Generative chemistry: drug discovery with deep learning generative models

Yuemin Bian[1,2] and Xiang-Qun Xie[1,2,3,4]*

[1]Department of Pharmaceutical Sciences and Computational Chemical Genomics Screening Center, School of Pharmacy; [2]NIH National Center of Excellence for Computational Drug Abuse Research; [3]Drug Discovery Institute; [4]Departments of Computational Biology and Structural Biology, School of Medicine, University of Pittsburgh, Pittsburgh, Pennsylvania 15261, United States.

*Corresponding Author: Xiang-Qun Xie, MBA, Ph.D.

Professor of Pharmaceutical Sciences/Drug Discovery Institute

Director of CCGS and NIDA CDAR Centers

335 Sutherland Drive, 206 Salk Pavilion

University of Pittsburgh

Pittsburgh, PA15261, USA

412-383-5276 (Phone)

412-383-7436 (Fax)

Email: xix15@pitt.edu

Abstract:

The *de novo* design of molecular structures using deep learning generative models introduces an encouraging solution to drug discovery in the face of the continuously increased cost of new drug development. From the generation of original texts, images, and videos, to the scratching of novel molecular structures, the incredible creativity of deep learning generative models surprised us about the height machine intelligence can achieve. The purpose of this paper is to review the latest advances in generative chemistry which relies on generative modeling to expedite the drug discovery process. This review starts with a brief history of artificial intelligence in drug discovery to outline this emerging paradigm. Commonly used chemical databases, molecular representations, and tools in cheminformatics and machine learning are covered as the infrastructure for the generative chemistry. The detailed discussions on utilizing cutting-edge generative architectures, including recurrent neural network, variational autoencoder, adversarial autoencoder, and generative adversarial network for compound generation are focused. Challenges and future perspectives follow.

1. INTRODUCTION

Drug discovery is expensive. The average cost for the development of a new drug now hits 2.6 billion USD and the overall discovery process takes over 12 years to finish[1, 2]. Moreover, these numbers keep increasing. It is critical to think and explore efficient and effective strategies to confront the growing cost and to accelerate the discovery process. The progression in the high-throughput screening (HTS) dramatically speeded up the task of lead identification by screening candidate compounds in large volume[3, 4]. When it comes to the lead identification, the concept can be further classified into two divisions, the structure-based approach[5, 6] and the ligand-based approach[7]. Combining with the significant progress in computation, the development of these two approaches has resulted in constructive virtual screening (VS) methodologies. Traditionally, with the structure of the target protein available, structure-based approaches including molecular docking studies[8-10], molecular dynamic simulations[11-13], fragment-based approach[10, 14], etc. can be applied to explore the potential receptor-ligand interactions and to virtually screen a large compound set for finding the plausible lead. Then with the identified active molecules for the given target, ligand-based approaches such as pharmacophore modeling[15, 16], scaffolding hopping[17, 18], and molecular fingerprint similarity search[19] can be conducted for modifying known leads and for finding future compounds. The rapid advancement in computational power and the blossom of machine learning (ML) algorithms brought the ML-based decision-making model[20, 21] as an alternative path to the VS campaigns in the past decades. There is increased availability of data in cheminformatics and drug discovery. The capability of dealing with large data to detect hidden patterns and to facilitate the future data prediction in a time-efficient manner favored ML in building VS pipelines.

It is encouraging to note the successful applications of the above-mentioned computational chemistry approaches and ML-based VS pipelines on drug discovery these days. The conventional methods are effective. However, the challenge remains on developing pioneering methods, techniques, and strategies in the confrontation of the costly procedure of drug discovery. The flourishing of deep learning generative models brings fresh solutions and opportunities to this field. From the generated human faces that are indistinguishable with real people[22], to the text generation tools that mimic the tone and vocabulary of certain authors[23], the astonishing creativity of deep learning generative models brings our understanding of the machine intelligence to a new level. In recent years, the expeditions toward generative chemistry mushroomed, which explored the possibility of utilizing generative models to effectively and efficiently design molecular structures with desired properties. Promising and compelling outcomes including the identification of DDR1 kinase inhibitors within 21 days using deep learning generative models[24] may indicate that we are probably at the corner of an upcoming revolution of drug discovery in the artificial intelligence (AI) era. This review article starts with a brief evolution of AI in drug discovery, and the infrastructures in both cheminformatics and machine learning. The state-of-the-art generative models including recurrent neural networks (RNNs), variational autoencoders

