. C. J. Strynadka, W. W. Wakarchuk and S. G. Withers, *Nat. Methods*, 2006, **3**, 609–614.
12. J. C. Baret, O. J. Miller, V. Taly, M. Ryckelynck, A. El-Harrak, L. Frenz, C. Rick, M. L. Samuels, J. B. Hutchison, J. J. Agresti, D. R. Link, D. A. Weitz and A. D. Griffiths, *Lab Chip*, 2009, **9**, 1850–1858.
13. J. J. Agresti, E. Antipov, A. R. Abate, K. Ahn, A. C. Rowat, J. C. Baret, M. Marquez, A. M. Klibanov, A. D. Griffiths and D. A. Weitz, *Proc. Natl. Acad. Sci. U. S. A.*, 2010, **107**, 4004–4009.
14. A. Fallah-Araghi, J. C. Baret, M. Ryckelynck and A. D. Griffiths, *Lab Chip*, 2012, **12**, 882–891.
15. P. Y. Colin, B. Kintses, F. Gielen, C. M. Miton, G. Fischer, M. F. Mohamed, M. Hyvonen, D. P. Morgavi, D. B. Janssen and F. Hollfelder, *Nat. Commun.*, 2015, **6**, 1–12.
16. J. Draper, A. J. Lloyd, R. Goodacre and M. Beckmann, *Metabolomics*, 2013, **9**, S4–S29.
17. T. Fuhrer, D. Heer, B. Begemann and N. Zamboni, *Anal. Chem.*, 2011, **83**, 7074–7080.
18. J. M. Buescher, S. Moco, U. Sauer and N. Zamboni, *Anal. Chem.*, 2010, **82**, 4403–4412.
19. W. Jian, M. V. Romm, R. W. Edom, V. P. Miller, W. A. LaMarr and N. Weng, *Anal. Chem.*, 2011, **83**, 8259–8266.
20. G. Madalinski, E. Godat, S. Alves, D. Lesage, E. Genin, P. Levi, J. Labarre, J.-C. Tabet, E. Ezan and C. Junot, *Anal. Chem.*, 2008, **80**, 3291–3303.
21. R. D. Smith, G. A. Anderson, M. S. Lipton, L. Pasa-Tolic, Y. F. Shen, T. P. Conrads, T. D. Veenstra and H. R. Udseth, *Proteomics*, 2002, **2**, 513–523.
22. H. Weisser, S. Nahnsen, J. Grossmann, L. Nilse, A. Quandt, H. Brauer, M. Sturm, E. Kenar, O. Kohlbacher, R. Aebersold and L. Malinstroemt, *J. Proteome Res.*, 2013, **12**, 1628–1644.
23. E. S. Baker, E. A. Livesay, D. J. Orton, R. J. Moore, W. F. Danielson, IIID. C. PriorY. M. IbrahimB. L. LaMarcheA. M. MayampurathA. A.

SchepmoesD. F. HopkinsK. TangR. D. Smith and M. E. Belov, *J. Proteome Res.*, 2010, **9**, 997–1006.
24. H.-K. Min, S.-W. Hyung, J.-W. Shin, H.-S. Nam, S.-H. Ahn, H. J. Jung and S.-W. Lee, *Electrophoresis*, 2007, **28**, 1012–1021.
25. L. Pasa-Tolic, P. K. Jensen, G. A. Anderson, M. S. Lipton, K. K. Peden, S. Martinovic, N. Tolic, J. E. Bruce and R. D. Smith, *J. Am. Chem. Soc.*, 1999, **121**, 7949–7950.
26. N. Castle, D. Printzenhoff, S. Zellmer, B. Antonio, A. Wickenden and C. Silvia, *Comb. Chem. High Throughput Screening*, 2009, **12**, 107–122.
27. C. Farre, A. Haythornthwaite, C. Haarmann, S. Stoelzle, M. Kreir, M. George, A. Brueggemann and N. Fertig, *Comb. Chem. High Throughput Screening*, 2009, **12**, 24–37.
28. W. M. Tang, J. S. Kang, X. Y. Wu, D. Rampe, L. Wang, H. Shen, Z. Y. Li, D. Dunnington and T. Garyantes, *J. Biomol. Screening*, 2001, **6**, 325–331.
29. E. M. Chan, C. Xu, A. W. Mao, G. Han, J. S. Owen, B. E. Cohen and D. J. Milliron, *Nano Lett.*, 2010, **10**, 1874–1885.
30. D. Perera, J. W. Tucker, S. Brahmbhatt, C. J. Helal, A. Chong, W. Farrell, P. Richardson and N. W. Sach, *Science*, 2018, **359**, 429.
31. M. Wleklinski, B. P. Loren, C. R. Ferreira, Z. Jaman, L. Avramova, T. J. P. Sobreira, D. H. Thompson and R. G. Cooks, *Chem. Sci.*, 2018, **9**, 1647–1653.
32. S.-D. Huang, C. Shang, P.-L. Kang and Z.-P. Liu, *Chem. Sci.*, 2018, **9**, 8644–8655.
33. B. Maryasin, P. Marquetand and N. Maulide, *Angew. Chem., Int. Ed.*, 2018, **57**, 6978–6980.
34. S. Lotfi and J. Brgoch, *Comput. Mater. Sci.*, 2019, **158**, 124–129.
35. L. Wilbraham, R. S. Sprick, K. E. Jelfs and M. A. Zwijnenburg, *Chem. Sci.*, 2019, DOI: 10.1039/C8SC05710A.
36. M. Tsubaki and T. Mizoguchi, *J Phys Chem Lett*, 2018, **9**, 5733–5741.
37. J. Staker, K. Marshall, R. Abel and C. M. McQuaw, *J. Chem. Inf. Model.*, 2019, **59**, 1017–1029.
38. Y. Zhuo, A. Mansouri Tehrani and J. Brgoch, *J. Phys. Chem. Lett.*, 2018, **9**, 1668–1673.

39. C. W. Coley, W. Jin, L. Rogers, T. F. Jamison, T. S. Jaakkola, W. H. Green, R. Barzilay and K. F. Jensen, *Chem. Sci.*, 2019, **10**, 370–377.
40. L. Turcani, R. L. Greenaway and K. E. Jelfs, *Chem. Mater.*, 2018, **31**, 714–727.
41. J. M. Di Nicola, T. Bond, M. Bowers, L. Chang, M. Hermann, R. House, T. Lewis, K. Manes, G. Mennerat, B. MacGowan, R. Negres, B. Olejniczak, C. Orth, T. Parham, S. Rana, B. Raymond, M. Rever, S. Schrauth, M. Shaw, M. Spaeth, B. Van Wonterghem, W. Williams, C. Widmayer, S. Yang, P. Whitman and P. Wegner, *Nucl. Fusion*, 2019, **59**, 12.
42. C. Haynam, P. Wegner, J. Auerbach, M. Bowers, S. Dixit, G. Erbert, G. Heestand, M. Henesian, M. Hermann and K. Jancaitis, *Appl. opt.*, 2007, **46**, 3276–3303.
43. E. I. Moses and C. R. Wuest, *Fusion Sci. Technol.*, 2005, **47**, 314–322.
44. G. MacBeath and S. L. Schreiber, *Science*, 2000, **289**, 1760–1763.
45. E. Brouzes, M. Medkova, N. Savenelli, D. Marran, M. Twardowski, J. B. Hutchison, J. M. Rothberg, D. R. Link, N. Perrimon and M. L. Samuels, *Proc. Natl. Acad. Sci. U. S. A.*, 2009, **106**, 14195–14200.
46. D. C. Pregibon, M. Toner and P. S. Doyle, *Science*, 2007, **315**, 1393–1396.
47. J. G. Caporaso, C. L. Lauber, W. A. Walters, D. Berg-Lyons, J. Huntley, N. Fierer, S. M. Owens, J. Betley, L. Fraser, M. Bauer, N. Gormley, J. A. Gilbert, G. Smith and R. Knight, *Isme Journal*, 2012, **6**, 1621–1624.
48. M. M. Young, N. Tang, J. C. Hempel, C. M. Oshiro, E. W. Taylor, I. D. Kuntz, B. W. Gibson and G. Dollinger, *Proc. Natl. Acad. Sci. U. S. A.*, 2000, **97**, 5802–5806.
49. A. M. L. Liekens, J. De Knijf, W. Daelemans, B. Goethals, P. De Rijk and J. Del-Favero, *Genome Biol.*, 2011, **12**, 12.
50. R. D. King, K. E. Whelan, F. M. Jones, P. G. K. Reiser, C. H. Bryant, S. H. Muggleton, D. B. Kell and S. G. Oliver, *Nature*, 2004, **427**, 247–252.
51. R. Lorenz, R. P. Monti, I. R. Violante, C. Anagnostopoulos, A. A. Faisal, G. Montana and R. Leech, *arXiv preprint arXiv:1506.02088*, 2015.

52. S. J. Zhang, Z. T. Song, G. Godaliyadda, D. H. Ye, A. U. Chowdhury, A. Sengupta, G. T. Buzzard, C. A. Bouman and G. J. Simpson, *Anal. Chem.*, 2018, **90**, 4461–4469.
53. Y. Zhang, G. M. D. Godaliyadda, N. Ferrier, E. B. Gulsoy, C. A. Bouman and C. Phatak, *Ultramicroscopy*, 2018, **184**, 90–97.
54. N. M. Scarborough, G. Godaliyadda, D. H. Ye, D. J. Kissick, S. J. Zhang, J. A. Newman, M. J. Sheedlo, A. U. Chowdhury, R. F. Fischetti, C. Das, G. T. Buzzard, C. A. Bouman and G. J. Simpson, *J. Synchrotron Radiat.*, 2017, **24**, 188–195.
55. P. Binev, W. Dahmen, R. DeVore, P. Lamby, D. Savu and R. Sharpley, in *Modeling Nanoscale Imaging in Electron Microscopy*, Springer, 2012, pp. 73–126.
56. J. Greenberg, K. Krishnamurthy and D. Brady, *Opt. Lett.*, 2014, **39**, 111–114.
57. A. C. Geiger, J. A. Newman, S. Sreehari, S. Z. Sullivan, C. A. Bouman and G. J. Simpson, in *High-Speed Biomedical Imaging and Spectroscopy: Toward Big Data Instrumentation and Management II*, International Society for Optics and Photonics, 2017, **vol. 10076**, p. 1007606.
58. S. Z. Sullivan, R. D. Muir, J. A. Newman, M. S. Carlsen, S. Sreehari, C. Doerge, N. J. Begue, R. M. Everly, C. A. Bouman and G. J. Simpson, *Opt. Express*, 2014, **22**, 24224–24234.
59. D. Garcia, *Comput. Stat. Data Anal.*, 2010, **54**, 1167–1178.
60. G. J. Wang, D. Garcia, Y. Liu, R. de Jeu and A. J. Dolman, *Environ. Modell. Softw.*, 2012, **30**, 139–142.
61. Z. Yu, J. B. Thibault, C. A. Bouman, K. D. Sauer and J. Hsieh, in *2008 15th IEEE International Conference on Image Processing*, IEEE, 2008, pp. 2600–2603.
62. G. Godaliyadda, D. H. Ye, M. D. Uchic, M. A. Groeber, G. T. Buzzard and C. A. Bouman, *Electron. Imaging*, 2016, **2016**, 1–8.
63. G. Godaliyadda, D. H. Ye, M. D. Uchic, M. A. Groeber, G. T. Buzzard and C. A. Bouman, *IEEE Trans. Comput. Imaging*, 2018, **4**, 1–16.
64. Y. Zhang, G. Godaliyadda, N. Ferrier, E. B. Gulsoy, C. A. Bouman and C. Phatak, *Electron. Imaging*, 2018, **2018**, 131–131-1316.
65. J. A. Nelder and R. Mead, *Comput. J.*, 1965, **7**, 308–313.

66. M. W. Routh, P. A. Swartz and M. B. Denton, *Anal. Chem.*, 1977, **49**, 1422–1428.
67. H. Winicov, J. Schainbaum, J. Buckley, G. Longino, J. Hill and C. E. Berkoff, *Anal. Chim. Acta*, 1978, **103**, 469–476.
68. G. E. Mieling, R. W. Taylor, L. G. Hargis, J. English and H. L. Pardue, *Anal. Chem.*, 1976, **48**, 1686–1693.
69. K. Kong, C. J. Rowlands, H. Elsheikha and I. Notingher, *Analyst*, 2012, **137**, 4119–4122.
70. C. J. Rowlands, S. Varma, W. Perkins, I. Leach, H. Williams and I. Notingher, *J. Biophotonics*, 2012, **5**, 220–229.
71. C. Cortes and V. Vapnik, *Mach. Learn.*, 1995, **20**, 273–297.
72. C. R. Rao, *J. R. Stat. Soc. B*, 1948, **10**, 159–203.
73. R. D. King, J. Rowland, S. G. Oliver, M. Young, W. Aubrey, E. Byrne, M. Liakata, M. Markham, P. Pir and L. N. Soldatova, *Science*, 2009, **324**, 85–89.
74. K. E. Whelan and R. D. King, *Bmc Bioinformatics*, 2008, **9**, 16.
75. R. F. Murphy, *Nat. Chem. Biol.*, 2011, **7**, 327–330.
76. D. A. Cohn, Z. Ghahramani and M. I. Jordan, *J. Artif. Intell. Res.*, 1996, **4**, 129–145.
77. M.-F. Balcan, A. Beygelzimer and J. Langford, *J. Comput. Syst. Sci.*, 2009, **75**, 78–89.
78. M. Saar-Tsechansky and F. Provost, *Mach. Learn.*, 2004, **54**, 153–178.
79. A. W. Naik, J. D. Kangas, C. J. Langmead and R. F. Murphy, *PLoS One*, 2013, **8**, e83996.
80. A. W. Naik, J. D. Kangas, D. P. Sullivan and R. F. Murphy, *Elife*, 2016, **5**, e10047.
81. L. P. Coelho, A. Ahmed, A. Arnold, J. Kangas, A.-S. Sheikh, E. P. Xing, W. W. Cohen and R. F. Murphy, in *Linking Literature, Information, and Knowledge for Biology*, Springer, 2010, pp. 23–32.
82. L. P. Coelho, J. D. Kangas, A. W. Naik, E. Osuna-Highley, E. Glory-Afshar, M. Fuhrman, R. Simha, P. B. Berget, J. W. Jarvik and R. F. Murphy, *Bioinformatics*, 2013, **29**, 2343–2349.
83. M. V. Boland, M. K. Markey and R. F. Murphy, *Cytometry*, 1998, **33**, 366–375.
84. X. Chen and R. F. Murphy, *J. Biomed. Biotechnol.*, 2005, 87–95.

85. P. Nikolaev, D. Hooper, F. Webber, R. Rao, K. Decker, M. Krein, J. Poleski, R. Barto and B. Maruyama, *Npj Comput. Mater.*, 2016, **2**, 16031.
86. P. Nikolaev, D. Hooper, N. Perea-Lopez, M. Terrones and B. Maruyama, *Acs Nano*, 2014, **8**, 10214–10222.
87. R. Rao, N. Pierce, D. Liptak, D. Hooper, G. Sargent, S. L. Semiatin, S. Curtarolo, A. R. Harutyunyan and B. Maruyama, *Acs Nano*, 2013, **7**, 1100–1107.
88. A. Liaw and M. Wiener, *R News*, 2002, **2**, 18–22.
89. R. Diaz-Uriarte and S. A. de Andres, *BMC Bioinf.*, 2006, **7**, 13.
90. C. Strobl, A. L. Boulesteix, A. Zeileis and T. Hothorn, *BMC Bioinf.*, 2007, **8**, 21.
91. V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan and B. P. Feuston, *J. Chem. Inf. Comput. Sci.*, 2003, **43**, 1947–1958.
92. J. R. Quinlan, *Mach. Learn.*, 1986, **1**, 81–106.
93. J. R. Quinlan, *Int. J. Human-Comput. Stud.*, 1999, **51**, 497–510.
94. L. Màrquez and H. Rodríguez, in *European Conference on Machine Learning*, Springer, Berlin, Heidelberg, 1998, pp. 25–36.
95. L. M. Roch, F. Häse, C. Kreisbeck, T. Tamayo-Mendoza, L. P. Yunker, J. E. Hein and A. Aspuru-Guzik, *PLoS One*, 2020, **4**, e0229862.
96. L. M. Roch, F. Häse, C. Kreisbeck, T. Tamayo-Mendoza, L. P. E. Yunker, J. E. Hein and A. Aspuru-Guzik, *Sci. Robot.*, 2018, **3**, 2.
97. A. Madry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, *arXiv preprint arXiv:1706.06083*, 2017.
98. A. Kurakin, I. Goodfellow and S. Bengio, *arXiv preprint arXiv:1611.01236*, 2016.
99. N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik and A. Swami, 2017.
100. L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein and J. D. Tygar, 2011.
101. B. Biggio and F. Roli, *Pattern Recognit.*, 2018, **84**, 317–331.
102. W. Y. Ma and S. Osher, *Inverse Problems and Imaging*, **vol. 6**, 2012, pp. 697–708.
103. E. H. Adelson, in *Human Vision and Electronic Imaging VI*, International Society for Optics and Photonics, 2001, **vol. 4299**, pp. 1–12.

104. V. Zantedeschi, M.-I. Nicolae and A. Rawat, presented in part at the Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Dallas, Texas, USA, 2017.
105. N. Papernot, P. McDaniel and I. Goodfellow, *arXiv preprint arXiv:1605.07277*, 2016.
106. I. J. Goodfellow, J. Shlens and C. Szegedy, *arXiv preprint arXiv:1412.6572*, 2014.
107. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.
108. K. a. E. Eykholt, Ivan and Fernandes, Earlence and Li, Bo and Rahmati, Amir and Xiao, Chaowei and Prakash, Atul and Kohno, Tadayoshi and Song, Dawn, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, June.
109. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

CHAPTER 19

Machine Learning for Heterogeneous Catalysis: Global Neural Network Potential from Construction to Applications

SICONG MA(0000-0001-5894-5910), PEI-LIN KANG(0000-0003-2147-2472), CHENG SHANG(0000-0001-7486-1514) AND ZHI-PAN LIU(0000-0002-2906-5217)*

Collaborative Innovation Centre of Chemistry for Energy Material, Shanghai Key Laboratory of Molecular Catalysis and Innovative Materials, Key Laboratory of Computational Physical Science, Department of Chemistry, Fudan University, Shanghai 200433, China
*Email: zpliu@fudan.edu.cn

19.1 Introduction

Heterogeneous catalysts are notable for their great complexity of material composition and surface structures. There is a long history of both experimental and theortical struggling to resolve the active site of catalysts that are responsible for activity and selectivity.[1–4] Recent experimental progress is represented by high spatial resolution techniques, *e.g.* the spherical aberration-corrected high-resolution transmission electron microscope, and synchrotron-based measurement, *e.g.* extended X-ray

absorption fine structure. As a direct tool to correlate atomic structure with energy, theoretical simulations have been widely practiced in modelling catalyst structure and even predicting activity, especially with the advent of density functional theory (DFT) calculations.[5] However, DFT simulation is generally limited to hundreds of atoms, and thus fails to explore exhaustively the phase space of complex catalysts, such as amorphous structures, or multicomponent oxide/alloys.[6–8]

To identify the active site of a catalyst from theory, it is essential to explore the complex PES of exposed surfaces. This requires not only a fast and reliable approach to evaluate the energetics of structures but also an efficient method to explore the structural phase space. As for PES calculation methods, most are based on quantum mechanics as represented by DFT calculations, while the empirical force field calculations, and more recently artificial neural network (NN)[9] potential method (see Figure 19.1), despite their limitations in transferability, are also often utilized in material applications.

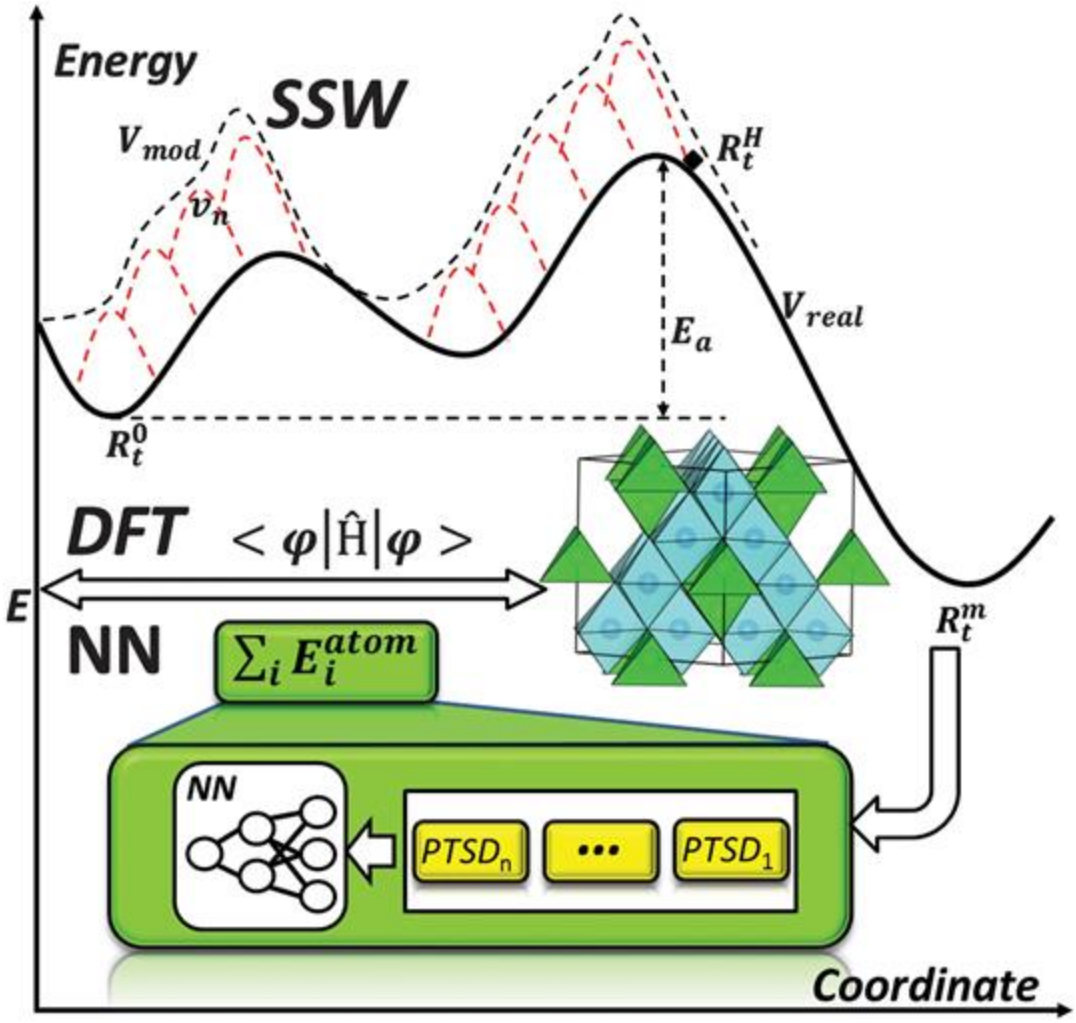


Figure 19.1 Combination of NN potential with SSW global optimization. The total energy, which is the sum of all atomic energies, can be obtained from NN potential. The atomic energy involves power-type structural descriptor (PTSD) computation and an NN evaluation. The SSW global optimization is utilized to explore the PES and produce new structures. Reproduced from ref. 27, https://doi.org/10.1063/1.5113673, under the terms of the CC BY 4.0 license, http://creativecommons.org/licenses/by/4.0/.

As catalytic conversion involves chemical bond making and breaking, DFT calculations have been the most popular way to provide an accurate description of reactions with a reasonable computational cost. The computation speed of DFT is sensitive to the complexity of the employed density-functional (*e.g.* PBE,[10] HSE06[11]) and has a poor scaling (at least *O(NlnN)*). It has great difficulty in exploring the reaction network in large catalytic systems (*e.g.* >100 atoms). As a promising alternative, the NN

potential method developed in the past decade demonstrates its power in treating complex PES problems, from gas-phase reactions to material dynamics.[12–15] Recently, it was also utilized to solve the structures of heterogeneous catalysts, for example, Pd(O),[16] Pt(H),[17] CuAu,[18,19] CuCeO,[20] and CuZnO.[21] Unlike traditional force field potentials, the NN potential is capable of describing chemical reactions with high accuracy as long as the training dataset contains reactive data, such as the transition state (TS).

Paralleling progress in PES evaluation methods, many PES exploration methods have been developed in past decades. As catalytic reactions and structural reconstruction generally occur above ambient temperatures involving high barrier processes, traditional molecular dynamic (MD) simulation is often not appropriate for active site identification. Instead, global optimization methods, which can overcome the high barrier on PES, are desirable. Common global optimization methods include simulated annealing,[22] genetic algorithm,[23] basin hopping[24] and stochastic surface walking (SSW)[25,26]*etc.*

This chapter outlines recent developments in catalyst PES exploration and active site identification. We will show that the combination of a global NN potential with the global optimization SSW method provides a powerful platform to resolve the catalyst structure, which can finally lead to the prediction of catalyst activity from the first principles dataset.[27]

19.2 Methods

Unlike classical force field potentials that utilize physical models to describe interatomic interactions, the NN potential is a numerical function fitted by a large number of parameters. Consequently, the NN potential has a limited predictability beyond the training dataset. The key to improving the quality of the NN potential, therefore, relies heavily on the PES representability of the dataset. To overcome this limitation of the NN potential, we proposed in 2017 a global-to-global scheme to generate a reliable and robust global-NN (G-NN) potential for material simulation.[28] This scheme combines the SSW global sampling method,[29] the high-

dimensional NN (HDNN) for PES description, and a self-learning procedure to expand the dataset and upgrade the G-NN potential. In the following, we elaborate each part of the scheme.

19.2.1 High-dimensional NN Architecture

The artificial NN method was firstly developed to understand signal processing in the brain.[30] In the following decades, NN evolved into a class of powerful algorithms applied to a variety of fields from numerical prediction and pattern recognition to data classification. It is most renowned for the powerful ability to establish the functional relationship between independent variables and target (dependent) values *via* non-linear "black box" data processing.

As an important tool for function fitting, the standard feed-forward NN (FFNN) has been utilized naturally for constructing the PES in small molecular systems, *e.g.* for gas-phase dynamics.[12–15] A major drawback of the traditional FFNN potential is that it is difficult to extend to systems of different size (variable number of atoms). A breakthrough was achieved in 2007 by Behler and Parrinello who implemented an HDNN architecture for material simulation. The method is applicable to high-dimensional systems containing thousands of atoms.[31–33] In the approach, the total energy E_{tot} of the system is written as the summation over all atoms (see eqn (19.1)), see Figure 19.2.

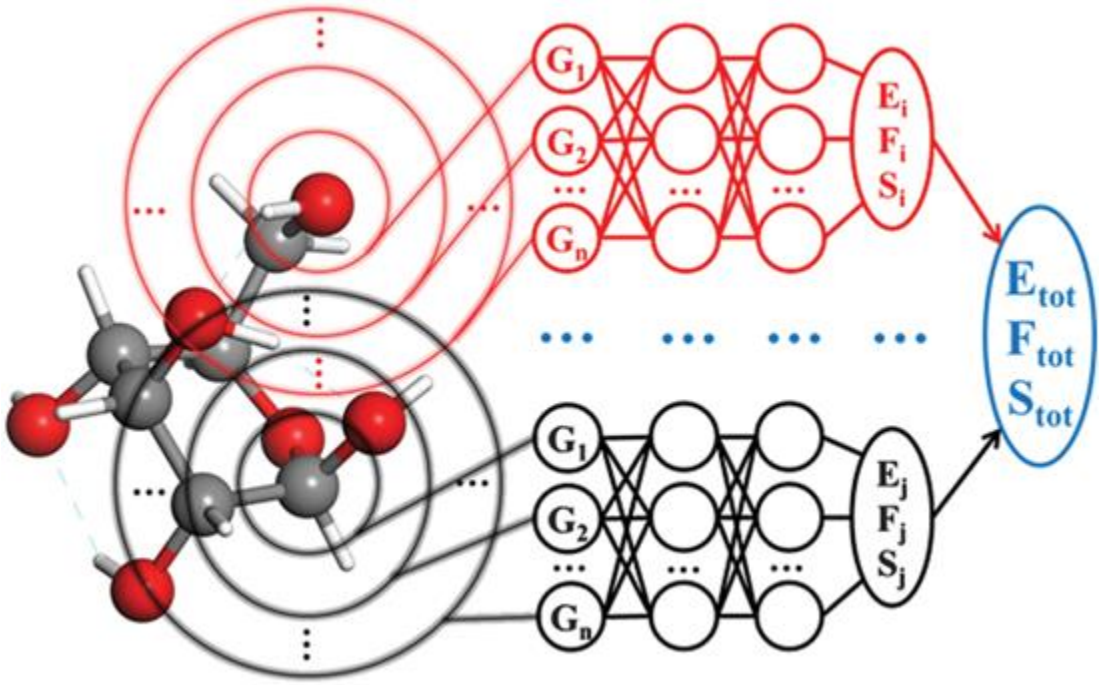


Figure 19.2 The atom-centred structure description scheme in the high-dimensional NN architecture. Each element has a different NN, and each atom is distinguished by its element and structure descriptors centred on it.

Each atom is represented by an atomic FFNN, where the input layer is a series of structural descriptors to represent the atomic bonding environment and the output layer yields the atomic energy E_i (i indexes atoms). This HDNN scheme is applicable to systems with any number of atoms as the total energy is a linear summation of all atomic energy.

$$E_{\text{tot}} = \sum_i E_i(\boldsymbol{R}) \tag{19.1}$$

The input nodes that are used to distinguish the atomic chemical environment are critical for the performance of the NN potential. They utilize the radial and angular information from neighbouring atoms to represent the multi-dimensional interaction between atoms. A good structural descriptor set must be a single-valued function so that different atomic environments yield different values, and they must be continuous and differentiable to enable the calculation of analytic derivatives of

energy.[34–36] More details of structure descriptors are discussed in the next subsection.

For one set of structural descriptors, the atomic force and static stress tensor matrix element $\sigma_{\alpha\beta}$ can be analytically derived according to eqn (19.2), where the force component $F_{k,\alpha}$, α=x, y or z, acting on the atom k is the derivative of the total energy with respect to its coordinate $R_{k,\alpha}$. By combining with eqn (19.2), the force component can be further related to the derivatives of the atomic energy with respect to jth structural descriptors of atom i, $G_{j,i}$.

$$F_{k,\alpha} = -\frac{\partial E^{\text{tot}}}{\partial R_{k,\alpha}} = -\sum_{i,j} \frac{\partial E_i}{\partial G_{j,i}} \frac{\partial G_{j,i}}{\partial R_{k,\alpha}}, \tag{19.2}$$

Similarly, the static stress tensor matrix element $\sigma_{\alpha\beta}$ can be analytically derived using the pairwise distance according to eqn (19.3).

$$\sigma_{\alpha\beta} = -\frac{1}{V} \sum_{i,j,d} \frac{(\boldsymbol{r_d})_\alpha (\boldsymbol{r_d})_\beta}{r_d} \frac{\partial E_i}{\partial G_{j,i}} \frac{\partial G_{j,i}}{\partial r_d}, \tag{19.3}$$

where $\boldsymbol{r_d}$ and r_d are the Cartesian distance vector and its module between atom i and j, respectively, and V is the volume of the structure.

19.2.2 Structural Descriptors

As NN potential correlates the geometry of structure with its energetics, one would wonder what kinds of structural descriptors can achieve the best performance to describe the multi-dimensional PES. Behler and Parrinello suggested a series of rotation-invariant symmetry functions as the structural descriptors, which are the mathematic functions of intrinsic coordinates (pair distance, angles) multiplying the cutoff radial function.[31] Following the same terminology of an empirical forcefield, these functions can be distinguished as two-body terms, three-body terms, and so on. As the radial parts of these functions are Gaussian functions, we label them as Gaussian-

type structure descriptors (GTSD). The mostly used two-body G^2 and three-body G^4 functions are described in eqn (19.4)–(19.6).

$$f_c(R_{ij}) = \begin{cases} 0.5 \times \tan h^3\left[1 - \frac{r_{ij}}{r_c}\right], & \text{for } r_{ij} \le r_c \\ 0 \text{ for } r_{ij} > r_c \end{cases} \tag{19.4}$$

$$G_i^2 = \sum_{j \neq i} e^{-\eta(r-r_s)^2} \cdot f_c(r_{ij}), \tag{19.5}$$

$$G_i^4 = 2^{1-\zeta} \sum_{j,k \neq i}^{\text{all}} (1 + \lambda \cos\theta_{ijk})^{\zeta} \cdot e^{-\eta\left(r_{ij}^2 + r_{ik}^2 + r_{jk}^2\right)} \cdot f_c(r_{ij}) \cdot f_c(r_{ik}) \cdot f_c(r_{jk}), \tag{19.6}$$

where r_{ij} is the inter-nuclear distance between atom i and j, θ_{ijk} is the angle centred at i atom with j, k being neighbours (i, j, k are atom indices). The key ingredients in GTSD are the cutoff function f_c that decays to zero beyond the r_c (see eqn (19.4)), the Gaussian-type radial function, and the trigonometric angular functions. By changing five parameters, r_c, r_s, η, ζ, and λ, a set of G^2 and G^4 functions can then be generated; these serve to distinguish the atomic environment of the centre atom i.

To improve the sensitivity of structural descriptors in learning global PES, we have designed a series of new structural descriptors, power-type structural descriptors (PTSD), see eqn (19.7)–(19.12).

$$S_i^1 = \sum_{j \neq i} R^n(r_{ij}), \tag{19.7}$$

$$S_i^2 = \left[\sum_{m=-L}^{L} \left| \sum_{j \neq i} R^n(r_{ij}) Y_{Lm}(\boldsymbol{r}_{ij}) \right|^2\right]^{\frac{1}{2}} \tag{19.8}$$

$$S_i^3 = 2^{1-\zeta} \sum_{j,k \neq i} (1 + \lambda \cos\theta_{ijk})^{\zeta} \cdot R^n(r_{ij}) \cdot R^m(r_{ik}) \cdot R^p(r_{jk}), \tag{19.9}$$

$$S_i^4 = 2^{1-\zeta} \sum_{j,k \neq i} \left(1 + \lambda \cos\theta_{ijk}\right)^{\zeta} \cdot R^n\left(r_{ij}\right) \cdot R^m(r_{ik}), \tag{19.10}$$

$$S_i^5 = \left[\sum_{m=-L}^{L} \left| \sum_{j,k \neq i} R^n\left(r_{ij}\right) \cdot R^m(r_{ik}) \cdot R^p\left(r_{jk}\right) \cdot \left(Y_{Lm}\left(\boldsymbol{r}_{ij}\right) + Y_{Lm}(\boldsymbol{r}_{ik})\right) \right|^2 \right]^{\frac{1}{2}}, \tag{19.11}$$

$$S_i^6 = 2^{1-\zeta} \sum_{j,k,l \neq i} \left(1 + \lambda \cos\delta_{ijkl}\right)^{\zeta} \cdot R^n\left(r_{ij}\right) R^m(r_{ik}) R^p(r_{il}), \tag{19.12}$$

In PTSD, S^1 and S^2 are two-body functions, S^3, S^4, and S^5 are three-body functions and S^6 is a four-body function. The replacement of the Gaussian function in GTSD by the power function in PTSD has several advantages: (i) the computational cost in numerical calculations is reduced; (ii) the adjustable parameters are reduced from two (r_s, η) to one (n), which simplifies the search for optimal parameters for the two-body function; (iii) the power function when combining with the decaying cutoff function can create radial distributions with flexible peak and shape, which fulfils a similar purpose to the Gaussian function; (iv) the introduction of different powers (n, m, p) in the three-body function can couple conveniently different radial distributions.

19.2.3 Neural Network Training

Once the network architecture is settled, the next step is to determine the weights and biases (NN parameters) in each NN subnet, a process known as the training of the NN. The number of weights and biases is typically $10^4 \sim 10^6$ for the standard FFNN with two hidden layers and 40 nodes in each layers. To train such a large number of NN parameters, a performance function J_{tot} is defined (see eqn (19.13)),[28,37] which measures the deviation of the NN output with respect to the training set properties. The training process will minimize J_{tot} until the accuracy of the NN predicted properties reaches the preset criteria.[36,38,39] The training set is a large structure dataset on PES with accurate energetics and forces, most often computed from first principles calculations.

$$J_{\text{tot}} = J_E + \rho J_F + \tau J_\sigma = \frac{1}{2N}\left(E^{\text{NN}} - E^{\text{real}}\right)^2 + \frac{\rho}{6N}\left(F_{k,\alpha}^{\text{NN}} - F_{k,\alpha}^{\text{real}}\right)^2 + \frac{\tau}{18}\left(\sigma_{\alpha\beta}^{\text{NN}} - \sigma_{\alpha\beta}^{\text{real}}\right)^2 \tag{19.13}$$

where ρ=1∼100 and τ=0.1∼1. This J_{tot} allows the fitting of all three properties simultaneously. For the global optimization of solids, both forces and stresses must be accurate; the most convenient way to do this is to allow the NN training to fit all three terms, either simultaneously or independently (depending on the adjustable parameters, ρ and τ, in eqn (19.13)).

Many gradient-based optimization algorithms have been used to optimize the network weights and biases, such as stochastic gradient descent, conjugate gradient, Levenberg–Marquardt (LM) *etc.* It is generally accepted that quasi-Newton second-order methods, such as Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) and LM, converge more rapidly to the true minimum for large data sets. It should be mentioned that, compared to the dominant computational effort required to generate the data set using quantum mechanics calculations, the training of the NN is, in fact, not the rate-determining step in the whole NN PES construction procedure.

19.2.4 Data Set Generation and SSW Global Optimization

Undoubtedly, the data set used for training the NN largely determines the quality of the NN PES. Previous work utilizes either local PES data, *e.g.* the MD trajectories of a single reaction, or the combination of local PES data from different structural sources, *e.g.* solids, slabs, and clusters. To increase the transferability of the NN potential, as well as to simplify data set generation, the structures produced from global optimization trajectories appear to be the better and more natural choice, which should represent as differently as possible the chemical environment of atoms.

There are currently a variety of global optimization methods for structure prediction, including, for example, simulated annealing,[22] genetic algorithm,[23] basin hopping,[24] and SSW[25,26]*etc.*, which should be able to search a wide area on the PES, and identify unbiasedly the global minimum, even starting from random structures. Among them, basin-hopping and genetic algorithms transform the PES by overlooking the transition region between minima and thus they are likely to miss important

structural patterns at the transition region. On the other hand, simulated annealing and minimum-hopping are based on extensive MD calculations at elevated temperatures, and both methods are rarely utilized for global search in combination with quantum mechanics (even DFT) calculations. Furthermore, the structure patterns from the MD trajectories being closely related could be overwhelmingly redundant for NN training and an iterative structure selection scheme may be required to produce a compact data set. In our approach, we use the SSW global optimization method for data set generation. In the following, we introduce the method in detail.

The central idea of the SSW method combines bias-potential-driven dynamics[40] and Metropolis Monte Carlo (MC).[41] It manipulates smoothly the structural configuration from one minimum to another on PES and relies on Metropolis MC at a given temperature to decide acceptance of the move. A series of consecutive minimum structures, both local and global, is generated in an SSW simulation, forming a continuous trajectory which can provide key information on the reaction pathways between minima. The method was initially developed for aperiodic systems,[26] such as molecules and clusters, and has been extended to periodic crystals.[29]

Each step in SSW, also termed an MC step, comprises three independent parts: (i) climbing, (ii) relaxation, and (iii) Metropolis MC. As schematically illustrated in a one-dimensional PES in Figure 19.1, such an MC step utilizes the climbing module to move uphill and the relaxation module to locate the minimum. Once a minimum is reached, the Metropolis MC is used to judge whether the structure will be accepted or refused. The climbing procedure lies at the heart of the SSW method and is elaborated below in detail.

The climbing module of the SSW involves repeated bias-potential-driven structure extrapolation and local geometry optimization, which gradually drags $\mathbf{R}^{0}_{t}$ to a high energy configuration $\mathbf{R}^{H}_{t}$, where "t" is the index of the current MC step (see Figure 19.1). Starting from the current minimum $\mathbf{R}^{0}_{t}$, SSW first generates a random direction $\mathbf{N}^{0}_{t}$, a normalized vector pointing to a direction to change the current geometry. To enable an unbiased exploration of PES, the initial direction $\mathbf{N}^{0}_{t}$ by design combines two

randomly generated vectors, the so-called global random mode $\mathbf{N}^g_t$ and the local random mode $\mathbf{N}^l_t$, as eqn (19.14).

$$\mathbf{N}^0_t = \frac{\mathbf{N}^g_t + \lambda \mathbf{N}^l_t}{\left|\mathbf{N}^g_t + \lambda \mathbf{N}^l_t\right|} \tag{19.14}$$

where the mixing parameter λ controls the relative proportion of these two displacement directions. Specifically, $\mathbf{N}^g_t$ in our implementation is set as a randomly generated normalized vector that obeys the Maxwell–Boltzmann velocity distribution at 300 K, as utilized in standard molecular dynamics, to generate the initial random velocity. As $\mathbf{N}^g_t$ distributes over a group of atoms, it represents a gentle, global displacement of the atoms. By contrast, $\mathbf{N}^l_t$ describes a stiff, local atomic move, and in our implementation, it is set as a collision movement between two distant atoms. For example, the $\mathbf{N}^l_t$ associated with an atom A (*e.g.*, the first atom in a system) and an atom B (the second atom in the system) can be derived as eqn (19.15), using their coordinates q_A and q_B. The atom pair A and B can be either chosen randomly or learned from previous trajectories, as long as the two atoms are not in close contact (*i.e.*, their distance >3 Å).

$$\mathbf{N}^l_t = \begin{pmatrix} q_B \\ q_A \\ 0 \\ \vdots \end{pmatrix} - \begin{pmatrix} q_A \\ q_B \\ 0 \\ \vdots \end{pmatrix} \tag{19.15}$$

Because reactions with low barriers generally involve soft normal mode directions, it would be ideal to soften the randomly generated $\mathbf{N}^0_t$ towards one eigenvector of the Hessian matrix with small eigenvalues (not necessarily the lowest one). However, the direct computation of the Hessian is expensive and not affordable in a PES search. To solve this, we have developed a numerical approach, the bias-potential constrained Broyden dimer (BP-CBD) method,[42] to soften $\mathbf{N}^0_t$ using eqn (19.16)–(19.19).

$$R_1 = R_0 + N_m \cdot \Delta R \tag{19.16}$$

$$C = \frac{(\boldsymbol{F_1} - \boldsymbol{F_0}) \cdot \boldsymbol{N_m}}{\Delta \boldsymbol{R}} \tag{19.17}$$

$$V_{R_1} = V_{\text{real}} + V_N \tag{19.18}$$

$$V_N = -\frac{a}{2} \cdot \left(\Delta \boldsymbol{R} \cdot \boldsymbol{N_m} \cdot \mathbf{N}_t^0\right)^2 \tag{19.19}$$

Following the unbiased dimer rotation method,[43] we have defined two images separated by a fixed distance of R on PES, namely, $\boldsymbol{R_0}$ and $\boldsymbol{R_1}$ (see eqn (19.16)). The rotation of the dimer by their force ($\boldsymbol{F_0}$ and $\boldsymbol{F_1}$) perpendicular to the dimer vector $\boldsymbol{N_m}$ using constrained Broyden optimization (CBD)[44] will converge to the softest eigenvector of Hessian, and the local curvature (C in eqn (19.17)) can be computed according to the finite difference equation. The softest eigenvector is, however, often not the desired one for reaction as it corresponds to the translational and rotational modes (C=0) in the basin region. Therefore, a biased rotation scheme as implemented in BP-CBD is developed, in which the potential of $\boldsymbol{R_1}$ is modified as eqn (19.18), where V_N is the bias potential added to the real PES V_{real} of $\boldsymbol{R_1}$ that is a quadratic function of coordinates $\boldsymbol{R_1}$ along $\mathbf{N}^0_t$ (see eqn (19.19)). As long as the parameter a (see eqn (19.19)) is large enough, the biased rotation can guarantee the rotation of dimer will not deviate far from $\mathbf{N}^0_t$. The force due to the bias potential in constraining the dimer rotation can be evaluated straightforwardly.

$$V_{\text{mod}} = V_{\text{real}} + \sum_{n=1}^{\text{NG}} v_n = V_{\text{real}} + \sum_{n=1}^{\text{NG}} w_n \times \exp\left[-\frac{\left((\boldsymbol{R_t} - \boldsymbol{R_t^n}) \cdot \boldsymbol{N_t^n}\right)^2}{2 \times ds^2}\right] \tag{19.20}$$

$$\boldsymbol{F}_{\text{mod}} = \boldsymbol{F}_{\text{real}} + \sum_{n=1}^{\text{NG}} w_n \times \exp\left[-\frac{\left((\boldsymbol{R_t} - \boldsymbol{R_t^n}) \cdot \boldsymbol{N_t^n}\right)^2}{2 \times ds^2}\right] \cdot \frac{(\boldsymbol{R_t} - \boldsymbol{R_t^n}) \cdot \boldsymbol{N_t^n}}{2 \times ds^2} \cdot \boldsymbol{N_t^n} \tag{19.21}$$

On moving from $\boldsymbol{R^n_t}$ to a high energy configuration $\boldsymbol{R^H_t}$, a modified PES V_{mod} is utilized, as shown in eqn (19.20), in which a series of

bias Gaussian potentials v_n (n is the index of the bias potential, n=1,2… H) is added one by one consecutively along the direction $\boldsymbol{N}^{\boldsymbol{n}}_{\boldsymbol{t}}$ and thus creates a series of local minima $\boldsymbol{R}^{\boldsymbol{n}}_{\boldsymbol{t}}$ along the moving trajectory on the modified PES. Similar techniques have been used in Metadynamics.[40] The local minimum $\boldsymbol{R}^{\boldsymbol{n}}_{\boldsymbol{t}}$ is identified using local geometry optimization, where the force can be evaluated according to eqn (19.21). The w and ds in eqn (19.20) control the height and the width of the Gaussian function v_n, respectively. While w can be computed on-the-fly to guarantee the success of the uphill move, ds is left as an adjustable parameter in the SSW simulation. It should be mentioned that $\boldsymbol{N}^{\boldsymbol{n}}_{\boldsymbol{t}}$ will always be updated from the initial random direction $\boldsymbol{N}^{\boldsymbol{0}}_{\boldsymbol{t}}$ at each $\boldsymbol{R}^{\boldsymbol{n}}_{\boldsymbol{t}}$ and is then refined using BP-CBD rotation.