(VAEs), adversarial autoencoders (AAEs), and generative adversarial networks (GANs) are focused on to discuss their fundamental architectures as well as their applications in the *de novo* drug design.

2. ARTIFICIAL INTELLIGENCE IN DRUG DISCOVERY

Artificial intelligence (AI) is the study of developing and implementing techniques that enable the machine to behave with human-like intelligence[25]. The concept of AI can be traced back to the 1950s when researches questioned whether computers can be made to handle automated intelligence tasks which are commonly fulfilled by humans[26]. Thus, AI is a broad area of research that includes both (1) methodologies employing learning processes and (2) approaches that no learning process is involved in. At the early stage, researchers believed that by defining a sufficient number of explicit rules to maneuver knowledge, the human-level AI can be expected (**Fig. 1a**). In the face of a specific problem, the human studying process on existing observations can contribute to the accumulation of knowledge. Explicit rules were expected to describe knowledge. By programming and applying these rules, the answers for future observations are anticipated. This strategy is also known as symbolic AI[27]. Symbolic AI is an efficient solution to logical problems, for instance, chess playing. However, when handling problems with blurry, unclear, and distorted knowledge, such as image recognition, language translation, and to our topic, the classification of active compounds from decoys for a therapeutic target, symbolic AI turned out to show limited capability. We may define explicit rules to guide the selection of general drug-like compounds, Lipinski's rule of five[28] for example, but it is almost impossible to exhaust specified rules for guiding the selection of agonists to cannabinoid receptor 2 or other targets[29]. Machine learning (ML) took over symbolic AI's position as a novel method with the ability to learn on its own.

ML allows computers to solve specific tasks by learning on their own[30, 31]. Through directly looking at the data, computers can summarize the rules instead of waiting for programmers to craft them (**Fig. 1b**). In the paradigm of ML-based problem solving, data and the answers to the data are functioned as input with rules as the outcome. The produced rules can then be applied to predict answers for future data. Statistical analysis is always associated with ML, while they can be distinguished at several aspects[32]. The application of ML is usually towards large and complex datasets, such as a dataset with millions of small molecules that cover a huge chemical space with diversified scaffolds, which statistical analysis can be incapable to deal with. The flourish of ML starts in the 1990s[33]. The method rapidly became a dominant player in the field of AI. Commonly used ML systems in drug discovery can be categorized into supervised learning, unsupervised learning, and reinforcement learning (**Fig. 1c**). In supervised learning, the algorithms are fed with both the data and the answers to these data (label). Protein family subtypes selectivity prediction is an example for classification: the classifier is trained with numbers of sample molecules along with their labels (the specific protein family member they interact with), and the well-trained classifier should be able to classify the future molecules[20, 29, 34, 35]. Quantitative structure-activity relationship analysis is an example for regression: the regressor is trained with