To sum up, the uphill movement from $\boldsymbol{R}^{\boldsymbol{n}}_{\boldsymbol{t}}$ to $\boldsymbol{R}^{\boldsymbol{H}}_{\boldsymbol{t}}$ is a repeated procedure containing (i) updating the direction $\boldsymbol{N}^{\boldsymbol{n}}_{\boldsymbol{t}}$ at $\boldsymbol{R}^{\boldsymbol{n}}_{\boldsymbol{t}}$; (ii) adding a new Gaussian function v_n and displacing $\boldsymbol{R}^{\boldsymbol{n}}_{\boldsymbol{t}}$ along the direction $\boldsymbol{N}^{\boldsymbol{n}}_{\boldsymbol{t}}$ by a magnitude of ds; and (iii) relaxing to $\boldsymbol{R}^{\boldsymbol{n+1}}_{\boldsymbol{t}}$ on the modified PES.

The overall efficiency of the SSW method depends on the choice of the step length of surface walking ds and the maximum number of Gaussian potentials $\boldsymbol{H}$. A typical value of ds ranges from 0.2 to 0.6 Å, being 10% to 40% of a typical bond length. The maximum number of Gaussian potentials ($\boldsymbol{H}$) is also system-dependent and in general a number in the range of 6∼15 is a sensible choice. With large ds and large $\boldsymbol{H}$, a large scope of PES can be explored rapidly, but at the expense of resolution on the reaction pathway between minima. Typically, for complex Lennard-Jones clusters (*e.g.*, LJ_{75} with multiple funnels and high barrier), the SSW simulation requires 300∼400 energy/force evaluation per MC step with ds=0.6 and $\boldsymbol{H}$=14. It takes typically 70∼80% computational efforts (energy and gradient evaluation) on the climbing and the remaining 20∼30% on relaxation to a minimum.

19.2.5 Self-learning Procedure

The self-learning procedure is described briefly as follows, also see Figure 19.3. In the first stage, an initial dataset is constructed by performing in

parallel short-time SSW sampling based on first-principle DFT calculations. These DFT calculation are often restricted to small systems (typically below 20 atoms) and with a low accuracy calculation setup to speed up the global PES sampling. After the PES data is obtained from SSW, a small dataset is randomly selected and computed using DFT with a high accuracy calculation setup. This stage produces a dataset with the most common atomic environment for the target PES.

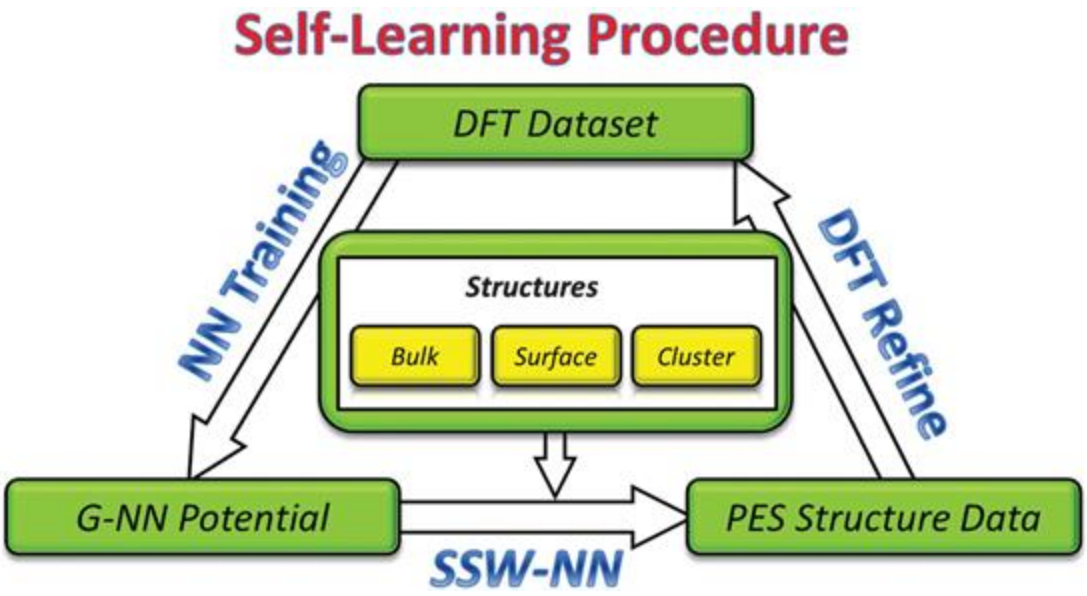


Figure 19.3 Self-learning procedure for the generation of global NN potential (G-NN). Reproduced from ref. 27, https://doi.org/10.1063/1.5113673, under the terms of the CC BY 4.0 license, http://creativecommons.org/licenses/by/4.0/.

The second stage generates an NN potential using the current first principles dataset. The involved hyper-parameters utilized in our NN training can be found in Table 19.1. Our G-NN potential typically has two or three hidden layers and 50∼120 neurons for each layer, which leads to the number of network parameters (weights and biases in NN) in the range of 10^4∼10^6. The optimization of such a large parameter space is achieved *via* the quasi-Newton L-BFGS method by simultaneously fitting the total energy, force, and stress from first principles that constitutes the cost function J_{tot} in eqn (19.13). The training procedure is controlled by a number of hyper-parameters, such as the relative weight among energy,

force, and stress ($1 : \rho : \tau$ in eqn (19.13), network weight initialization method, type of neuron activation function, training epochs *etc.*[45,46] Due to the structural variety of the global PES, the typical accuracy for G-NN is 5∼10 meV per atom for root mean square errors (RMSE) of energy and 0.1∼0.2 eV $Å^{-1}$ for RMSE of force.

Table 19.1 A prototype of hyperparameters used in our previous works.

Hyperparameters	Values
Optimization method	L-BFGS
Weight ratio $1 : \rho : \tau$	1 : 5 : 0.05
Number of hidden layers	2∼3
Neurons number in input layer	100∼400
Neurons number in each hidden layer	50∼120
Weight initialization method	Xavier initialization
Type of neuron activation function	$\tan h^3$
Size of training data	20 000∼100 000
Training epochs	∼20 000
Number of weights and bias	10^4∼10^6

The third stage expands the dataset by carrying out long-time SSW global PES sampling using the current NN potential, the so-called SSW–NN method. These SSW–NN simulations start from a variety of initial structures with different morphology, including bulk, surface and clusters, different chemical compositions, and different numbers of atoms per unit cell. A small additional dataset is thus obtained from the SSW sampling trajectories, containing the structures on PES either randomly selected or exhibiting a new atomic environment (*e.g.* out-of-bounds in structural descriptor, unrealistic energy/force/curvature). After calculating these additional data by DFT, they are added into the training dataset and the whole self-learning procedure returns back to the previous stage.

Typically, after ∼100 iterations, a robust and accurate NN potential can be obtained with a compact training set that contains the most representative structures. We emphasize that it is better to adopt consistent and high accuracy calculation setups in constructing the training dataset using first principles calculations, which can benefit greatly data

transferability and compatibility between systems; this also helps to reduce the NN fitting error.

19.3 Applications

19.3.1 PES Exploration

The thermodynamics and kinetics of a material are determined by the underlying PES. This can be conveniently and efficiently established using an SSW–NN simulation. In recent years, we have mapped out the PESs for a number of systems, *e.g.* single element crystal (boron),[28] molecular crystal (ice),[47] metal oxide (TiO_2, $ZnCr_2O_4$).[48,49] The PESs of TiO_2 and $ZnCr_2O_4$ are projected onto a two-dimensional contour map, as shown in Figure 19.4, where the structural order parameter (**OP**) and the relative energy are the ***x***- and ***y***-axis. The density of the map represented by different colours indicates the density of states on PES, showing the energy degeneracy of structural configurations at the same structural **OP**.

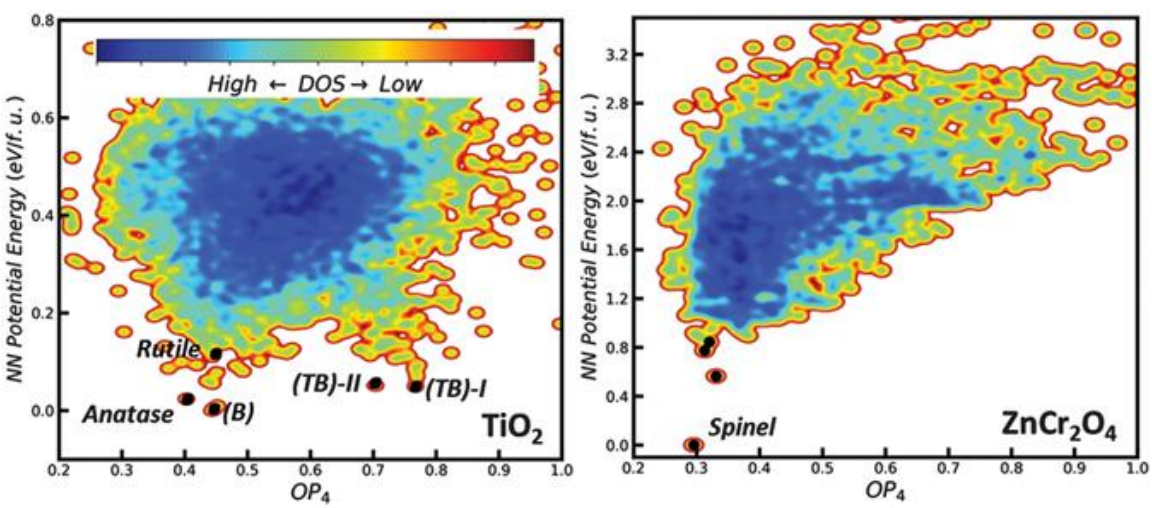


Figure 19.4 Global PESs of TiO_2 and $ZnCr_2O_4$ as determined from SSW–NN global search. The ***x*** axis is the common structure fingerprint for the crystal, namely, the distance-weighted Steinhardt-type order parameter (**OP**) with different angular moment L=2, 4, 6, which can distinguish different minima, and the ***y*** axis is the relative energy ($\Delta \boldsymbol{E}$) of minima. Reproduced from ref. 48 with permission from American Chemical Society,

Copyright 2018, and from ref. 79 with permission from Springer Nature, Copyright 2019.

The **OP** is defined by eqn (19.22).[50]

$$\mathbf{OP}_l = \left(\frac{4\pi}{2l+1} \sum_{m=-l}^{l} \left| \frac{1}{N_{\text{bonds}}} \sum_{i \neq j} e^{-\frac{r_{ij}-r_c}{2r_c}} Y_{lm}(\boldsymbol{n}) \right|^2 \right)^{1/2} \tag{19.22}$$

where Y_{lm} is the spherical harmonic function of degree l and order m; $\boldsymbol{n}$ is the normalized direction between all bonded atoms; i and j are atoms in the lattice, r_{ij} is the distance between atom i and j, and r_c is set at 60% of the typical single bond length for i and j atoms. N_{bonds} is the number of bonds. By choosing a suitable degree, the order parameter can measure the short- and medium-range ordering of atoms in the lattice and thus distinguish important crystal structures and amorphous structures. For example, it is often straightforward to tell the coordination number from the OP value, *e.g.* six-coordinated Ti atom with $\mathbf{OP_2}$=0.3∼0.5 and five-coordinated Ti atom with $\mathbf{OP_2}$=0.6∼0.8 in TiO_2 PES.

As shown in Figure 19.4, the PESs of the TiO_2 and $ZnCr_2O_4$ have rather different shapes: for TiO_2 the PES is shaped somewhat like a butterfly, while for $ZnCr_2O_4$ the PES is funnel-shaped. At the bottom of the PES, where the low energy structures lie, the TiO_2 PES demonstrates that there are many energetically similar TiO_2 crystal phases but with different structural patterns, having multiple funnels in the butterfly shape. By contrast, the $ZnCr_2O_4$ PES has only a well-defined global minimum, a spinel type structure; all other structures are much higher in energy. It is not surprising that more than 10 different TiO_2 crystal structures have been synthesized to date (*e.g.* anatase, rutile, (R) and (H) *etc.*), while only a spinel phase is known for $ZnCr_2O_4$.

In addition to the global minimum, the global PES can also reveal unknown metastable structures, which may act as important phase-transition intermediate states or exhibit attractive physicochemical properties. From the global PES of ice, we have identified the low energy

ice phase with vacancies, which are the key intermediate in the lowest energy pathways for the cubic-to-hexagonal solid phase transition, a reaction central to ice nucleation.[47] We found that the solid transition is facilitated by water vacancy at the ice basal plane interface. The results have profound significance for understanding some everyday phenomena that we observe, ranging from humidity in clouds to the unusual slipperiness of ice. For the TiO_2, it is known in general chemistry that the [TiO_6] octahedron is the common building block for TiO_2 crystals. However, a new class of unprecedented microporous TiO_2 crystalline phases, named TiO_2(TB) in Figure 19.4, have been identified from the global PES, which features a special [TiO_5] trigonal bipyramid building block, a large pore size (5∼7 Å), and high thermal stability.[48] These microporous materials are predicted to be candidate anode materials for Li-ion and Na-ion batteries.[51]

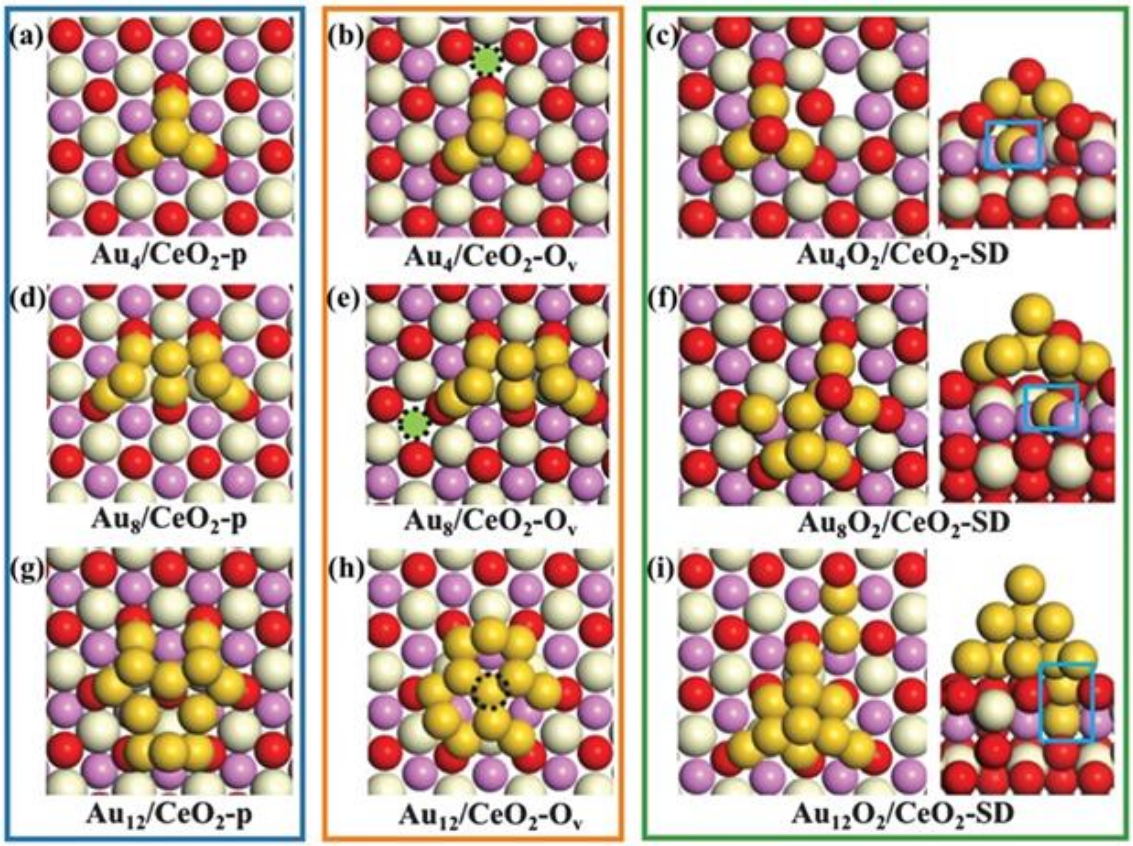
19.3.2 Ultrasmall Au Supported on CeO_2

The atomic structure of supported catalysts must rank top in PES exploration problems. The dual-complexity of nanoparticles/support and the large size of the interfacial system are two key challenges to global structure search. With the advent of SSW–NN, it becomes feasible to answer some long-standing puzzles about supported catalysts, for example, the nature of cationic Au on ceria support. The past two decades have witnessed tremendous research interest in Au-based catalysts that may exhibit outstanding low-temperature catalytic ability for a wide range of reactions.[52,53] Among the catalysts, ceria supported gold (Au/CeO_2) can catalyze the water-gas-shift (WGS) reaction starting from ∼350 K to produce high-grade hydrogen; it has a high stability.[54,55] Owing to the good oxygen storage ability,[56] it was long assumed that the oxygen vacancies on the CeO_2 support play key roles in the Au/CeO_2 system to stabilize the Au nanoparticles.[57] However, seminal work by Fu *et al.* suggested that oxidized gold (*e.g.* Au^+ or Au^{3+}) is the active site for the WGS reaction.[54] The origin of the high catalytic activity of Au/CeO_2 has been a hot subject of debate ever since.

To explore the configurations of ultrasmall Au clusters on CeO_2 and determine their thermodynamics and kinetic stability, we have performed the SSW–NN and enhanced molecular dynamics (MD)[58,59] simulation to obtain the AuCeO global dataset and to build a robust and accurate AuCeO G-NN potential. Our simulations contain structures from 14 to 444 atoms per cell and cover different Au : Ce : O ratios with different morphology forms *e.g.* bulks, layers, and clusters. The final AuCeO global dataset contains 33 654 structures. The AuCeO G-NN potential contains 186 PTSDs for each element, and compatibly, the network involves three-hidden layers (186-60-50-50-1 net), equivalent to ∼51 000 network parameters in total. The final RMSE of energy and force are 6.115 meV per atom and 0.152 eV $Å^{-1}$ respectively.

By establishing the ternary Au-Ce-O G-NN potential based on a first principles global dataset, we have searched for the global minima for a series of Au/CeO_2 systems, the Au cluster size from four to twelve atoms and the CeO_2 support including pristine, the surface with O vacancy and the surface with structural defects (missing O-Ce-O, CeO_2-SD), see Figure 19.5a–i. The energetics of Au clusters on ceria can be evaluated by relating the supported system to the unsupported surface, Au bulk metal and the O_2 molecule (if necessary). As shown, the formation energy (E^{Au}_{form}) of Au clusters on pristine CeO_2 drops gradually from +0.78 eV to +0.69 and to +0.64 eV (eV/Au atom). They are generally poorer compared to them on the other two CeO_2 surfaces, suggesting the ultrasmall clusters do not prefer to anchor on a pristine surface, and the larger cluster is thermodynamically more favoured. The presence of the O vacancy, interestingly, does not much increase the E^{Au}_{form} for the Au clusters. The similar energetics for Au on pristine CeO_2 and CeO_2 with O vacancy indicates that the O vacancy does not induce much additional charge transfer, *e.g.* from CeO_2 support to Au in forming the negatively charged Au cluster. In fact, the additional electrons due to the missing of a surface O are stored in nearby surface Ce atoms, forming two Ce^{3+}. This explanation is supported by DFT calculations that the net magnetic spin for $Au_x/CeO_2–O_v$ systems is generally larger than 2, suggesting that the Au adsorption does not quench the spin on Ce atoms.

On the other hand, the oxidized Au clusters on CeO_2-SD are much stabilized as reflected by the lower E^{Au}_{form}. In particular, Au_4O_2/CeO_2-SD has a negative formation energy, −0.085 eV/Au atom, which suggests that the CeO_2-SD can act as the anchoring site for ultrasmall oxidized Au clusters below 4 Au atoms. With the increase of the cluster size, the energetic preference at the CeO_2-SD sites disappears gradually.


(a)
Au4/CeO2-p
(b)
Au4/CeO2-Ov
(c)
Au4O2/CeO2-SD
(d)
Au8/CeO2-p
(e)
Au8/CeO2-Ov
(f)
Au8O2/CeO2-SD
(g)
Au12/CeO2-p
(h)
Au12/CeO2-Ov
(i)
Au12O2/CeO2-SD

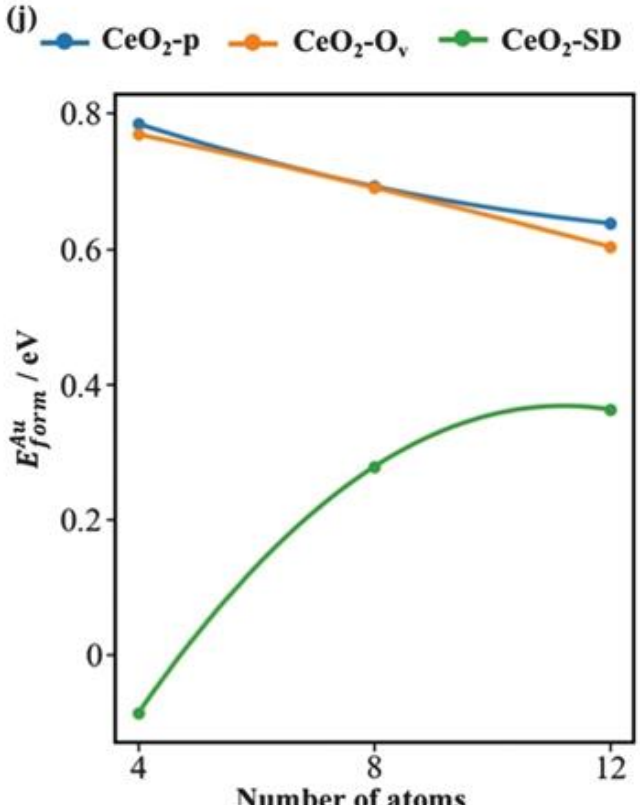

(j)
CeO2-p
CeO2-Ov
CeO2-SD
0.8
0.6
0.4
0.2
0
E^Au_form / eV
4
8
12
Number of atoms

Figure 19.5 Global minimum structures for Au clusters on CeO_2 determined by SSW–NN. (**a–i**) Structure snapshots of Au clusters on three different CeO_2 surfaces. For Au_x/CeO_2-SD, both the top and the side views are shown for clarity. O atoms are in red/pink, and Ce atoms are in white. The surface layer O and the subsurface O are in red (darker) and pink (lighter), respectively. Au atoms are in yellow (for example, the central cluster in (g)). The oxygen vacancy is indicated by dashed circles in Au_x/CeO_2-O_v. The embedded Au atom is highlighted by a blue rectangle. (**j**) The formation energy of Au cluster relative to the CeO_2 surface and Au bulk.

Our results show that the ultrasmall cationic Au clusters attaching to surface structural defects are the only stable structural pattern, *e.g.* Au_4O_2/CeO_2-SD, where Au clusters can donate electrons to the underlying Ce mediated by the Au–O bonding (see Figure 19.5j). The Au_4O_2/CeO_2-SD is believed to be the potential stable active structure for an ultrasmall Au cluster on the CeO_2 surface. The other clusters on different CeO_2 surfaces have a strong energy preference to grow into bulky Au metal. We demonstrate that the global PES exploration is critical for understanding the morphology of metal clusters on oxide support, which now can be realized *via* the NN method.

19.3.3 Hydrogen Evolution Reaction on Amorphous TiO_xH_y

Black titania (TiO_2), which absorbs strongly across the entire visible-light spectrum, has been found to exhibit hydrogen evolution reaction (HER) activity several orders higher than convention TiO_2 material.[60–65] Synthesized by hydrogenating pristine TiO_2, the as-synthesized black TiO_2 commonly exhibits a core–shell structure with the amorphous shell, a few nanometers thick, coated on the anatase crystals.[61,66–68] The amorphous shell is believed to provide the catalytic active site and to be responsible for the enhanced HER activity. However, structure determination for the amorphous shell is challenging both experimentally and theoretically, without even mentioning the need to explain the high HER activity.

To resolve the HER active sites on the amorphous TiO_xH_y shell, we have performed SSW–NN simulations to obtain the TiO_xH_y global dataset with

143 786 structures and build a robust and accurate TiO_xH_y G-NN potential. The structures in the database cover different Ti : O : H ratios, mainly Ti_4O_7, Ti_4O_8, $Ti_4O_8H_x$ x=1∼4, $Ti_8O_{16}H_x$, x=1∼4 with bulk, layer, and cluster forms, and also contain large surface systems such as $Ti_{56}O_{112}H_x$ and molecular systems (H_2, H_2O). The TiO_xH_y G-NN potential utilizes 201 PTSDs for each element, *i.e.* 77 two-body PTSDs, 108 three-body PTSDs and 16 four-body PTSDs, and the network involves two-hidden layers, each with 50 neurons, equivalent to 38 103 network parameters in total. The final RMSE of energy and force are 9.8 meV per atom and 0.22 eV $Å^{-1}$ respectively.

Using the TiO_xH_y G-NN potential, the thermodynamics phase diagram of TiO_2 bulk and surface in contact with H_2 at different temperatures and pressures can thus be determined quantitatively.

We found that, among common anatase surfaces, only the ridged anatase (112) surface can reconstruct significantly by surface H atom and a local high H coverage, 0.69 ML (see Figure 19.6A), can gradually build up during the surface amorphization. The amorphous surface exposes different Ti cations, 25% four-coordinated Ti_{4c}, 50% five-coordinated Ti_{5c} and 25% six-coordinated Ti_{6c} atoms (see Figure 19.6B). Consistently, the Ti-O bond length has a wide distribution, from 1.8 to 2.2 Å, as compared to 1.9∼2.1 Å on a perfect TiO_2 surface. This high H coverage not only renders the black colour (see Figure 19.6C) of the amorphous TiO_2 but also provides a low energy reaction channel for HER: a transient Ti-H hydride becomes likely to form on the exposed Ti atoms of amorphous surface and its reaction with neighbouring OH has a much lower barrier. The nascent TiH hydride can react facilely with the neighbouring OH to produce H_2 (the barrier is 0.6 eV), where the barrier is more than 1 eV lower than the traditional H coupling channel *via* two surface OH groups (barrier >1.9 eV), as shown in Figure 19.6D–F.[49,69] This finding of TiH hydride on hydrogenated amorphous TiO_2 is exceptional as it is known to be highly unstable on crystalline TiO_2 both in theory and from experiment. The stability of TiH hydride can be attributed to the electron transfer from the reduced Ti cation

and the strong polarization of neighbouring OH groups, both being only available on amorphous surfaces with high H coverage.

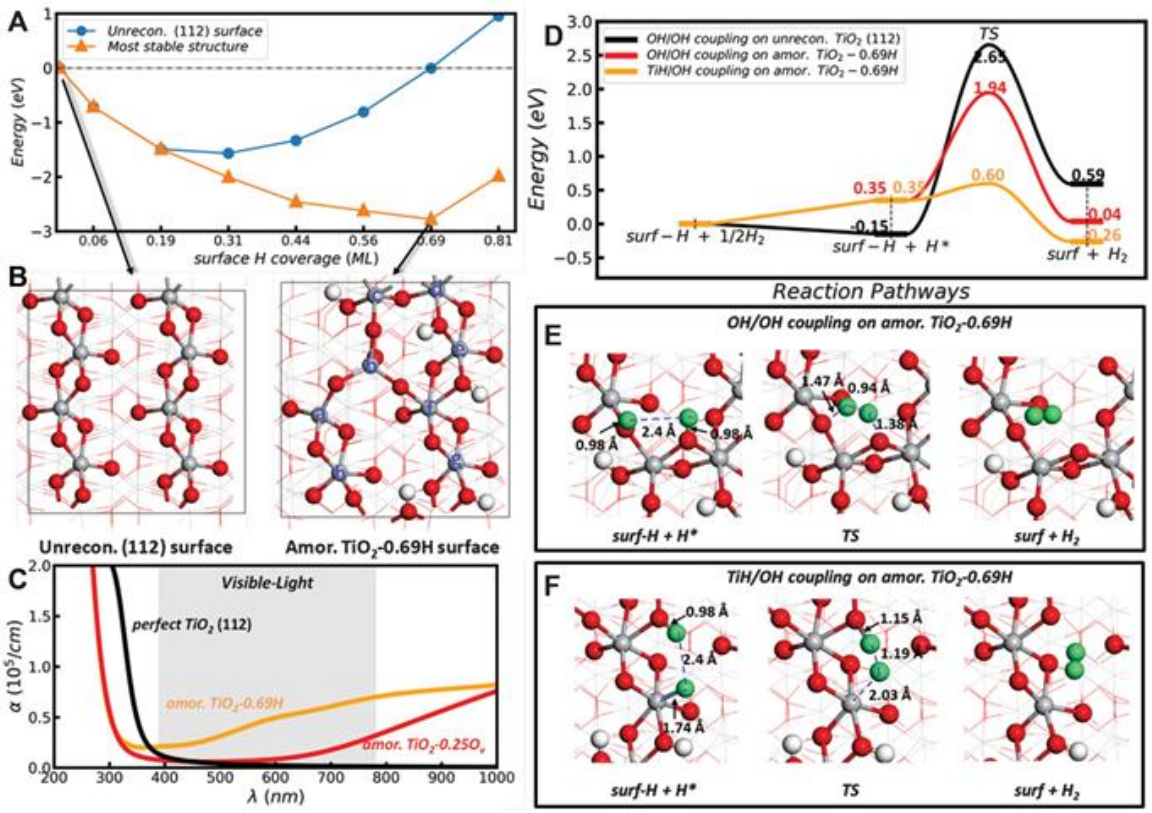


Figure 19.6 (**A** and **B**) the relative energy and selected structures for the H covered (112), TiO_2-*x*H with the coverage x from 0 to 0.81 ML. All energetics are from DFT calculations and with respect to the energy of pristine (112) and H_2 molecule. In (**B**), the reconstructed surfaces after hydrogenation become amorphous with the appearance of various Ti coordinations. For the (4×2) (112) surface studied, there are two Ti_{4c}, four Ti_{5c} and two Ti_{6c} in amorphous TiO_2-0.69H. (**C**) The optical absorption spectrum for TiO_2 surfaces. (**D**) The energetic profiles of H coupling *via* the OH/OH coupling and the TiH/OH coupling mechanisms on the pristine (112) surface and amorphous TiO_2-0.69H. (**E**, **F**) The reaction snapshot from the top view of (112). Ti: light grey balls; O: red (darker) balls. H: white balls (H in reaction: balls connected with dotted lines). Reproduced from ref. 49 with permission from American Chemical Society, Copyright 2018.

We note that after the theoretical work was published, recent experimental work reveals the presence of TiH hydride on amorphous TiO_2, where the characteristic H chemical shift δ value in 1H Nuclear Magnetic

Resonance spectroscopy has a negative peak (−0.6 ppm), which grows with an increase of hydrogenation time.[70]

19.3.4 Syngas Conversion on ZnCrO Catalysts

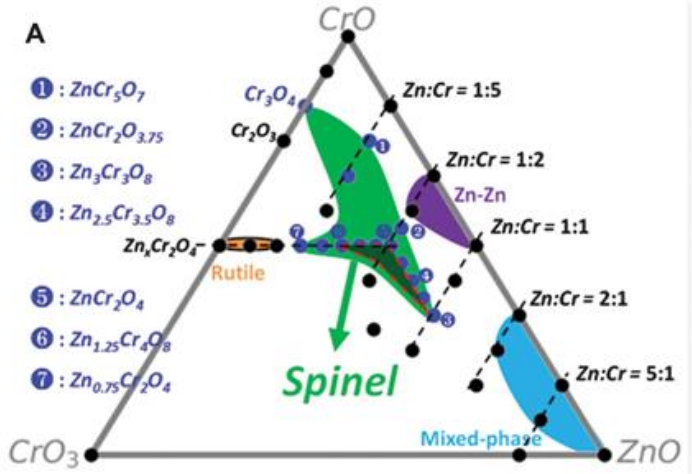
As one of the first generation industry catalysts developed by BASF for syngas-to-methanol,[71] zinc-chromium oxide (ZnCrO) catalyst has been extensively studied since the 1930s. It has been observed that the Zn : Cr ratios can significantly influence the syngas-to-methanol catalytic activity and selectivity.[72–75] The best activity and selectivity is generally achieved in the window of Zn : Cr=1 : 1 to 2 : 1, *e.g.* methanol yield 90∼600 $g*kg_{cat}^{-1}*h^{-1}$ and selectivity 80% at the Zn : Cr=1 : 1 at 300 °C.[75] The Zn : Cr=1 : 2 catalysts with pure $ZnCr_2O_4$ spinel, by contrast, yield rather poor activity and selectivity, <5 $g*kg_{cat}^{-1}*h^{-1}$ and 14% selectivity to methanol (∼45% to alkane).[75] An increase of reaction temperature to 400 °C, by contrast, leads to preferential generation of methane rather than methanol on Zn : Cr=1 : 2 catalysts (methane selectivity ∼75%). For the structures, it is believed that $ZnCr_2O_4$ with the spinel crystal form is the most stable phase as it forms after calcination at high temperature.[76] The atomic structure of ZnCrO catalysts with Zn : Cr ratios between 1 : 1 and 1 : 2 remains, however, uncertain as no clear evidence of ZnO formation was found, although the X-ray diffraction (XRD) patterns exhibit peak broadening and small peak shifting with respect to the $ZnCr_2O_4$ spinel phase.[75,77,78] Further increasing Zn : Cr ratios (>1 : 1), ZnO aggregation in the catalysts has been detected by XRD, but quantitative analysis shows that the amount of ZnO is far less than the theoretical value if the catalyst is assumed to be composed of $ZnCr_2O_4$ and ZnO.[75,77] The ZnCrO structural uncertainty hinders the active site resolving; no one exactly knows what is the active site appearing on the catalyst with Zn : Cr=1 : 1.

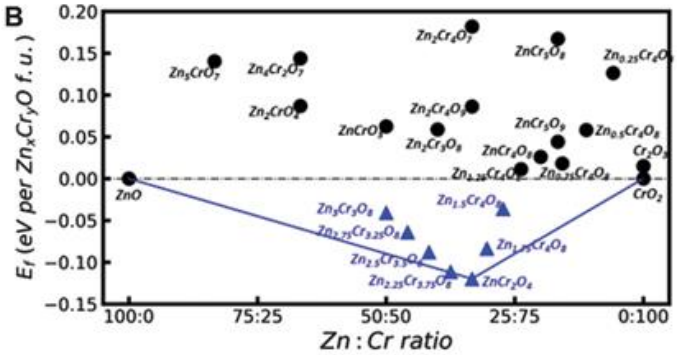
To resolve where and how syngas conversion occurs on ZnCrO catalyst, we have performed the SSW–NN simulation to obtain the ZnCrO global dataset and to build a robust and accurate ZnCrO G-NN potential. Our simulations contain structures from 10 to 84 atoms per cell and cover different Zn : Cr : O ratios, *i.e.* ZnO, CrO_x, $ZnCr_xO_y$, with different

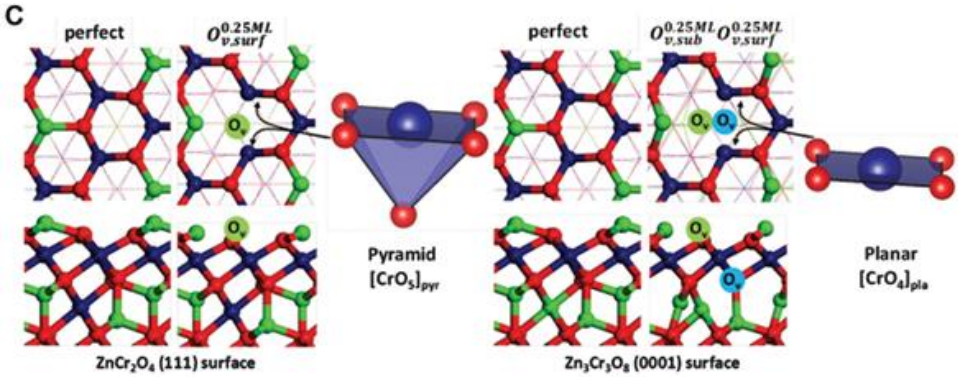
morphology forms *e.g.* bulks, layers and clusters. The final ZnCrO global dataset contains 38 285 structures. The ZnCrO G-NN potential contains 324 PTSDs for each element, *i.e.* 132 two-body PTSDs, 170 three-body PTSDs and 22 four-body PTSDs, and compatibly, the network involves three-hidden layers (324-80-60-60-1 net), equivalent to 103 743 network parameters in total. The final RMSE of energy and force are 4.3 meV per atom and 0.128 eV $Å^{-1}$ respectively.[79]

Based on the ZnCrO G-NN potential, we are able to follow for the first time the ZnCrO structural evolution with the change of Zn : Cr : O and establish the Zn-Cr-O ternary phase diagram.

As shown in Figure 19.7A, the spinel-type crystalline structures appear as a major motif for the ZnCrO ranging from Zn : Cr=0 : 1 to 1 : 1. Thermodynamics phase diagram of Zn-Cr-O further reveals the presence of a small stable composition island, *i.e.* Zn : Cr : O=6 : 6 : 16–3 : 8 : 16, where the oxide alloy tends to crystallize into a spinel phase (see Figure 19.7B). The derivation of Zn : Cr ratio from 1 : 2 to 1 : 1 leads the generation of $Zn_3Cr_3O_8$ metastable crystal phase which contains the highest concentration of unusual $[ZnO_6]$ octahedra (O_h) in bulk. This subtle structural difference turns out to critically affect syngas conversion activity and selectivity. Owing to the presence of $[ZnO_6]_{Oh}$, the oxygen vacancy (O_v) formation ability increases appreciably and extends from surface to subsurface with the increase of Zn : Cr ratio, as occurring dynamically under syngas reaction conditions. Under syngas reaction conditions, the catalyst surfaces with Zn : Cr=1 : 1 expose the extraordinary four-coordinated planar Cr^{2+} cation, while only five-coordinated Cr cation with the pyramid geometry can be exposed on the catalyst surfaces with Zn : Cr=1 : 2 (see Figure 19.7C).


A
CrO
CrO3
ZnO
Cr3O4
Cr2O3
ZnxCr2O4
Rutile
Spinel
Zn-Zn
Mixed-phase
Zn:Cr = 1:5
Zn:Cr = 1:2
Zn:Cr = 1:1
Zn:Cr = 2:1
Zn:Cr = 5:1
1 : ZnCr5O7
2 : ZnCr2O3.75
3 : Zn3Cr3O8
4 : Zn2.5Cr3.5O8
5 : ZnCr2O4
6 : Zn1.25Cr4O8
7 : Zn0.75Cr2O4


B
Ef (eV per ZnxCryO f. u.)
Zn : Cr ratio
100:0
75:25
50:50
25:75
0:100
ZnO
CrO2
ZnCr2O4


C
perfect
Pyramid
[CrO5]pyr
perfect
Planar
[CrO4]pla
ZnCr2O4 (111) surface
Zn3Cr3O8 (0001) surface

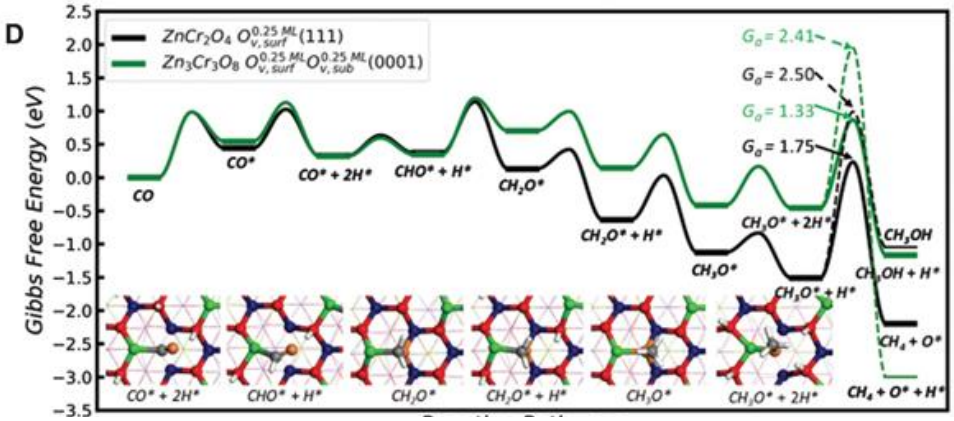

D
Gibbs Free Energy (eV)
ZnCr2O4 (111)
Zn3Cr3O8 (0001)
CO
CO*
CO* + 2H*
CHO* + H*
CH2O*
CH2O* + H*
CH3O*
CH3O* + 2H*
CH3OH
CH3OH + H*
CH4 + O*
CH4 + O* + H*
Go = 2.41
Go = 2.50
Go = 1.33
Go = 1.75

Figure 19.7 (**A**) Ternary Zn-Cr-O phase diagram. The green region (centre top) maps out that the composition with the spinel-type skeleton structure as the global minimum, where numbered circles indicate the composition. (**B**) Convex hulls for all ZnCrO structures. The central triangle represents the negative formation energy compared with ZnO and CrO_2 phases. (**C**) The perfect and O-vacant $ZnCr_2O_4O^{0.25\ ML}{}_{v,\ surf}$ (111) and $Zn_3Cr_3O_8O^{0.25\ ML}{}_{v,\ surf}O^{0.25\ ML}{}_{v,\ sub}$ (0001) surface structures. The local $[CrO_x]$ configurations nearby O_v are highlighted. These O_v surfaces are available under reaction conditions. (**D**) Gibbs free energy reaction profiles for syngas conversion on two ZnCrO catalysts at 573 K and 2.5 MPa syngas (H_2:CO=1.5). The reaction snapshots are shown in the inset of (**D**), (where, in the online version, the colour scheme for atoms is as follows: Zn: green; Cr: purple; O: red; O in CO: orange; C: grey and H: white). Reproduced from ref. 79 with permission from Springer Nature, Copyright 2019.

Further DFT investigation on the syngas conversion reaction on $Zn_3Cr_3O_8$ and $ZnCr_2O_4$ proves that the four-coordinated planar Cr^{2+} cation site on $Zn_3Cr_3O_8$ surface is critical for methanol selectivity. On the other hand, the $ZnCr_2O_4$ phase shows a low catalytic activity and predominantly yields methane (see Figure 19.7D). The microkinetics simulation based on DFT reaction energetics (methanol yield: ∼670 g kg_{cat}^{-1} h^{-1} on $Zn_3Cr_3O_8$; methane yield: ∼0.08 g kg_{cat}^{-1} h^{-1} on $ZnCr_2O_4$) further rationalizes the sharp difference in activity and selectivity observed experimentally: the ZnCrO catalysts with Zn : Cr=1 : 1 shows the high activity and high methanol selectivity (methanol yield 80∼600 g*kg_{cat}^{-1}*h^{-1} and selectivity 80% at 573 K), but the ZnCrO catalysts with Zn : Cr=1 : 2 show low activity and low methanol selectivity (<5 g*kg_{cat}^{-1}*h^{-1} with 14% selectivity to methanol at 573 K and∼82% to hydrocarbon (91% CH_4) at 673 K).[75,80]

The catalytic activity and selectivity differences between two different spinel phases can be largely attributed to the sensitivity of the CH_3O adsorption strength on two different surfaces (see Figure 19.7D). Too strong adsorption of CH_3O on $ZnCr_2O_4$ surface apparently hinders the formation of CH_3OH and also leads to a high barrier for CH_4 formation, whilst the moderate adsorption of CH_3O on $Zn_3Cr_3O_8$ surface promotes the CH_3OH

channel. The findings on ZnCrO catalysts not only strengthen our understandings, from bulk to surfaces and to active sites, on the high-temperature syngas conversion on oxides, but also suggest that other complex oxide systems hotly debated in experiment, *e.g.* ZnZrO, CuZnO, ZnMnO, and ZnFeO, should now be possible to explore with the advent of the SSW–NN method.

19.4 Prospects

This chapter overviews the SSW–NN method and its recent applications in heterogeneous catalysis. While SSW–NN holds great promise in solving some challenging tasks in heterogeneous catalysis, ranging from catalyst active site structure to catalytic activity as demonstrated in the above examples, one must bear in mind that a successful application of NN potential relies much on the sampled dataset to the target problem. There are still some challenges to be solved in the future.

(i) Reaction PES sampling, especially molecular reactions, have a much higher computational cost than typical material PES sampling. This is due to the general high reaction barrier of molecular reactions, even in the presence of surfaces. It becomes especially problematic to capture reactions with a sharp TS region (*i.e.* characterized by a large negative curvature), where the exact TS position is difficult to capture by random sampling. As introduced above, the current NN potential generally focuses on materials such as TiO_2 and ZnCrO, and a giant step forward would couple molecular reactions with the solid materials to establish reactive NN potentials for heterogeneous catalysis.

(ii) The iterative self-learning in building the global NN potential may become highly computationally demanding when the system size is large and the bonding complexity increases. In particular, the large configurational space of the bonding environment in multi-element systems is a key problem in constructing the NN potential. For example, there are many possible bonding patterns for carbon with

other elements (C, H, O, and N) in organic chemistry, which makes it challenging to build a general-purpose reactive NN potential for organic chemistry.

To enhance the reaction sampling and to reduce the system degree of freedom effectively, we expect that the combination of other techniques with SSW–NN, such as pattern recognition artificial intelligence methods, coarse graining, and rigid body methods, could be the future direction to power up SSW–NN for complex catalysis problems.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (2018YFA0208600), National Science Foundation of China (21573149, 21533001 and 91745201).

References

1. I. E. Wachs and C. A. Roberts, *Chem. Soc. Rev.*, 2010, **39**, 5002.
2. J. K. Nørskov, T. Bligaard, B. Hvolbæk, F. Abild-Pedersen, I. Chorkendorff and C. H. Christensen, *Chem. Soc. Rev.*, 2008, **37**, 2163.
3. T. F. Jaramillo, K. P. Jørgensen, J. Bonde, J. H. Nielsen, S. Horch and I. Chorkendorff, *Science*, 2007, **317**, 100.
4. U. Deka, I. Lezcano-Gonzalez, B. M. Weckhuysen and A. M. Beale, *ACS Catal.*, 2013, **3**, 413.
5. R. G. Parr, in *Horizons of Quantum Chemistry*, Springer, 1980, p. 5.
6. A. Y. Khodakov, W. Chu and P. Fongarland, *Chem. Rev.*, 2007, **107**, 1692.
7. S. K. Seth, I. Saha, C. Estarellas, A. Frontera, T. Kar and S. Mukhopadhyay, *Cryst. Growth Des.*, 2011, **11**, 3250.
8. J. Kotakoski, A. Krasheninnikov, U. Kaiser and J. Meyer, *Phys. Rev. Lett.*, 2011, **106**, 105505.
9. T. Kohonen, *Neural Netw.*, 1988, **1**, 3.