molecules sharing similar scaffold along with their biological activity data (Ki, IC_{50}, and EC_{50} values for example), and the well-trained regressor should be able to predict the numeric activity values for future molecules with the similar scaffold[10, 36]. In unsupervised learning, the algorithms are trained with unlabeled data. For instance, a high-throughput screening campaign may preselect a smaller representative compound set from a large compound database using the clustering method to group molecules with similar structures into clusters[37, 38]. A subset of molecules selected from different clusters can then offer improved structural diversity to cover a bigger chemical space than random pickup. In reinforcement learning, the learning system can choose actions according to its observation of the environment, and get a penalty (or reward) in return[39]. To achieve the lowest penalty (or highest reward), the system must learn and choose the best strategy by itself.
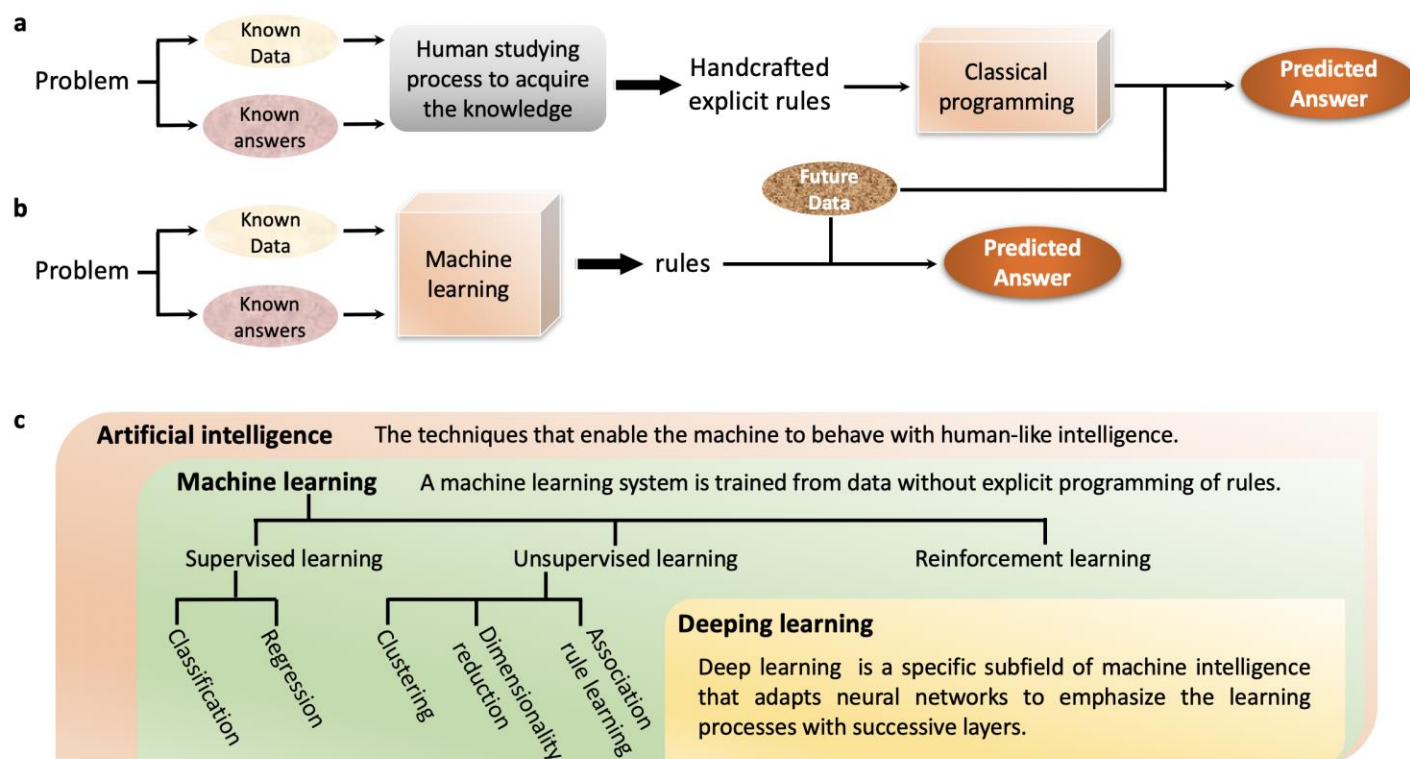

Figure 1. From artificial intelligence to deep learning. a. The programming paradigm for symbolic AI. b. The programming paradigm for ML. c. The relationship among artificial intelligence, machine learning, and deep learning.