10. J. P. Perdew, K. Burke and M. Ernzerhof, *Phys. Rev. Lett.*, 1996, **77**, 3865.
11. J. Heyd, G. E. Scuseria and M. Ernzerhof, *J. Chem. Phys.*, 2003, **118**, 8207.
12. B. G. Sumpter and D. W. Noid, *Chem. Phys. Lett.*, 1992, **192**, 455.
13. T. H. Fischer, W. P. Petersen and H. P. Lüthi, *J. Comput. Chem.*, 1995, **16**, 923.
14. L. Raff, M. Malshe, M. Hagan, D. Doughan, M. Rockley and R. Komanduri, *J. Chem. Phys.*, 2005, **122**, 084104.
15. S. Manzhos, X. Wang, R. Dawes and T. Carrington, *J. Phys. Chem. A*, 2006, **110**, 5295.
16. J. R. Boes and J. R. Kitchin, *Mol. Simulat.*, 2017, **43**, 346.
17. G. Sun and P. Sautet, *J. Am. Chem. Soc.*, 2018, **140**, 2812.
18. N. Artrith and A. M. Kolpak, *Nano Lett.*, 2014, **14**, 2670.
19. N. Artrith and A. M. Kolpak, *Comput. Mater. Sci.*, 2015, **110**, 20.
20. J. S. Elias, N. Artrith, M. Bugnet, L. Giordano, G. A. Botton, A. M. Kolpak and Y. Shao-Horn, *ACS Catal.*, 2016, **6**, 1675.
21. N. Artrith, B. Hiller and J. Behler, *Phys. Status Solidi B*, 2013, **250**, 1191.
22. S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, *Science*, 1983, **220**, 671.
23. S. Sivanandam and S. Deepa, in *Introduction to Genetic Algorithms*, Springer, 2008, p. 165.
24. D. J. Wales and J. P. Doye, *J. Phys. Chem. A*, 1997, **101**, 5111.
25. C. Shang and Z.-P. Liu, *J. Chem. Theory Comput.*, 2013, **9**, 1838.
26. X.-J. Zhang, C. Shang and Z.-P. Liu, *J. Chem. Theory Comput.*, 2013, **9**, 3252.
27. S. Ma, C. Shang and Z.-P. Liu, *J. Chem. Phys.*, 2019, **151**, 050901.
28. S.-D. Huang, C. Shang, P.-L. Kang and Z.-P. Liu, *Chem. Sci.*, 2018, **9**, 8644.
29. C. Shang, X.-J. Zhang and Z.-P. Liu, *Phys. Chem. Chem. Phys.*, 2014, **16**, 17845.
30. W. S. McCulloch and W. A. Pitts, *Bull. Math. Biophys.*, 1943, **5**, 115.
31. J. Behler and M. Parrinello, *Phys. Rev. Lett.*, 2007, **98**, 146401.
32. J. Behler, *J. Chem. Phys.*, 2011, **134**, 074106.
33. J. Behler, *J. Phys.: Condens. Matter*, 2014, **26**, 183001.

34. A. P. Bartók, R. Kondor and G. Csányi, *Phys. Rev. B*, 2013, **87**, 184115.
35. K. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. Müller and E. Gross, *Phys. Rev. B*, 2014, **89**, 205118.
36. J. Behler, *J. Chem. Phys.*, 2016, **145**, 170901.
37. S.-D. Huang, C. Shang, X.-J. Zhang and Z.-P. Liu, *Chem. Sci.*, 2017, **8**, 6327.
38. J. Behler, *Phys. Chem. Chem. Phys.*, 2011, **13**, 17930.
39. J. Behler, *Angew. Chem., Int. Ed.*, 2017, **56**, 12828.
40. M. Iannuzzi, A. Laio and M. Parrinello, *Phys. Rev. Lett.*, 2003, **90**, 238302.
41. N. Metropolis, A. W. Rosenbluth, N. M. Rosenbluth, A. H. Teller and E. Teller, *J. Chem. Phys.*, 1953, **21**, 1087.
42. C. Shang and Z.-P. Liu, *J. Chem. Theory Comp*, 2012, **8**, 2215.
43. G. Henkelman and H. Jónsson, *J. Chem. Phys.*, 1999, **111**, 7010.
44. C. Shang and Z.-P. Liu, *J. Chem. Theory Comput.*, 2010, **6**, 1136.
45. D. Coyle, G. Prasad and T. M. McGinnity, *IEEE Trans. Syst. Man Cybern. B*, 2009, **39**, 1458.
46. T. Domhan, J. T. Springenberg and F. Hutter, *Twenty-fourth International Joint Conference on Artificial Intelligence*, **vol. 2015**, 2015.
47. S.-H. Guan, C. Shang, S.-D. Huang and Z.-P. Liu, *J. Phys. Chem. C*, 2018, **122**, 29009.
48. S. Ma, S.-D. Huang, Y.-H. Fang and Z.-P. Liu, *ACS Appl. Energy Mater.*, 2018, **1**, 22.
49. S. Ma, S.-D. Huang, Y.-H. Fang and Z.-P. Liu, *ACS Catal.*, 2018, **8**, 9711.
50. X.-J. Zhang, C. Shang and Z.-P. Liu, *Phys. Chem. Chem. Phys.*, 2017, **19**, 4725.
51. S.-H. Choe, C.-J. Yu, K.-C. Ri, J.-S. Kim, U.-G. Jong, Y.-H. Kye and S.-N. Hong, *Phys. Chem. Chem. Phys.*, 2019, **21**(16), 8408–8417.
52. M. Haruta, *Nature*, 2005, **437**, 1098.
53. A. S. K. Hashmi and G. J. Hutchings, *Angew. Chem., Int. Ed.*, 2006, **45**, 7896.
54. Q. Fu, H. Saltsburg and M. Flytzani-Stephanopoulos, *Science*, 2003, **301**, 935.

55. H. Sakurai, T. Akita, S. Tsubota, M. Kiuchi and M. Haruta, *Appl. Catal., A*, 2005, **291**, 179.
56. N. J. Lawrence, J. R. Brewer, L. Wang, T.-S. Wu and J. Wells-Kingsbury, *et al.Nano Lett.*, 2011, **11**, 2666.
57. D. Widmann, R. Leppelt and R. J. Behm, *J. Catal.*, 2007, **251**, 437.
58. A. Barducci, M. Bonomi and M. Parrinello, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2011, **1**, 826.
59. J. Kästner, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 2011, **1**, 932.
60. X. Chen, L. Liu and F. Huang, *Chem. Soc. Rev.*, 2015, **44**, 1861.
61. X. Chen, L. Liu, Y. Y. Peter and S. S. Mao, *Science*, 2011, **331**, 746.
62. A. Naldoni, M. Allieta, S. Santangelo, M. Marelli, F. Fabbri, S. Cappelli, C. L. Bianchi, R. Psaro and V. Dal Santo, *J. Am. Chem. Soc.*, 2012, **134**, 7600.
63. X. Yu, B. Kim and Y. K. Kim, *ACS Catal.*, 2013, **3**, 2479.
64. N. Liu, C. Schneider, D. Freitag, M. Hartmann, U. Venkatesan, J. Müller, E. Spiecker and P. Schmuki, *Nano Lett.*, 2014, **14**, 3309.
65. H. Lu, B. Zhao, R. Pan, J. Yao, J. Qiu, L. Luo and Y. Liu, *RSC Adv.*, 2014, **4**, 1128.
66. Y. H. Hu, *Angew. Chem., Int. Ed.*, 2012, **51**, 12410.
67. X. Chen, L. Liu, Z. Liu, M. A. Marcus, W.-C. Wang and N. A. Oyler, *et al.Sci. Rep.*, 2013, **3**, 1510.
68. Z. Wang, C. Yang, T. Lin, H. Yin, P. Chen and D. Wan, *et al.Energy Environ. Sci.*, 2013, **6**, 3007.
69. U. Aschauer and A. Selloni, *Phys. Chem. Chem. Phys.*, 2012, **14**, 16595.
70. Y. Guo, S. Chen, Y. Yu, H. Tian, Y.-L. Zhao and Y. J.-C. Ren, *et al.J. Am. Chem. Soc.*, 2019, **141**, 8407–8411.
71. K. Waugh, *Catal. Today*, 1992, **15**, 51.
72. M. C. Molstad and B. F. Dodge, *Ind. Eng. Chem.*, 1935, **27**, 134.
73. E. Errani, F. Trifiro, A. Vaccari, M. Richter and G. Del Piero, *Catal. Lett.*, 1989, **3**, 65.
74. M. C. Bradford, M. V. Konduru and D. X. Fuentes, *Fuel Process. Technol.*, 2003, **83**, 11.
75. H. Song, D. Laudenschleger, J. J. Carey, H. Ruland, M. Nolan and M. Muhler, *ACS Catal.*, 2017, **7**, 7610.

76. R. Dumitru, F. Manea, C. Păcurariu, L. Lupa, A. Pop, A. Cioablă, A. Surdu and A. Ianculescu, *Catalysts*, 2018, **8**, 210.
77. G. Del Piero, F. Trifiro and A. Vaccari, *J. Chem. Soc., Chem. Commun.*, 1984, 656.
78. M. Bertoldi, B. Fubini, E. Giamello, F. Trifirò and A. Vaccari, *J. Chem. Soc., Faraday Trans.*, 1988, **84**, 1405.
79. S. Ma, S.-D. Huang and Z.-P. Liu, *Nat. Catal*, 2019, **2**, 671.
80. K. Cheng, B. Gu, X. Liu, J. Kang, Q. Zhang and Y. Wang, *Angew. Chem., Int. Ed.*, 2016, **128**, 4803.

CHAPTER 20

A Few Guiding Principles for Practical Applications of Machine Learning to Chemistry and Materials

S. SHANKAR*[a] AND R. N. ZARE[b]

[a] Harvard University, Applied Physics, Harvard Paulson School of Engineering and Applied Sciences, 29 Oxford Street, Cambridge, MA 02189, USA
[b] Department of Chemistry, Stanford University, 333 Campus Drive, Stanford, CA 94305, USA
*Emails: SShankar@seas.harvard.edu; SShankar@Material-Alchemy.Org

20.1 Introduction

All the structures or devices that affect humans, from a small microprocessor to large airplanes and bridges, have underlying materials and chemicals that enable their forms and functionalities. All chemical and material properties, and their responses to external conditions, are based on chemical composition and structural characteristics. Systematic characterization of materials consists of the following four components: **experiments, theory, data,** and **computations**. Given the complex nature of inorganic and organic materials, all of these specific components are

necessary for a fundamental understanding that can help the design of materials for human needs. These components are illustrated in Figure 20.1 as distinct, but interconnected, which will help in framing this chapter. We will now briefly summarize each of the components with specific examples.

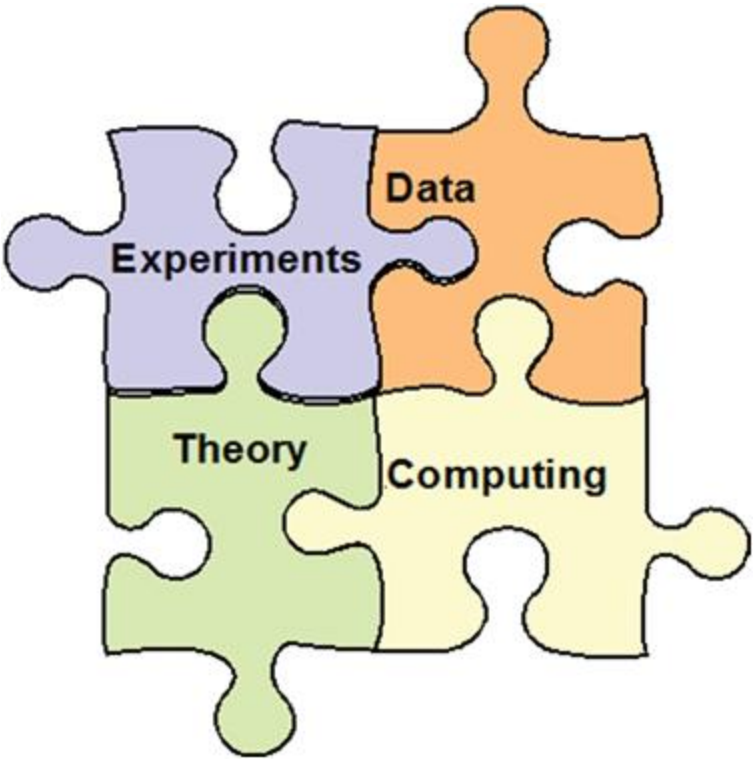


Figure 20.1 Four components for fundamental characterization of chemicals and materials.

Experiments refer to physical measurements that are used to quantify specific attributes of materials under specific conditions. For example, spectroscopic methods are used to measure chemical bonding and composition of chemicals and materials; X-ray diffraction measurements reveal crystallographic information, secondary-ion mass spectrometry is used for estimating the composition of solid surfaces and thin films, laser-induced fluorescence is used for detecting chemical species and energetics of excited states, calorimetry is used to measure thermal characteristics, *etc.* In an industrial setting, experiments are used to measure responses of chemicals and materials to external fields or conditions, as in four-point probes for electrical resistivities, thermocouples to measure temperatures in

processing equipment, tensile measurements to gage mechanical responses, *etc.*

Theory refers to fundamental understanding of physical and chemical characteristics of chemicals and materials. Examples include Schrodinger equation formalism for quantum mechanical description of electronic structure of matter, classical equations of motion for dynamics of particles, constitutive laws for mechanical properties, transition state theory for chemical reaction barriers, thermodynamic laws for ensembles of systems associated with heat, *etc.* Theory provides a basis for linking both computer simulations and experimental measurements, and by its definition is also extrapolative, an important point that we will revisit during the discussion of the guiding principles.

Computations, similar to physical experimental measurements (*in-silico* experiments), refer to simulated experiments done in computers. Examples of computational methods include the use of integration procedures for differential equations, Monte Carlo simulations for probabilistic processes, thermodynamic methods for phase equilibrium, many-body methods for excited-state properties, Green's function methods for inhomogeneous and non-equilibrium attributes, statistical and empirical methods for correlations, *etc.* Computational simulations may include combining one or more of the techniques with other components. For example, Density Functional Theory (DFT), a simplification of the *theoretical* Schrodinger equation formalism, can be used to *compute* ground-state energies.

Data is the fourth component and is considered critical to transfer and linking of information between the other components. For example, data based on chemical composition from spectroscopic measurements are used to build chemical structures of materials in computers or to establish the rate of chemical reactions. Using appropriate theoretical and numerical formalisms, these chemical structures may be simulated (using computers) to understand dynamics under different conditions than those measured experimentally. Simulation of protein folding is an example in which measured crystallographic structures of proteins are used to estimate long-term dynamics of protein behavior. Data can also be used to develop theory. An example of this is the blackbody radiation law developed by Planck using experimental spectroscopic data of the solar spectrum and building on existing correlations.

For a complete understanding of realistic chemical and material systems, the criticality of combining all these components can be illustrated with an example from plasma processing. Plasma-based processing is one of the most critical steps in semiconductor manufacturing in very large-scale integration of solid-state circuits, the basis of all present-day computers. As plasmas are “cold, but reactive” systems with electrons at high energies (10 000 K) compared to the energies of other species like ions, molecules, and chemical specific (greater than 300 K), most of the active chemical processes in plasmas are in non-equilibrium. Understanding these non-equilibrium interactions are necessary for the control of these processes in semiconductors. One of the successful efforts undertaken by a consortium of semiconductor companies in the 1990s quantified different aspects of plasma chemistry for perfluorinated hydrocarbons systematically.[1–4]Figure 20.2 illustrates examples of the above four components as applied to plasma processing. To address the complexity of the non-equilibrium chemistry, the components involved combining measurements (*e.g.* Ion Fourier Transform Spectrometry, film thickness), theory (*e.g.* Binary Encounter Bethe Model, Binary-Encounter-Dipole Model), computations (*e.g.* Variational Green's Function methods, plasma dynamics, gas phase and surface chemistry calculations), and data (*e.g.* electron collision cross sections, ion reaction rates, etching rates).

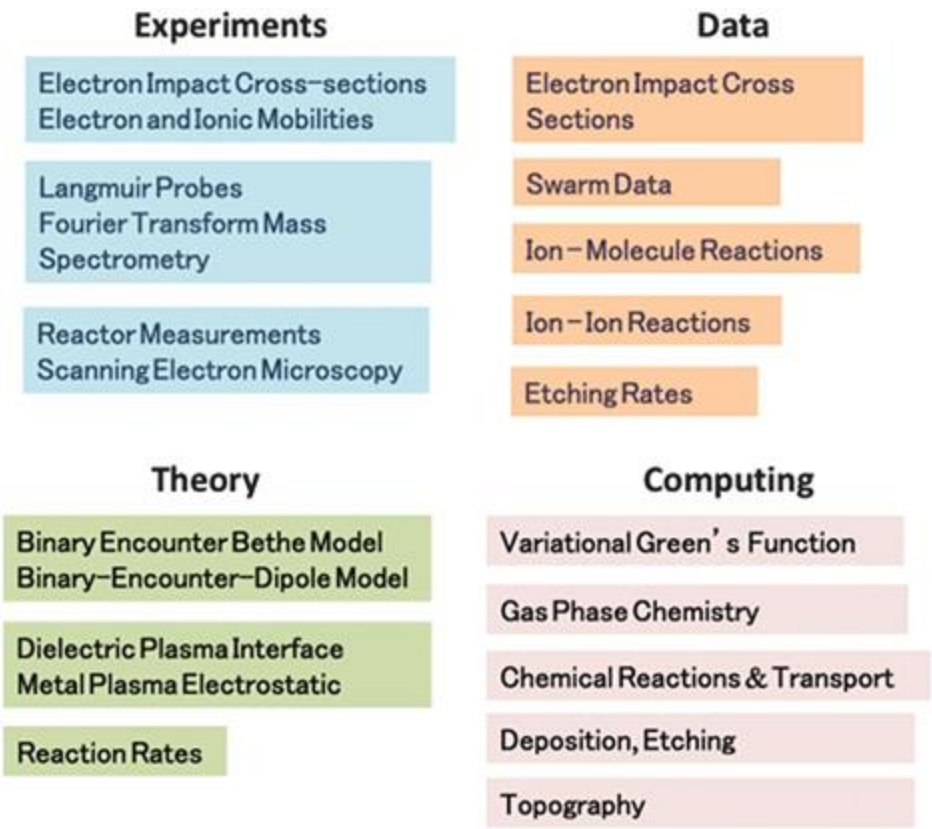


Figure 20.2 Four components for fundamental chemical or material characterization illustrated for plasma processing.

20.2 Guiding Principles for Applications of Machine Learning

One set of methods, collectively known as Machine Learning (ML) or Artificial Intelligence (AI)-based methods, which are based on prior data, have recently become a powerful approach for both the characterization and design of materials in biology, chemistry, and material science.[5–8] ML models are approximations that develop and quantify correlations between input data ("descriptor") and an output signal ("observation"). These approximations are generally *learned* from a "training set", before the accuracy of the model is evaluated by using a separate "test set". These methods, based on numerical and statistical methods to analyze

experimental data and computer simulations, have accelerated in the past decade due to a variety of factors: 1) Availability of powerful and widely available hardware enabled by Moore's law;[9] 2) Availability of large quantities of training data from high-throughput experimental measurements and computer simulations; 3) Advances in statistical methods and computational algorithms for analysis of data;[10] 4) Wider availability of data, hardware, and software brought about by economical and ubiquitous connectivity.

With the confluence of the above events, ML methods have been used extensively in a wide variety of fields in natural and physical sciences and engineering, including biology, chemistry, physics, chemical engineering, material science, genetics and physiology. ML methods fundamentally consist of formulating an appropriate representation of the data that identifies the features most important for a given application. The data could represent geometrical, structural features, or chemical features, as discrete values or analog functions. They are typically transferred from their native form to digital data, which are then transformed to finite dimensional vectors or some metric of the vectors (*e.g.* Euclidean distances). The key is to extract information with high statistical certainty, which forms a basis representing the underlying relationships. This is done by means of kernels, or by using other non-linear mapping methods. To satisfy these key requirements, an appropriate feature extractor needs to be developed that maps the original data into optimal representations. Once these representations are identified, a model is used to identify and separate patterns of data. Another set of methods that are being widely used are called Artificial Neural Networks, in which several nodes are interconnected as in a graph emulating the behavior of interconnected neurons in a brain.[11] Each of the nodes is activated through weights of the edges and a non-linear local switching function. The main advantage of these latter techniques is their ability to automatically encode complex input–out relationships. On the other hand, the underlying information on physical or causal relationships is lost in this approach. In addition to the references mentioned above, there are several review papers and special editions of journals related to applications of these techniques to specific problems in chemistry and materials science.[12,13]

The main purpose of all these ML methods is to systematically reduce dimensions of the original problem by using transformations, without losing the key information that is necessary to reproduce the key chemical or material characteristics consistently. Although these techniques are very powerful, they can be misleading in applications to natural and physical sciences for a variety of reasons: data are relatively sparse in scientific applications compared with pattern recognition applications used in multi-media and consumer applications; the physical and natural laws are complex and need to be self-consistent at any point in time; external surrounding effects and boundary conditions can influence material and chemical behaviors. Given the widespread use of these techniques as mostly black box methods, we have developed a few specific guiding principles that will help other practitioners in cautious application of these techniques. Each of these guiding principles is stated and illustrated with specific examples taken from literature and our own empirical observations from both academic and industrial settings. These examples are neither meant to be comprehensive reviews of all possible ML methods nor to be detailed recommendations of the application of any specific ML method to chemical and materials science problems. In addition, as will be seen below, each of these examples could be associated with more than one guiding principle. Our focus is to use the examples from biology, chemistry, physics, statistics, and engineering to emphasize the need for caution in the broad use of ML to these fields.

Guiding Principle 1: *Use ML for Interpolation but with Care for Extrapolation*

This principle enumerates the point that, although ML methods are very effective in their ability to interpolate in the ranges in which these models are trained, caution is necessary during extrapolation outside the range of the training set. Unlike the theory that describes the physical phenomena in consideration and its applicability, the training data are determined by the limits of available data, including the specific properties or characteristics, the composition and structural ranges, the conditions of experimental measurement (or simulations), and the spatial/temporal scales. Any ML model, which has been trained and characterized statistically, is applicable

within these limits. However, extrapolation outside these limits may provide misleading conclusions, which we will illustrate with our example of the formulation of the periodic table of elements.

One of the most successful applications of classification is the periodic table, celebrated recently by scientists and engineers in 2019 as the "International Year of the Periodic Table" of chemical elements. Formulation of the table is attributed to Dmitri Mendeleev, who demonstrated in 1869 that the chemical elements then known, when arranged by their respective atomic weights, demonstrated periodicity.[14] See Figure 20.3 for Mendeleev's initial published version of his table. Although patterns in the behavior of chemical elements have been observed before, we will use the example of Mendeleev's periodic table to emphasize the difference between interpolation and extrapolation. This is because Mendeleev made many successful predictions of new elements and their chemical natures in addition to correcting known atomic weights of existing materials. The atomic weight could be viewed as a "descriptor" as defined in the ML literature. A few of the key points that Mendeleev used to classify elements are summarized below, and are discussed in more detail in his papers[14,15] and elsewhere:[16]

1. *Elements exhibit periodicity of "properties" when arranged according to the magnitude of their atomic weights*;
2. *The atomic weights of chemically analogous elements either have similar values or increase in a uniform fashion;*
3. *The arrangement according to the atomic weight corresponds to the valencies of the elements and, to a certain degree, to differences in their chemical behavior, e.g. Li, Be, B, C, N, O, F;*
4. *The magnitude of the atomic weight determines the properties of the element;*
5. *It allows one to foresee the discovery of many new elements, e.g. analogs of Si and Al with atomic weights between 65 and 75;*
6. *It is to be expected that some atomic weights will require correction, e.g. Te cannot have an atomic weight of 128, but rather one between 123 and 126;*
7. *The above table suggests new analogies between elements;*

Series.	Group I. R_2O.	Group II. RO.	Group III. R_2O_3.	Group IV. RH_4. RO_2.	Group V. RH_3. R_2O_5.	Group VI. RH_2. RO_3.	Group VII. RH. R_2O_7.	Group VIII. RO_4.
1	H=1							
2	Li=7	Be=9.4	B=11	C=12	N=14	O=16	F=19	
3	Na=23	Mg=24	Al=27.3	Si=28	P=31	S=32	Cl=35.5	
4	K=39	Ca=40	—=44	Ti=48	V=51	Cr=52	Mn=55	Fe=56, Co=59 Ni=59, Cu=63
5	(Cu=63)	Zn=65	—=68	—=72	As=75	Se=78	Br=80	
6	Rb=85	Sr=87	? Y=88	Zr=90	Nb=94	Mo=96	—=100	Ru=194, Rh=104 Pd=106, Ag=108
7	(Ag=108)	Cd=112	In=113	Sn=118	Sb=122	Te=125	I=127	
8	Cs=133	Ba=137	? Di=138	? Ce=140	....	....	....	....
9	....	....	....	....	....	....	....	....
10	....	....	? Er=178	? La=180	Ta=182	W=184	....	Os=195, In=197 Pt=198, Au=199
11	(Au=199)	Hg=200	Tl=204	Pb=207	Bi=208	....	....	
12	....	....	....	Th=231	....	U=240	....	....

Figure 20.3 Mendeleev's Original Periodic Table.

There are many key points that can be seen in Mendeleev's "general deductions" as a forerunner to the current pattern recognition using ML methods. His prediction of classifying elements into groups based on atomic weights, possible existence of newer elements, and correction to the existing atomic weights were all validated by empirical observations. Mendeleev was able to establish the differences between physical properties (*e.g.* atomic weights) and chemical properties that determine how elements combine with other elements (*e.g.* acid or base formation). Even today, the periodic table is a tool for classification, description, analysis, and prediction, as the ML methods purport to do. Although Mendeleev used this formalism to analyze similarities in chemical reactions and his classification seemed to explain a few of the key observations, he was aware of the dangers of overly extending this system to chemical reactions as expressed in his words:[14] "*Because of the diversity of the relations existing between the simple substances, one cannot think of representing their system in the form of a continuous series, since the mutual relations of the substances are extraordinarily***diverse**…". We have highlighted the word "diverse" to make the point that chemistry is complex in the number of possibilities and

hence any interpolation built on specific data can be limiting in its application.

Based on this classification, Mendeleev predicted the existence of several unknown elements through the interpolation of atomic weights. An example of the success of this method is evident in his prediction of properties of eka-silicon, which was later discovered and named as germanium[16] (see Table 20.1).

Table 20.1 Comparisons between predicted properties of Eka silicon and Germanium. Adapted from ref. 16 with permission from Oxford Publishing Limited, Copyright 2007.

Property	Ekasilicon	Germanium
Atomic mass	72	72.61
Density	5.5	5.35
Melting point (°C)	High	947
Color	Gray	Gray
Oxide type	Refractory oxide	Refractory dioxide
Oxide density	4.7	4.7

Mendeleev further made many predictions using atomic weight as the classifying parameter. His estimate of atomic weights together with the successful (*e.g.* eka-boron or scandium with atomic weight 44) and unsuccessful predictions (*e.g.* coronium with atomic weight 0.4) are given in Table 20.2. Several interesting observations are evident from these predictions, two of which show the limitations of this approach: 1) Values of some atomic weights are estimated to be less than one (or lighter than hydrogen); 2) The success of the prediction rate is about 50%. As atomic weight was used as the *only* key variable to classify the elements based on the known data in the late nineteenth century, this was a remarkable success rate. However, it also shows the difficulty of trying to extrapolate based on known patterns and behaviors of complex systems, representing the field of chemistry, in a simple graphical form. Replacement of the atomic weight by atomic number and incorporation of the principles of quantum mechanics in the twentieth century led to a more accurate way of classifying elements. New theory, experimental measurements, and computations provided a

more detailed and consistent classification. As mentioned before, Mendeleev himself was aware of the limitations during the formulation, but became more confident as a result of the earlier validation, including the discovery of new elements (*e.g.* discovery of germanium in 1886), and tried to extrapolate broadly. Extrapolation of the known classification in itself was limiting, as evident from Mendeleev's predictions (including atomic weights less than 1). This limited its application within chemistry, where the complexity restricts description by simple correlations. Mendeleev's approach stands as a red-flag warning to investigators when applying scientific principles to extrapolate outside the proper context.

Table 20.2 Mendeleev's successful and unsuccessful predictions. Adapted from ref. 16 with permission from Oxford Publishing Limited, Copyright 2007.

Mendeleev's element	Predicted atomic weight	Experimental atomic weight	Current element
Coronium	0.4	Not found	Not found
Ether	0.17	Not found	Not found
Eka-boron	44	44.6	Scandium
Eka-cerium	54	Not found	Not found
Eka-aluminum	68	69.2	Gallium
Eka-silicon	72	72	Germanium
Eka-manganese	100	99	Technetium
Eka-molybdenum	140	Not found	Not found
Eka-niobium	146	Not found	Not found
Eka-cadmium	155	Not found	Not found
Eka-iodine	170	Not found	Not found
Eka-cesium	175	Not found	Not found
Tri-manganese	190	186	Rhenium
Dvi-teurium	212	210	Polonium
Dvi-cesium	220	223	Francium
Eka-tantalum	235	231	Protactinium

Guiding Principle 2: *Ensure Consistency Between Sources of Data Used in the ML Development and the Targeted Application*

It is important that the measurements used in building the model or the theory used in simulating specific properties are consistent with the end applications. Since ML models as they exist today do not have any intelligence beyond the training data on which they are based, this principle is intuitive, and yet it can be misapplied, as we will demonstrate in the example below.

In the microelectronics industry, new materials are integrated in electronic devices for advancing Moore's law. Integration of heterogeneous materials leads to many possible interfaces between these materials. Characterizing these interfaces is critical to ensure that the integrated devices function without failures. One specific attribute of these interfaces is adhesion. Adhesive strength depends on the intrinsic interface chemical bonds and the extrinsic processing conditions under which these interfaces were formed. As a result, detailed characterization of composition, interface morphology, and structure are critical for interfaces but this is difficult and time-consuming due to the buried nature of these surfaces and the multiple surfaces that make up a device. To address this problem, several methods for estimating adhesion between copper and other materials were evaluated by Kong *et al.*[17] A DFT model for estimating interface adhesion energy was compared with the use of an ML method based on experimental measurements of wetting angle for each interface, to compare and contrast the methods in estimating the adhesion. The DFT-based simulations were done at low temperatures (0K), where only the electronic degrees of freedom are taken into account. The wetting experiment is at a higher temperature as the material is heated beyond its melting point, and implicitly includes a phase transition, and the resulting structural and chemical changes. The comparisons are given in Figure 20.4.[17]

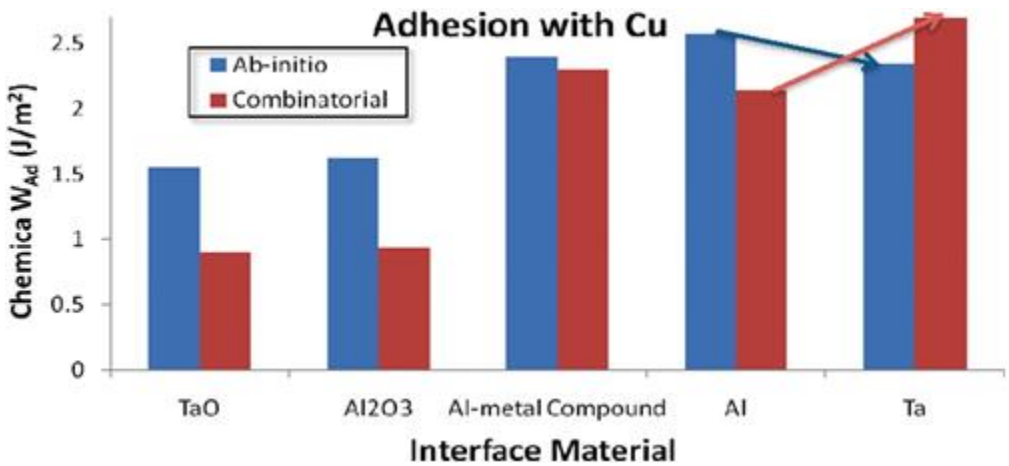


Figure 20.4 Comparisons between quantum-based simulations and ML based on experimental measurements.

Overall, both ways of estimating adhesion are mostly accurate and consistent within the bounds of their applicability. As wetting is measured after increasing the temperature, it is likely that in reality, materials form complex chemical compounds or phases as determined by phase equilibria and kinetics, unlike *ab initio* methods in which the materials are relatively pure and the calculations are done for a temperature of 0 K. Because of the nature of the wetting measurements, the experimental data are expected to capture the realistic phases and chemical nature of the interfaces. Because of the gap between the model assumptions and the collected experimental data, it is likely that oxidation during the process of wetting transforms the interface structures and hence the models make divergent predictions. This can be seen by comparing the phase diagram of Cu–Al (Figure 20.5) with that of Cu–Ta. Copper and tantalum are immiscible under the conditions of measurement and hence are closer to the ideal surface approximations of DFT. Contrast this with Cu–Al, which forms a complex phase equilibrium including chemical compounds, which are possibly captured by wetting experiments. Hence models that are based on single-crystalline interfaces will be inaccurate, as seen from Figure 20.4.

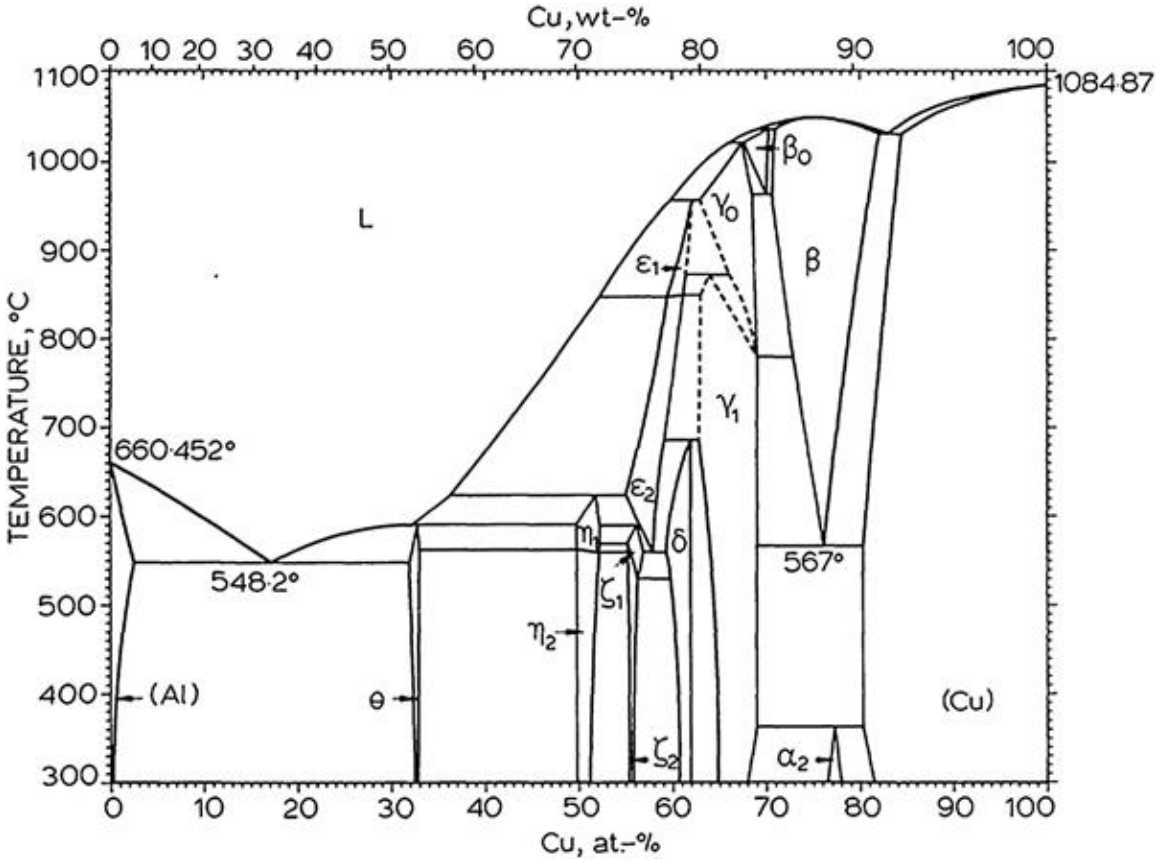


Figure 20.5 Complex phase diagram of Cu–Al. Reproduced from ref. 38 with permission from Taylor and Francis, Copyright 1985.

One of the other applications in which one of us applied ML methods in the late 1990s (at that time known as reduced-order models or heuristics) was for advanced process control. As semiconductor technology was advancing towards smaller feature dimensions, the processes used for depositing-processing-growing thin films, and testing devices were also increasing in complexity. One of the biggest hurdles for efficient manufacturing included the need for manual process optimization before and after equipment maintenance. To address these difficulties, a team of engineers across all of the Intel manufacturing fabrication plants developed advanced process control systems, in which experimental data were used to train physically-based models and integrate them in the station controllers attached to the process equipment. The models based on chemical reactions and diffusion were trained using data from equipment inputs and the

resulting process outputs. These led to elimination of misprocessing, and by middle 2000s, over forty applications were demonstrated during manufacturing.[18,19] The success of these models was largely due to the carefully controlled experimentation in manufacturing process equipment, which underscores the need for conducting measurements to train models that are consistent with the desired application conditions.

Guiding Principle 3: *Correlation is not Causation*

This is one of the most fundamental principles in tying empirical observations to underlying causes. Scientific discoveries need to differentiate between observed correlations and the underlying causation.[20] The main difference between the two is determined by time separation of a cause leading to an effect. It is possible for multiple effects to have the same underlying cause or for multiple causes to lead to the same effect. Correlations are statistical associations of data from a given distribution and are relatively valid as long as the external and boundary conditions remain the same. Causation, on the other hand, is predictive and is valid in both static and changing conditions and is not constrained by the statistical distributions.[21] Correlation is statistical and does not indicate the modulation of the distribution function if the conditions in which the data collected were to change due to interventions. As a result, it is difficult to differentiate between causes and effects using only correlational analysis. Although most ML methods traditionally use statistically significant correlations to establish relationships, they are unable to establish causal relationships. To illustrate this we will go to biological systems and try to draw similarities for chemical and material systems.

In evolutionary biology, as the existing conditions are used to trace the causes of these events, understanding the differences between cause and effect is critical. One specific illustration of this is given by de Duve[22] in listing different mechanisms of singularity, which is defined as an occurrence of a singular outcome or event. de Duve specifies six different causes that can lead to the same outcome. These are 1) *Deterministic Necessity* (deterministic laws of nature leading to a singular outcome for every singular cause); 2) *Selective Bottleneck* (externally imposed constraints leading to a singular outcome); 3) *Restrictive Bottleneck*

(internally imposed constraints evolving to a singular outcome); 4) *Pseudo-Bottleneck* (time-dependent progressive attrition of all other outcomes except one); 5) *Frozen Accident* (rare statistical events leading to a singular outcome); 6) *Fantastic Luck* (singular or a highly improbable event leading to a singular outcome). Using the same premise, we can list similar mechanisms for complexity in cause-effect relationships in chemistry and materials. In each of these mechanisms, the outcome is a specific attribute of material that is experimentally measured or simulated, as given below.

1. **Controlled measurements**: The measurement is done under tightly controlled conditions, and the resulting attribute is measured. The key point to note is that the measurement is isolated well enough that the specific property or attribute is controlled. Examples include tensile measurements for elastic moduli, spectroscopic measurements for chemical composition, and low-temperature measurement for electronic properties.
2. **Internal and external interactive effects**: The components of the system interact with one another and also with the environment. Examples include heterogeneous phase equilibrium of materials, and thin film properties modulated by the substrate on which they are measured.
3. **Interactions across length scales**: The different scales interact resulting in the measured property or attribute. These specific interactions are functions of the internal constraints such as composition, sample size, *etc.* Examples include grain growth, which in turn is dependent on the processing or sample size and preparation conditions, which determines the mesoscopic morphology of materials resulting in measured properties of polycrystalline materials.
4. **Interactions across time scales:** Examples include the use of equilibrium properties in lieu of time-dependent kinetics, resulting in the same chemical composition only under narrow conditions.
5. **Rare event effects**: This could result from relatively rare events affecting the property that is being measured. Examples include contamination of the measuring instruments or from errors in software used in calibration or computing. This is especially applicable for highly sensitive measurements or large and complex calculations.

6. **External errors:** This results from equipment set-up or human errors including confirmation biases used in the analysis. Examples include equipment set-up or when there are multiple steps in deconvolving and uncoupling complex properties.

These six mechanisms illustrate the principle that it is difficult to correlate an outcome with its corresponding causation. The focus of this principle is to try to eliminate the alternate mechanisms so that the correlation has a higher probability of being tied to causation. The example of the periodic table formulation illustrated in one of the previous principles also suffered from tying correlations strongly to causation. Although the original periodic table was established based on the correlations of atomic weights with physical and chemical properties, the modern periodic table incorporates the subsequent scientific advances based on the underlying atomic numbers and electronic structures.[23] Thus, using quantum mechanics, the periodic table has been expanded to link with the principles behind the causation.

Guiding Principle 4: *Optimize Information Extraction when Using ML*

This principle is fundamental to the intent of ML in building patterns based on data. As the patterns are expected to reveal the underlying relationships, ML is about quantifying information about the system itself. By information, we refer to the underlying causal relationships, which may include physical and natural laws, conservation laws, symmetries, thermodynamics, or other constitutive relationships. As these are not easy to formulate, compute, or measure for complex chemical and material systems, using data to identify patterns is a powerful alternative, within the range of applicability of the data.

Identifying the appropriate laws for all systems are due to the following: complex interactions, isolation of systems of interest, and quantification of many-body/many-scale effects. The scope of the specific material or chemistry being addressed by physical laws is limited when applying to a real system and hence needs simplifications without trading-off accuracy. This becomes even more difficult when chemical reactions are involved, as

the atomic and electronic movements are out of local equilibrium. For example, the accurate many-body models for chemistry scale as $O(n^3\text{-}n^8)$, where n is the number of particles.[24] Most real-world applications consist of more than a single molecule and may contain $O(10^{23})$ molecules. Hence, when applied to a real chemical or material system, the accuracy is traded off with the reduction of complexity of the chemistry or materials either in the number of atoms or electrons or in reducing the time scales. This is qualitatively illustrated in the Figure 20.6 where we have plotted the data requirement from empirical observations against the information (or prediction accuracy) obtained from the fundamental laws. Moving from left to right, the complexity is addressed by the use of more empirical data using less predictive theory, computations, or measurements. As we go from left to right, boxes go from white (representing a fully predictive model) to ML-based models shown as black boxes (representing all empiricism). We will illustrate an example of this guiding principle below, in which ML methods were used to estimate electronic correlation energies using other quantum chemistry models.

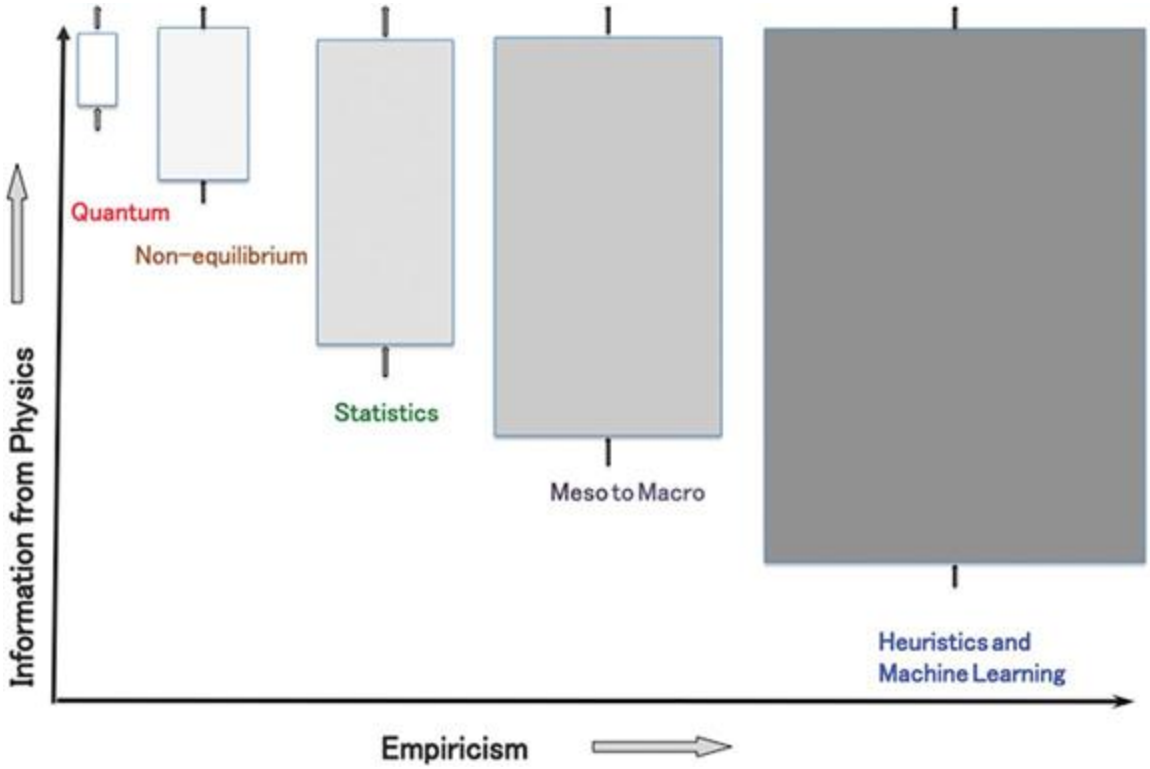


Figure 20.6 Information *vs* for different models compared with ML, indicating how information from physical theories is traded off with empirical data.

Hartree-Fock (HF) and DFT models are typically used to compute ground-state energetics of atoms, molecules, and materials in the solid state. Both these methods find approximations to the Schrodinger equation, which is a many-body, three-dimensional time-dependent formalism for electronic/particle wave functions. The practical difficulties in application of these models are twofold: 1) In addressing the so-called electronic correlation energies for higher prediction accuracies; and 2) In scaling up the computational problem to larger numbers of electrons and atoms, without sacrificing accuracy. The errors result from treating the interaction of one electron with another as a function of a smoothed and averaged electron density. To address the second difficulty, many approximations, including those based on ML, are made to represent correlations. Brockherde *et al.*[25] used ML to develop methods addressing the limitations of DFT, while Miller and co-workers[26,27] used machine learning to build more accurate models of electronic correlation energy based on self-consistent HF methods. We will use the latter as an example of how ML was used to optimize information extraction as related to correlation energy.

In this example, Miller and co-workers[26,27] used ML to build models for increasing the "chemical" accuracy of predictability for quantum models. This accuracy amounts to total energies within ∼1 kcal mol^{-1} (∼4 kJ $mole^{-1}$).[24] The error introduced due to the use of HF methods can affect those cases in which chemical bond breaking is important. To address this, the researchers used features based on molecular orbital properties, such as Fock, Coulomb, and exchange matrix elements. These addressed both the challenges in trading off predictability with the size of descriptor set. The models based on ML were compared with the more accurate post-Hartree–Fock methods (Coupled-Cluster, Moller–Plesset perturbation theory). The scalability and transferability of the ML-based models were demonstrated by further refining the techniques by optimizing the feature set and extending it to thermal conditions, and tested on 7211 organic molecules.[27] Their success in optimization of information extraction is evident, as they needed fewer molecular geometries and computations to reach the same accuracy. This work illustrates the guiding principle, where optimal use of

ML methods can circumvent the slow scaling of more accurate quantum chemistry models, without trading off accuracy.

Guiding Principle 5: *Combine Different Methods Consistently, Including Experiments, Theory, and Simulations, to Provide a Larger Window of Applicability*

This principle lays out the need for combining all the four components consistently for an integrated solution. Combining experiments, theory, computing, and data ensures self-consistent and systematic analysis. This, in turn, ensures predictability of the analysis and a wider range of applicability. This principle is in alignment with some of the other principles (Principle 1 on limiting use of ML for extrapolation, Principle 3 on causation *vs.* correlation, and Principle 4 on optimizing information extraction). As indicated before, it is important that the theory and measurements used in building the model are consistent with the end applications for which ML is being used.

An illustrative example with the combination of experimental approaches, data, computing, and theory on altered cancer metabolism is given below. This example focused on a clinical application of clear cell renal cell carcinoma (ccRCC).[28] As the most common and lethal subtype of kidney cancer, ccRCC is currently diagnosed using intraoperative frozen section analysis during a partial nephrectomy. If a surgeon can be informed which tissue is to be excised and which is to be retained, this would greatly improve the chances of a successful operation.

Owing to the time consuming and unreliable nature of the analysis, this study determined if desorption electrospray ionization mass spectrometry imaging (DESI-MSI) combined with statistical ML methods can be used as an alternate molecular diagnostic and prognostic tool for delineating surgical margins. In this technique, the tissue sample is mounted on a flat surface that can be translated in the x and y directions. The surface is bombarded with charged microdroplets and the splash of yet smaller microdroplets enters a mass spectrometer, producing a two-dimensional chemical map having very rich content. In this case, the experiment involved DESI-MSI of 23 pairs of fresh-frozen benign and cancer tissue samples. Based on small metabolites, fatty acids, and lipids obtained from

DESI-MSI, a binary classifier was successfully trained with 85% accuracy per patient to learn the differences between benign and cancer tissue based on changes in the relative abundances of metabolites. As the training procedure returns a sparse model, the set of chemical compounds was highly interpretable and then used to experimentally identify which particular metabolites are either overexpressed or suppressed in cancer tissues compared with normal tissues. The link between the model features and identifiable metabolites is critical, because it gives confidence in the predictions and has led to novel insights into alterations in cancer metabolism, including the Krebs cycle, fatty acid, and amino acid pathways. For example, this methodology, which found the ratio of glucose to arachidonic acid as highly predictive (>70%) in distinguishing normal and cancer tissue, suggests a dependence on high glucose production and fatty acid breakdown for sustaining cancer growth.

Although the initial aim was to improve frozen section analysis, it can be extended into a ‘wider range of applicability’, namely, to study metabolic vulnerabilities in kidney cancer to help guide the development of new therapeutic drugs. It is also noteworthy that this example is in accordance with Principle 4 on optimizing information extraction, where the changes in kidney cancer metabolic usage are captured by differences in the relative abundances of metabolites from DESI-MSI.

20.3 Concluding and Cautionary Remarks

ML methods have been found to be very powerful in many consumer applications and effective in applications to solve many aspects of chemistry and materials as evidenced by the many publications mentioned previously. Our intent in this chapter is twofold: to frame the ML applications in the context of the four components of experiments, theory, computing, and data, and to offer cautionary perspectives on the use of ML methods. We have done this by offering five specific guiding principles and how they can be used for a systematic analysis.

We revisit the potential of ML to identify key correlations between different components: experiments, theory, computation, and data. Further,

we will illustrate with examples to indicate two different levels in which ML can be applied. This is illustrated in Figure 20.7a and b representing the two levels: micro or intra and macro or inter.

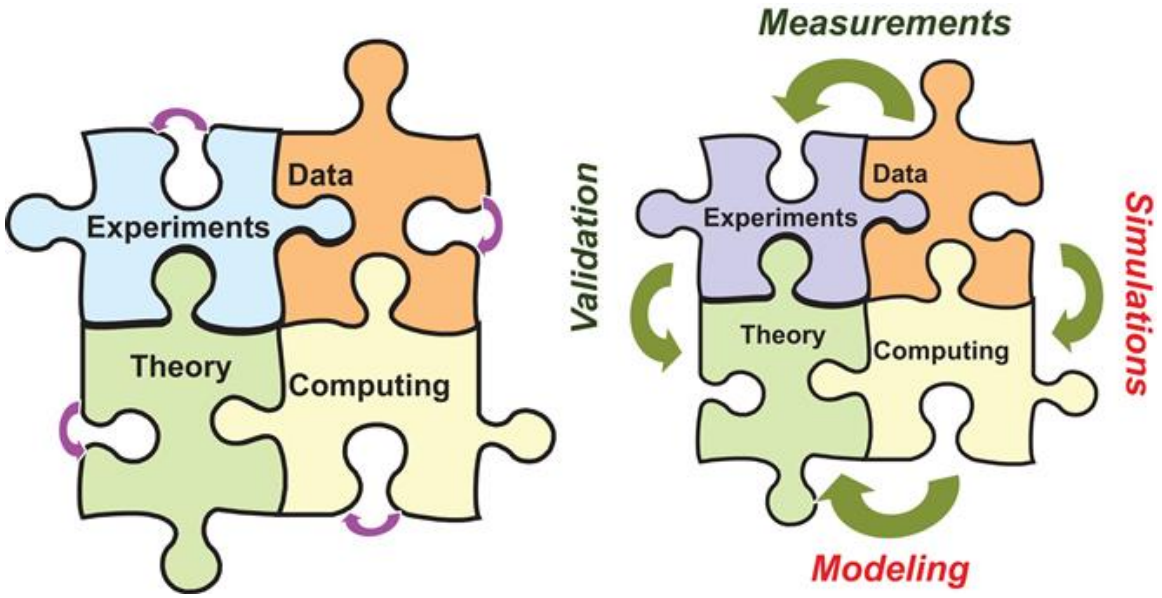


Figure 20.7 Application of ML at two different levels: (a) Micro/Intra: In the first level, ML can be used to accelerate each of the components; (b) Macro/Inter: In the second level, ML can be used to link between the components accelerating the overall analysis.

At the first level, using ML within the four components (*micro* or *intra* links) enables acceleration of the analysis. Examples include optimizing transition state analysis,[29] accelerating computations of ground state energetics,[25–27] ML used for learning interatomic potentials for atomistic simulations,[30] and ML trained on reaction data including failed or unsuccessful hydrothermal syntheses to predict reaction outcomes for the crystallization of vanadium selenites,[31] *etc.*

At the second level, linking between the different components enables acceleration of the overall analysis (*macro* or *inter* links). Each of the links between the components provides a specific role: **measurements** provide the link between experiments and data, **validation** acts as the link between experiments and theory, **modeling** is the link between computing and theory, while **simulations** provide a link between computing and data. Examples include linking between ML methods and experiments to drive

the analysis for metallic glasses,[32] experiments, computations, and data to control processes or accelerate analysis,[17–19] computations and data linked with ML to develop theoretical understanding of surface reactions,[33] using deep neural networks to gain insight into quantum mechanical-based theory of stability of molecules,[34] and linking of ML with flow reactors for automated optimization of diverse chemical reactions,[35] and by combining experiments, theory, data, and computing, Zare's team[28] demonstrating a clinical application for clear cell renal cell carcinoma, as elaborated in the previous section.

Our five guiding principles are intended to help with applying Machine Learning to physical sciences and are based on years of developing and applying methods to address problems in chemistry and material science. As these techniques become more widely adopted, our hope is that these principles will be used as safeguards during their application in solving problems in chemistry, engineering, and materials science. It is also important to note that these principles are not meant to be prescriptive but are guidelines for both scientists and the students who wish to apply ML to physical sciences. It also emphasized through these principles that ML is not a substitute to any of the four critical components (experiments, theory, computing, data) needed in fundamental understanding.

We illustrate this final overarching principle with the discussions between Dyson and Fermi on meson-proton scattering in 1953.[36] The intent of their collaboration was to compare Dyson's theoretical model and computations with Fermi's measurements. Although the comparisons were close, Fermi was critical of the arbitrary nature of the cut-off parameters. Fermi's opinion represented the prevalent understanding among the physicists that with four arbitrary parameters, one could fit an elephant and with five parameters, the elephant could be made to wiggle its trunk. Using five complex parameters, Mayer *et al.*[37] demonstrated that it was indeed the case. The key message is that validation of experimental data by parameters or methods without physical basis in itself is limiting, and there is a need for caution in the extrapolation of these methods. It is not the closeness of the model fit with data that is important, but rather the fundamental causation behind the physical phenomena for which the data have been collected. This example encapsulates our guiding principles and is

consistent with the limitation in application of frequency-based statistics to observations or data for a causal analysis. It is further illustrated by Dyson's own retrospective observations, roughly fifty years after the discussion with Fermi. In Dyson's own words:

> *The crucial discovery that made sense of the strong forces was the quark. Mesons and protons are little bags of quarks. Before Murray Gell-Mann discovered quarks, no theory of the strong forces could possibly have been adequate. Fermi knew nothing about quarks, and died before they were discovered. But somehow he knew that something essential was missing in the meson theories of the 1950s. His physical intuition told him that the pseudoscalar meson theory could not be right. And so it was Fermi's intuition, and not any discrepancy between theory and experiment, that saved me and my students from getting stuck in a blind alley.*

This exchange between Fermi and Dyson and the retrospective summary clearly illustrate the central premise of this chapter, namely, that correlations driven only by validation with data (either experimental or computational or both) can be misleading and hence should be augmented by systematic analysis based on physical principles and underlying theory. As we have indicated, ML methods are powerful tools to accelerate learning in each of the four components and in the development of the underlying theory, methods, and algorithms by linking between the components. In the absence of an underlying theory, ML can still be useful for interpolation, but should be applied with caution outside the domain on which ML was trained. It is our hope that the wide use of ML methods will be tempered by the proposed guiding principles in this work.

Acknowledgements

The authors would like to acknowledge the Camille and Henry Dreyfus Foundation for the opportunity to present and discuss a different perspective on the use of ML in chemical sciences and materials engineering. The chapter builds on previous joint projects with several researchers and

practitioners over the last two decades and discussions in several meetings including the Dreyfus Foundation ML meeting (2019), University of California Los Angeles, Institute of Pure and Applied Mathematics Workshop on Machine Learning and Many Particle Physics, (2016) and Stanford Class on Translation (2012). SS would like to acknowledge the support of the Hearst Foundation and the UCLA IPAM Fellowship. In addition, we acknowledge the information provided by Thomas Miller of Caltech, which was used to illustrate one of the guiding principles.

References

1. V. McKoy, C. Winstead and C.-H. Lee, *J. Vac. Sci. Technol., A*, 1998, **16**, 324.
2. W. L. Morgan, *Adv. At., Mol., Opt. Phys.*, 2000, **43**, 79.
3. S. Shankar, B. V. McKoy and W. L. Morgan, *Sixth U.S. National Congress on Computational Mechanics*, U.S. Association for Computational Mechanics, Dearborn, Michigan, 2001.
4. K. Yoshida, S. Goto, H. Tagashira, C. Winsted, B. V. McKoy and W. L. Morgan, *J. Appl. Phys.*, 2002, **91**, 2637.
5. H. M. Cartwright, *Using Artificial Intelligence in Chemistry and Biology: A Practical Guide*, CRC Press, Boca Raton, FL, 2008.
6. S. Shankar, Machine Learning for Materials Design: Combination of Theoretical methods, Heuristics, and Hybrid Techniques, *Workshop on Synergies between Machine Learning and Physical Models*, University of California-Los Angeles, Dec 5–9, 2016.
7. K. T. Butler, D. W. Davies, H. M. Cartwright, O. Isayev and A. Walsh, *Nature*, 2018, **559**, 547.
8. B. Sanchez-Lengeling and A. Aspuru-Guzik, *Science*, 2018, **361**(6400), 360.
9. https://www.intel.com/content/www/us/en/silicon-innovations/moores-law-technology.html; accessed Jan 31, 2020.
10. T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York City, USA, 2009.
11. Y. LeCun, Y. Bengio and G. Hinton, *Nature*, 2015, **521**, 436.

12. M. Rupp, *Int. J. Quantum Chem.*, 2015, **115**, 1003.
13. I. Tanaka, K. Rajan and C. Wolverton, *MRS Bull.*, 2018, **43**, 659.
14. D. I. Mendeleev, On the relationship of the properties of the elements to their atomic weights, in *Mendeleev on the Periodic Law*, ed. W. B. Jensen, Zeitschrift fur Chemie, Mineola, NY, 2002, pp. 405–406.
15. D. I. Mendeleev, On the periodic regularity of the chemical elements, *Mendeleev on the Periodic Law, Annalen der Chemieund Pharmacie*, 8(Suppl), ed. W. B. Jensen, 2002, pp. 133–229.
16. E. R. Scerri, *The Periodic Table: Its Story and Its Significance*, Oxford University Press, Oxford, UK, 2007.
17. C. S. Kong, M. Haverty, H. Simka, S. Shankar and K. Rajan, *Model. Simul. Mater. Sci. Eng*, 2017, **25**, 065014.
18. J. Luke, T. Albertson, Y. H. Lin, S. Shankar and D. Pantuso, *Intel. Test Assembly J.*, 2003, **6**, 481.
19. S. Shankar, K. Knutson and Y. H. Lin, *DOTS: Advanced Paradigm in Process Control – Yesterday, Today, and Tomorrow*, Intel Advanced Process Control (APC) Summit, Chandler, 2003.
20. J. Woodward, *Philos. Sci.*, 2014, **81**, 691.
21. J. Pearl, *Statist. Surv.*, 2009, **3**, 96.
22. C. de Duve, *Singularities*, Cambridge Univ. Press, New York, 2005.
23. L. N. Ross, *Synthese*, 2018, DOI: 10.1007/s11229-018-01982-0.
24. F. Jensen, *Introduction to Computational Chemistry*, John Wiley & Sons, 2nd edn, 2007.
25. F. Brockherde, L. Vogt, L. Li, M. E. Tuckerman, K. Burke and K.-R. Müller, *Nat. Commun.*, 2017, **8**, 872.
26. M. Welborn, L. Cheng and T. F. Miller III, *J. Chem. Theory Comput.*, 2018, **14**, 4772.
27. L. Cheng, M. Welborn, A. S. Christensen and T. F. Miller III, *J. Chem. Phys.*, 2019, **150**, 131103.
28. K. Vijayalakshmi, V. Shankar, R. M. Bain, R. Nolley, G. A. Sonn, C.-S. Kao, H. Zhao, R. Tibshirani, R. N. Zare and J. D. Brooks, Identification of Diagnostic Metabolic Signatures in Clear Cell Renal Cell Carcinoma Using Mass Spectrometry Imaging, *Int. J. Cancer*, 2020, **147**, 256.
29. Z. Pozun, K. Hansen, D. Sheppard, M. Rupp, K.-R. Muller and G. Henkelman, *J. Chem. Phys.*, 2012, **136**, 174101.

30. J. Behler, *J. Chem. Phys.*, 2016, **145**, 170901.
31. P. Raccuglia, K. C. Elbert, P. D. F. Adler, C. Falk, M. B. Wenny, A. Mollo, M. Zeller, S. A. Friedler, J. Schrier and A. J. Norquist, *Nature*, 2016, **533**, 73.
32. F. Ren, L. Ward, T. Williams, K. J. Laws, C. Wolverton, J. Hattrick-Simpers and A. Mehta, *Sci. Adv.*, 2018, **4**, eaaq1566.
33. Z. W. Ulissi, A. J. Medford, T. Bligaard and J. K. Nørskov, *Nat. Commun.*, 2017, **8**, 14621.
34. K. T. Schütt, F. Arbabzadah, S. Chmiela, K.-R. Müller and A. Tkatchenko, *Nat. Commun.*, 2017, **8**, 13890.
35. A.-C. Bédard, A. Adamo, K. C. Aroh, M. Grace Russell, A. A. Bedermann, J. Torosian, B. Yue, K. F. Jensen and T. F. Jamison, *Science*, 2018, **361**, 1220–1225.
36. F. Dyson, *Nature*, 2004, **427**, 297.
37. J. Mayer, K. Khairy and J. Howard, *Am. J. Phys.*, 2010, **78**, 648.
38. J. L. Murray, The aluminium-copper system, *Int. Metals Rev.*, 1985, **30**(1), 211.

Subject Index

absorber, 343–344
Accelerated Molecular Discovery (AMD), 137, 153, 160
accelerated property prediction, 289–290
Accelerating Therapeutics for Opportunities in Medicine (ATOM), 156
adaptive neuro-fuzzy inference system (ANFIS), 343, 344
adsorption, 344–345
adversarial spectroscopy, 475–479
Air Force Research Laboratory
    Autonomous Research System, 466, 467
AlexNet, 379
algorithms
    AlphaGo, 38, 144
    backpropagation, 175
    decision trees, 59–64
    deploying, 31–35
    estimator, in chemical engineering, 353–356
    evolutionary, 221, 222, 241, 301
    genetic, 241, 242, 247, 293, 321, 322, 325, 341, 356–357
    inpainting, 454, 455
    interpolation, for dynamic sampling, 458
    Lasso, 55
    Metropolis–Hastings, 199
    multi-objective genetic, 325
    phase correction, 91–93
    setting, 20–23

TensorFlow, 11
AlphaGo algorithm, 38, 144
AMDET (absorption, distribution, metabolism, excretion, and toxicity) properties, 252–255
American Chemical Society, 161
American Society for Metals (ASM), 312
amorphous TiO_xH_y, hydrogen evolution reaction, 502–505
antimalarial drug discovery, 398–420
applicability domain, 407–408
computational design, 413–419
dataset and molecular descriptors, 400–403
druglikeness, 419
ms-QSBER-EL model, 403–408
physicochemical and structural information, 408–413
application programming interfaces (API), 154, 156, 315
artificial intelligence (AI), 17, 38, 207, 340–341
definition of, 3–8, 35
energy management, modeling and separation of, 346–347
estimator algorithms, in chemical engineering, 353–356
Mechanical Sympathy
additional data, 127
challenges, 122–124
global warming, 124–126
network and learning, 129–130
noise in data, 126
odd information, 128–129
plenty of data, 127–128
trained AI, testing of, 120–122
for organic materials prediction, 280–302
accelerated property prediction, 289–290
building block libraries, 285
case studies, 297–302
data extraction, 283–285
de novo molecular design, of materials building blocks, 294–295

descriptors, 287–289
phase space exploration, 293–294
structure prediction, 290–293
synthetic accessibility, 296–297
training data, 285–287
powered chemical innovation, for defense applications, 136–162
automating synthetic planning, 137–147
self-driving laboratory, 159–162
solutions to data challenges, 153–159
statistical learning, data challenges for, 147–153
reactors, modeling and separation of, 345
separation units, modeling and separation of
absorber and desorber, 343–344
adsorption, 344–345
distillation columns, 342–343
whole plants, modeling and separation of, 345–346
artificial neural networks (ANNs), 16, 112, 113, 116–119, 174–175, 181, 341
demystifying, 398–420
inorganic materials discovery, 323
molecular and target data analysis, 263–264, 269, 273
polymerization process, 233–240
applications, types of, 234–236
types, 236–240
ASM Phase Diagram Database, 312
ASPIRE program, 154
AstraZeneca, 188
astrochemistry, chemical networks in, 195–204
Bayesian approach, 199–201
chemical reactions databases, completeness and reliability of
gas-phase networks, 197–198
grain-surface networks, 198–199
complex networks, machine learning techniques for, 202–203
AtomNet model, 385
atom-to-atom mapping, 171
Automatic-Flow for Materials Discovery (AFLOW), 313, 315, 333

automating synthetic planning, 137–147
adoption, 146–147
assessment, 145–146
known routes, optimizing, 138–139
reaction outcomes, prediction of, 140–142
Autonomous Research System (ARES), 466, 467
autonomous science, 450–482
data-limited training, generative adversarial networks for
adversarial spectroscopy, 475–479
image recognition applications, 471–475
linear analysis, 479–481
design of experiments, 453–470
active machine learning, 464–466
interpolation algorithms, for dynamic sampling, 458
materials synthesis, dynamic sampling for, 466–467
modular architectures, 467–470
noniterative dynamic sampling, 463–464
Raman hyperspectral imaging, SLADS for, 460–463
simplex approaches, 456–458
small data challenges, sparse sampling strategies for, 454–456
supervised learning approach for dynamic sampling, 458–460
X-ray diffraction imaging, SLADS for, 460–463

backpropagation algorithm, 175
balanced accuracy, 34
Bayesian approach to astrochemistry, 199–201
Bayesian program learning (BPL), 332
BenevolentAI, 38
bias, 114, 148–149
inductive, 381
bias-potential constrained Broyden dimer (BP-CBD) method, 496, 497
BindingDB, 155
Bing, 146
biomaterials
nanomaterials–biology interface, 215–220
bioreactors, nanomaterials–biology interface in, 220–221

Blizzard Entertainment, 12
boosting, 65
bootstrapping, 65
Born–Oppenheimer approximation, 77, 89
BP, 340
Buchwald–Hartwig reaction, 179, 180

CAS, 155, 185
CASD software, 170, 185, 187, 188, 190
CASP system, 143, 145–147
CASTEP, 426
cell therapies, nanomaterials–biology interface in, 220–221
Center for Drug Evaluation and Research (CDER), 161
ChEMBL, 9, 400
ChemDataExtractor (CDE), 114, 284, 285
chemical engineering, 340–361
    AI-based estimators, selection and applications of, 357–360
    control and operation
        chemical processes, controlling, 348–351
        process control, baseline of, 347–348
        traditional controllers with integrated machine learning, 351–352
    estimation and prediction
        AI estimator algorithms, 353–356
    future trends, 361
    modeling and simulation, 341–347
        separation units, 342–345
chemical fingerprints, 19
chemical processes, controlling, 348–351
chemical reactions databases, completeness and reliability of
    gas-phase networks, 197–198
    grain-surface networks, 198–199
chemical synthesis, machine learning for, 169–191
    methods, 174–179
    nature of data
        data sources, 170–173

molecular descriptors, 173–174
real-life applications, 188–189
synthetic route design, 179–188
reaction prediction, 186–187
reaction prediction, optimizing, 187–188
retrosynthesis, 183–186
ChemOS, 470
ChemPlanner™ (formerly ARChem Route Designer), 185, 189
Chemputer, 223
ChemTS library, 294
Citrine Informatics, 333
Clonal Selection (CS), 344
cluster resolution feature selection (CR-FS), 327
complex networks, machine learning techniques for, 202–203
Complex Organic Molecules (COMs), 196, 198
complex polymerization process, 227–248
artificial neural networks, 233–240
applications, 234–236
types, 236–240
difficulties in, 227–229
free-radical polymerization process, phenomenological modeling of, 229–233
methyl methacrylate free radical polymerization, 243–248
soft-computing hybrid configurations, 240–243
computational science
definition of, 1–3
computers
as scientists, 1–14
strengths and weaknesses of, 2
computer vision (CV), 147, 149, 151, 157
CoNLL Shared Tasks, 147
convolutional neural networks (CNNs), 289, 294, 385
core-loss spectrum, 424–445
database construction, 426
data-driven prediction and interpretation method, 427–430
decision tree, for material information, 427

ELNES/XANES, quantification of structures and properties from, 438–441
feed-forward neural network, 427
mono-metal oxides, O-K edges of, 431–434
noisy spectra, quantification of structures and properties from, 442–444
polymorphous SiO_2, O-K edges of, 434–438
spectral data clustering, 426–427
correlation, 522–523
Coulomb matrix, 84
covalent organic frameworks (COFs), 299
cross-validation (CV), 324–325
Crystallographic Open Database, 286
Crystal Structure Analysis by Particle Swarm Optimization (CALYPSO), 327
crystal structure prediction (CSP), 290–292

data augmentation (DA), 443
data cleaning, 317–319
data gathering, 17–20
data generation
and predictive models, closed-loop integration of, 158–159
through simulation, 157–158
data-limited training, generative adversarial networks for
adversarial spectroscopy, 475–479
image recognition applications, 471–475
linear analysis, 479–481
data mining, 320, 469
data pre-processing, 317–321
data reduction, 320–321
data representation, 316–317
data transformation, 319–320
decision trees, 58–65, 286, 322
algorithms, 59–64
ensembles, 64–65

information, 58–59
for material information, 427
deep learning (DL), 45, 255, 273, 289, 374, 385, 388
DeepMind, 12
AlphaGo algorithm, 38, 144
deep neural networks (DNNs), 118, 293–295
inorganic materials discovery, 323
deep reinforcement learning, 144–145, 351
Defense Advanced Research Projects Agency (DARPA), 162, 223
Accelerated Molecular Discovery, 137, 153, 160
Battlefield Medicine program, 137
Synergistic Discovery and Design (SD2) program, 155
defense applications, AI-powered chemical innovation for, 136–162
automating synthetic planning, 137–147
assessment, 145–146
known routes, optimizing, 138–139
reaction outcomes, prediction of, 140–142
retrosynthesis, 143–145
self-driving laboratory, 159–162
autonomous research, implications of, 160–162
solutions to data challenges, 153–159
data generation and predictive models, closed-loop integration of, 158–159
data generation through simulation, 157–158
experimental data collection, automation and standardization of, 154–155
leveraging experiments, 157
sharing data, across laboratories, 155–156
statistical learning, data challenges for, 147–153
adoption, 146–147
bias, 148–149
narrow collection of tasks, 150–151
standard data representation, 151–153
unreliable and non-standardized data, 149–150
de novo molecular design, of materials building blocks, 294–295

density functional theory (DFT), 157, 174, 211, 282, 286, 287, 290–292, 297, 298, 375, 495
heterogeneous catalysis, 489
inorganic materials discovery, 312–313
material property prediction, 329
physically inspired architectures, 390
representational learning, 377
structure prediction, 327, 328
Department of Defense (DoD), 136, 137, 147, 160, 162
design of experiments (DoE), 138, 293–294
autonomous, 453–470
active machine learning, 464–466
interpolation algorithms, for dynamic sampling, 458
materials synthesis, dynamic sampling for, 466–467
modular architectures, 467–470
noniterative dynamic sampling, 463–464
Raman hyperspectral imaging, SLADS for, 460–463
simplex approaches, 456–458
small data challenges, sparse sampling strategies for, 454–456
supervised learning approach for dynamic sampling, 458–460
X-ray diffraction imaging, SLADS for, 460–463
desorber, 343–344
Dice Similarity, 49–50
Differential Evolution (DE), 241
discretization, 320
distance-matrix based descriptors, 85
distillation columns, 342–343
DRAGON, 207, 288, 302, 402
druggability prediction, based on physico-chemical molecular properties, 256
dye-sensitized solar cell database, 286, 287
dynamic sampling
interpolation algorithms for, 458
for materials synthesis, 466–467
supervised learning approach for, 458–460

electron energy-loss near-edge structure (ELNES), 424–426, 429, 430, 444
    quantification of structures and properties from, 438–440
electronic lab notebooks (ELNs), 154, 172–173
Emerald Cloud Lab, 155
energy management, AI for modeling and separation of, 346–347
Environmental Protection Agency (EPA), 147
evolutionary algorithms (EA), 221, 222, 241, 301
evolutionary methods, nanomaterials–biology interface, 221–223
excited states, adaptive sampling for, 87–89
experimental data collection, automation and standardization of, 154–155
expert systems (ESs), 341
Exscientia, 38
extended connectivity fingerprints (ECFPs), 19, 20, 288, 376, 382
extended X-ray absorption fine structure (EXAFS), 426
extrapolation, 516–519
ExxonMobil, 341

Faber–Christensen–Huang–Lilienfeld (FCHL) representation, 80, 85–86
FactoMineR, 264
FAIR (Findable, Accessible, Interoperable, Reusable) Data Principles, 155, 156
FCHL (Faber–Christensen–Huang–Lilienfeld) representation, 80, 85–86
feed-forward (multilayer perceptron) neural networks, 236–237
    core-loss spectrum, 427
    heterogeneous catalysis, 491
Findable, Accessible, Interoperable, Reusable (FAIR) Data Principles, 155, 156
fingerprints, 46–47
    chemical, 19
    extended connectivity, 19, 20, 288, 376, 382
    Morgan, 173, 288, 298
flaws, overcoming, 27–31
Food and Drug Administration (FDA), 147, 162
    Center for Drug Evaluation and Research, 161
free-radical polymerization process, phenomenological modeling of, 229–233

functional organic materials, 301–302
fuzzy neural networks (FNN), 326

gas-phase networks, 197–198
Gaussian process (GP), 285–286, 323
Gaussian Process Regression (GPR), 292, 293, 302
Gaussian-type structure descriptors (GTSD), 492, 493
GE, 341
generative adversarial networks (GAN), 295, 299, 470
    for data-limited training
        adversarial spectroscopy, 475–479
        image recognition applications, 471–475
        linear analysis, 479–481
genetic algorithm (GA), 241, 242, 247, 293, 341, 356–357
    inorganic materials discovery, 321, 322
    multi-objective, 325
Google, 146, 156, 161
    Dataset Search, 113
    DeepMind, 12, 38
    Scholar, 113
gradient boosting decision tree (GBDT) method, 329
grain-surface networks, 198–199
grand canonical Monte Carlo (GCMC), 210, 211
graph convolutional (GC) method, 382–384, 387
graph models
    molecular representations, 381–384
graph neural networks (GNNs), 382, 384, 386
grid models
    molecular representations, 384–386
GTN Ltd, 38

hands-free execution of experiments, 155
Harvard Clean Energy Project, 285, 287
Harvard Organic Photovoltaic Dataset, 286, 287
heterogeneous catalysis, 488–509
    applications

amorphous TiO_xH_y, hydrogen evolution reaction, 502–505
PES exploration, 499–501
ultrasmall Au/CeO_2, 501–502
ZnCrO catalysts, syngas conversion on, 505–508
methods
data set generation, 494–497
high-dimensional NN architecture, 490–492
neural network training, 493–494
self-learning procedure, 497–499
SSW global optimization, 494–497
structural descriptors, 492–493
hierarchical clustering, 426–427
hierarchical machine learning (HML), 301–302
high-dimensional NN architecture (HDNN), 490–492
high throughput screening (HTS), 18, 450, 451
HSE, 341
humans, strengths and weaknesses of, 2
hyperplanes, 66

ICSYNTH, 185, 188
ImageNet, 147
imitation learning, 332
implantable cells, nanomaterials–biology interface in, 220–221
InfoChem, 185, 188
information extraction, optimizing, 524–525
Inorganic Crystal Structure Database (ICSD), 312, 331–332
inorganic materials discovery, 311–334
applications of
material property prediction, 329–331
structure prediction, 326–329
synthesis optimization, 325–326
challenges and opportunities
efficient data representation, generation of, 332
model accessibility, increasing, 333
model interpretability, enhancement of, 332–333

small data sets, 331–332
workflow for, 314–325
data collection, 315
data pre-processing, 317–321
data representation, 316–317
model construction, 321–324
model evaluation, 324–325
inpainting algorithm, 454, 455
interpolation, 516–519
interpolation algorithms, for dynamic sampling, 458
interstellar medium (ISM), 195–199, 202
inverse neural modeling (INN), 244
2-iodoxybenzoic acid (IBX), 139

Jordan–Elman neural networks, 238

kernel ridge regression (KRR), 80, 286, 289, 293, 323
kernels, 66–68
K-Fold Cross Validation (*K*-Fold CV), 178–179
Khazana, 286, 287
Khiops, 320
Kinetic Database for Astrochemistry (KIDA), 198
k-Nearest Neighbours (*k*NN), 52–54, 322
Kuhn Caret, 264

Lasso (east absolute shrinkage and selection operator) algorithm, 55
Lawrence Livermore National Laboratory (LLNL), 156
least square support vector machine (LSSVM), 325
linear discriminant analysis (LDA), 460
linear model (LR), 79–80
linear regression, 54–58
Logic and Heuristics Applied to Synthetic Analysis (LHASA), 170
logistic regression, 375
long short-term memory (LSTM), 13

machine learning (ML), 38–39

algorithm, deploying, 31–35
algorithm, setting, 20–23
challenges to, 206–208
data gathering, 17–20
definition of, 8–14
flaws, overcoming, 27–31
guiding principles for practical applications, 512–529
opportunities for, 206–208
questions, framing, 16–17
training procedure, 23–27
see also individual entries
machine-learnt potentials (MLP), 291
machine-readable format, recording data in, 154
Make-It program, 146, 147, 153
material property prediction, 329–331
Materials Genome Initiative (MGI), 313
MatErials Graph Network (MEGNet), 316, 333
Materials Platform for Data Science (MPDS), 312, 315
Materials Project (MP), 286, 313, 315, 333
Materials Project Nanoporous Database, 287
materials synthesis, dynamic sampling for, 466–467
Matminer, 316
Matthews correlation coefficient (MCC), 34
Max Planck Institute for Colloids and Interfaces, 223
Mechanical Sympathy, 109–133
challenges to, 110–111
history of, 109–110
problems and solutions
artificial intelligence, 119–129
databases, 114–115
loss function, 131
network and dataset problems, 132
network and learning, 129–130
network weights, 132–133
overfitting, 132
parameters, setting, 117–119

samples, 111–113
slow learning of network, 130–131
synthetic route prediction, 115
training, 116–117
MedChemInformatics, 37–72
curation, 40
dataset assembly, 40
Deep Learning, 45
fingerprints, 46–47
good descriptors, overtraining and characteristics of, 52
machine learning methods
Deep Learning and Random Forests, 58–65
k-Nearest Neighbours, 52–54
linear regression, 54–58
Support Vector Machine, 65–71
model building, 40–41
model quality and statistics, 50–52
coefficients of determination, 51–52
Kendall's Tau, 52
molecular descriptors, 46–47
molecule comparison
Dice Similarity, 49–50
Tanimoto Similarity, 47–48
Tversky Similarity, 48–49
Neural Nets, 45
“observables” and “features,” 45–47
outliers, 43–45
Principal Component Analysis, 41–42
QSAR modeling, activity range and distribution of activities for, 42
3D model, 42
message passing neural networks (MPNNs), 383–384
metal–organic frameworks (MOFs), 299–301
methylenimmonium cation
energy conservation, 96–97
ML surface hopping dynamics, 94–96
tools, 97–103

methyl methacrylate free radical polymerization, 243–248
Metropolis–Hastings (MH) algorithm, 199
Metropolis Monte Carlo (MC), 495
MilliporeSigma (formerly Sigma-Aldrich), 146
model-based iterative reconstruction (MBIR), 456
model predictive control (MPC), 348, 351, 352
molecular descriptors, 173–174
molecular drug delivery, 251–275
    data and druggability scoring, 257–259
    ML classification results, 266–269
    molecular parameters and targets, 259
    statistical and program approach, 259–266
molecular footprints, 375–377
molecular representations, 380–391
    graph models, 381–384
    grid models, 384–386
    physically inspired architectures, 388–391
    sequence models, 386–387
Molecular Transformer (MT), 184, 187
MoleculeNet, 289
mono-metal oxides, O-K edges of, 431–434
Monte Carlo Tree Search (MCTS), 144, 186, 293, 294, 299
Morgan fingerprints, 173, 288, 298
ms-QSBER-EL model, 399–401, 405–408
    building, 403–405
    computational design, 413–419
    internal and external performance, 407
    molecular descriptors, 406, 408–413
multi-layer feed-forward (MLFF) neural network, 82, 84
multi-layer perceptron–artificial neural network (MLP–ANN) method, 342–343
multi-layer perceptrons (MLPs), 375, 403
multi-objective genetic algorithm (MOGA), 325
multiple linear regression (MLR), 326
multi-step retrosynthesis, 185–186

named entity recognition (NER), 284, 285
nanomaterials–biology interface, 206–224
    biomaterials, 215–220
    bioreactors, 220–221
    cell therapies, 220–221
    evolutionary methods, 221–223
    implantable cells, 220–221
    importance of, 208–209
    materials, 209–211
    regenerative medicine, 215–220
    scale of, 208–209
    special issues with complex materials, 208–209
National Center for Translational Science (NCATS), 154
National Ignition Facility, Lawrence Livermore National Laboratory, 451
National Institutes for Standards and Technology, 137
National Institutes of Health (NIH), 137, 147
National Science Foundation, 161
natural language processing (NLP), 147, 149, 151, 157, 159, 284, 315
nearest neighbour model, 6–7
negative predictive value (NPV), 34
Neural Nets, 45
NeuralNetTools, 264
neural networks (NNs), 81–83, 97–103
Neural Turing machines (NTM), 332
NIMS Materials Database (MatNavi), 312, 315
noisy spectra, quantification of structures and properties from, 442–444
NOMAD, 286
nonadiabatic couplings (NACs), 77, 79, 84, 89, 91–94, 97
nonadiabatic molecular dynamics (NAMD) simulations, 76–104
    machine learning methods
        descriptors, 84
        distance-matrix based descriptors, 85
        FCHL representation, 85–86
        kernel ridge regression, 80
        linear model, 79–80
        neural networks, 81–83

support vector regression, 80–81
training process, 83–84
methylenimmonium cation
energy conservation, 96–97
ML surface hopping dynamics, 94–96
tools, 97–103
phase correction algorithm, 91–93
surface hopping dynamics, 93–94
training set generation
excited states, adaptive sampling for, 87–89
initial set, 86–87
wave function phases, 89–91
noniterative dynamic sampling, for autonomous DoE, 463–464
non-learned molecular representations, 374–378
computable features, 377–378
molecular footprints, 375–377
Nonlinear Iterative Partial Least Squares (NIPALS), 57

Open Quantum Materials Database (OQMD), 313, 315
Open Spectral Dataset, 286
Optical Structure Recognition Application (OSRA), 157
Ordinary Least Squares (OLS), 55–58
organic electronics, 297–299
Organic Materials Database, 286, 287
organic materials prediction, artificial intelligence for, 280–302
accelerated property prediction, 289–290
building block libraries, 285
case studies, 297–302
data extraction, 283–285
descriptors, 287–289
phase space exploration, 293–294
structure prediction, 290–293
synthetic accessibility, 296–297
training data, 285–287
orthogonal polynomial feed-forward neural network (OPFNN), 238

partial density of states (PDOS), 424
Partial Least Squares (PLS), 55–58
particle swarm optimization (PSO), 322
Patent and Trademark Office, 223
Pauling File. *see* Pearson's Crystal Data (PCD)
Pearson's Crystal Data (PCD), 312
phase correction algorithm, 91–93
phase space exploration, 293–294
    of porous organic materials, 299–301
physically inspired architectures
    molecular representations, 388–391
plasma-based processing, 514
PoLyInf, 287
polymers of intrinsic microporosity (PIMs), 282
polymorphous SiO_2, O-K edges of, 434–438
porous molecular materials (PMMs), 299, 300
porous organic cages (POCs), 300, 301
porous organic materials
    phase space exploration of, 299–301
    property prediction of, 299–301
porous polymer networks (PPNs), 299, 300
positive predictive value (PPV), 34
posterior probability distributions (PPDs), 199
potential energy surfaces (PESs), 76–78, 87–89, 91, 93, 94, 97, 98, 103
    exploration, 499–501
principal component analysis (PCA), 41–42, 116–118, 177, 218, 219, 239, 320, 321
    druggability prediction, 255, 257, 259
    molecular and target data analysis, 259–269, 273
process chemistry, 190
process control, baseline of, 347–348
property-labeled materials fragments (PLMF), 329
proportional, integral, and derivative (PID) controllers, 347, 348, 351, 352
PubChem, 155
Python Materials Genomics (Pymatgen), 315, 316

QML toolkit, 80, 84
quantitative materials structure–property relationship (QMSPR), 329
quantitative structure–activity relationship (QSAR), 39, 54, 147, 161, 207, 235, 288, 357, 360, 375
    activity range and distribution of activities for, 42
quantitative structure–biological effect relationships (ms-QSBER), 399, 400, 402
quantitative structure–property relationship (QSPR), 207, 210, 235, 357, 360, 375
quantum mechanics, 2–3
QuBiLS-MAS v1.0, 402, 413

radial basis function (RBF), 319, 323, 403
Raman hyperspectral imaging, SLADS for, 460–463
Random Forests (RF), 6, 58–65
    algorithms, 59–64
    druggability prediction, 255, 259
    information, 58–59
    molecular and target data analysis, 263–264
Reaction Decoder Tool (RDT), 181
reaction prediction, optimizing, 187–188
reactors, AI for modeling and separation of, 345
real-time strategy (RTS) game, 12
Reaxys, 155, 169, 171
receiver operating characteristic (ROC) curve, 34, 35
recurrent neural network (RNN), 238, 293, 294, 299, 386–388
recursive feature elimination (RFE), 321
recursive partitioning (RP)
    druggability prediction, 253, 257
    molecular and target data analysis, 273
regenerative medicine, nanomaterials–biology interface in, 215–220
ReLeaSE framework, 295
representation learning, 372–394
    challenges to, 392–393
    data considerations, 391–392

future research, directions for, 392–393
molecular representations, 380–391
graph models, 381–384
grid models, 384–386
physically inspired architectures, 388–391
sequence models, 386–387
need for, 378–380
non-learned molecular representations, 374–378
computable features, 377–378
molecular footprints, 375–377
retrosynthesis, 143–145, 180
multi-step, 185–186
single-step, 183–184
RStudio, 263–264
Rule of Five (Ro5), 251–252, 253, 255, 283

‘safe-by-design’ paradigm, 212
safety, environmental, legal, economics, control, and throughput (SELECT) criteria, 190
SciFinder, 169, 171, 185
scoring functions (SFs), for molecular drug delivery, 251–275
SCScore, 143–144
SELECT (safety, environmental, legal, economics, control, and throughput) criteria, 190
self-corrected retrosynthesis predictor (SCROP), 183, 184
self-driving laboratory (SDL), 159–162
autonomous research, implications of, 160–162
self-learning procedure, 497–499
Self Organising Map ANN approach, 254
self-organizing maps (SOMs), 218–219
sensitivity, 33
separation units, AI for modelling and simulation of
absorber and desorber, 343–344
adsorption, 344–345
distillation columns, 342–343
sequence-based models, 182

sequence models
molecular representations, 386–387
sequence-to-sequence (seq2seq) model, 142, 143, 145
sequential quadratic programming (SQP), 244, 247
sharing data, across laboratories, 155–156
similarity-based methods, 181, 188
simplex approaches, for autonomous DoE, 456–458
Simplified Molecular-Input Line-Entry System (SMILES) strings, 142–144, 153, 171, 173–174, 182–184, 207, 288, 293–295, 373, 374, 381, 384, 386–388, 402
single-step retrosynthesis, 183–184
siRNA, 214–215
small data challenges, sparse sampling strategies for, 454–456
SMILES (Simplified Molecular-Input Line-Entry System) strings, 142–144, 153, 171, 173–174, 182–184, 207, 288, 293–295, 373, 374, 381, 384, 386–388, 402
smooth overlap of atomic positions (SOAP), 286, 289, 298
SOAP-REMatch kernel, 292
soft-computing hybrid configurations, 240–243
sources of data, consistency between, 520–522
sparse sampling strategies, for small data challenges, 454–456
specificity, 33
SSW–NN method, 499, 500
stable noisy optimization by branch and fit (SNOBFIT), 138
StarCraft II, 12–14
statistical learning in chemistry, data challenges for, 147–153
bias, 148–149
narrow collection of tasks, 150–151
standard data representation, 151–153
unreliable and non-standardized data, 149–150
STATISTICA v13.5.0.17, 403, 405
Stochastic Neighbour Embedding t-SNE approach, 273, 275
stochastic surface walking (SSW)
global optimization, 494–497
structure prediction, 290–293, 326–329
supervised learning, 39

inorganic materials discovery, 322–323
supervised learning approach for dynamic sampling (SLADS), 458–460
for Raman hyperspectral imaging, 460–463
for X-ray diffraction imaging, 460–463
support vector classification (SVC), 327
support vector machine (SVM), 175–176, 319, 323, 460
druggability prediction, 254–257
hyperplanes, 66
kernels, 66–68
least square, 325
material property prediction, 331
support vectors, 68–71, 175
support vector regression (SVR), 80–81, 347, 355–356
surface hopping dynamics, 93–94
methylenimmonium cation, 94–96
Surface Hopping including ARbitrary Couplings (SHARC) method, 93, 94
synthesis optimization, 325–326
synthetic accessibility (SA), 296–297
synthetic route design, 179–188
reaction prediction, 186–187
reaction prediction, optimizing, 187–188
retrosynthesis, 183–186
Synthia™ (formally Chematica), 146, 181, 185, 189

Tanimoto Similarity, 47–48
template-based approaches to synthetic planning, 140–142, 180–181
template-free methods, 142, 181–182, 184
Tensor Field Networks, 386
TensorFlow algorithm, 11
time of flight secondary ion mass spectrometry (ToF-SIMS), 217–219
ToxCast, 9
Tox21, 147
traditional controllers with integrated machine learning, 351–352
training procedure, 23–27
Transcriptic, 155
Transfer Learning, 113

transfer learning, 332
Turing test, 3, 5, 7, 17, 145
Tversky Similarity, 48–49

ultrasmall Au/CeO_2, 501–502
United States Patent and Trademark Office (USPTO), 171, 172, 183
Universal Structure Predictor: Evolutionary Xtallography (USPEX), 327
University of Washington (UW)
    Biofabrication Center, 155
unsupervised learning, 39, 176–177
    inorganic materials discovery, 323–324

validation, 118
variational autoencoders (VAE), 293, 294, 387

Walter Reed Army Institute of Research, 161
wave function phases, NAMD simulation, 89–91
Weisfeiler–Lehman Difference Network (WLDN), 186–187
whole plants, AI for modeling and separation of, 345–346

X-ray absorption fine structure (XAFS), 426
X-ray absorption near-edge structure (XANES), 424–426, 429, 430, 444
    quantification of structures and properties from, 438–441
X-ray diffraction imaging, SLADS for, 460–463

Zhu–Nakamura approximation, 78
ZnCrO catalysts, syngas conversion on, 505–508