Deep learning (DL) is a specific subfield of ML that adapts neural networks to emphasize the learning processes with successive layers (**Fig. 1c**). DL methods can transfer the representation at one level to a higher and more abstract level[40]. The feature of representation learning enables DL methods to discover representations from the raw input data for tasks such as detection and classification. The word "deep" in DL reflects this character of successive layers of representations, and the number of layers determines the depth of a

DL model[41]. In contrast, conventional ML methods that transform the input data into one or two successive representation spaces are sometimes referred to as shallow learning methods. The vast development in the past decades brought DL great flexibility on the selection of architectures, such as the fully connected artificial neural network (ANN) or multi-layer perceptron (MLP)[42], convolutional neural network (CNN)[43], and recurrent neural network (RNN)[44]. The rise of generative chemistry is largely benefited from the extensive advancement of generative modeling, which predominantly depends on the flourishing of DL architectures. The successful application of the Long Short-Term Memory (LSTM) model[45], a special type of RNN model, on text generation inspired the simplified molecular-input line-entry system (SMILES)-based compound design. And the promising exercise of using the Generative Adversarial Network (GAN) model[46] for image generation motivated the fingerprint and graph centered molecular structural scratch. The major reason for DL to bloom rapidly can be that the very method provides solutions to previously unsolvable problems and outperforms the competitors with simplified representation learning process[26, 40]. It is foreseen that the process of molecule design can evolve into a more efficient and effective manner with the proper fusion with DL.

3. DATA SOURCES AND MACHINE LEARNING INFRASTRUCTURES

Deep learning campaigns start with high-quality input data. The successful development of generative chemistry models relies on cheminformatics and bioinformatics data for the molecules and biological systems. **Table 1** exhibits some routinely used databases in drug discovery for both small and large biological molecules. In a typical case of structure-based drug discovery, a 3D model of the protein (or DNA/RNA) target is critical for the following steps on evaluating potential receptor-ligand interactions. PDB database[47] is a good source for accessing structural information for large biological systems, and the UniProt database[48] will be a convenient source for sequence data. Regarding chemicals, PubChem[49] can be a go-to place. PubChem is comprehensive. It currently contains ~103 million compounds (with unique chemical structures) and ~253 million substances (information about chemical entities). If the major focus is on bioactive molecules, ChEMBL[50] can be an efficient database to interact. ChEMBL currently documents ~2 million reported drug-like compounds with bioactivity data for 13,000 targets. Supposing that the interest is more on studying the existing drugs on the market instead of drug-like compounds, the DrugBank[51] can serve. To date, DrugBank records ~ 14,000 drugs, including approved small molecule drugs and biologics, nutraceuticals, and discovery-phase drugs. With virtual screening campaigns, adding some commercially available compounds to in-house libraries are preferred as they may further increase the structural diversity and expand the coverage of the chemical space. Once potential hits were predicted to be among these compounds, the commercial availability gives them easy access for future experimental validations. Zinc database[52] now archives ~230 million purchasable compounds in ready-to-dock format. It is worth mentioning that constructing topic-specific and target-specific databases is trending. ASD[53] is one example that files allosteric modulators and related macromolecules to facilitate the research on allosteric

modulation. The rising of Chemogenomics databases[54, 55] for certain diseases and druggable targets is another example that these libraries focus on particular areas of research.

With the input data ready, the next consideration is transforming the data into machine-readable format. **Table 2** lists commonly used molecular representations. SMILES[56] describes molecular structures in a text-based format using short ASCII strings. Multiple SMILES strings can be generated for the same molecule with different starting atoms. This ambiguity led to the effects of canonicalization that determines which of all possible SMILES will be used as the reference SMILES for a molecule. Popular cheminformatics packages such as OpenEye[57] and RDKit[58] are possible solutions for standardizing the canonical SMILES[59]. The canonical SMILES is a type of well-liked molecular representation in generative chemistry models as it packs well with language processing and sequence generation techniques like RNNs. Usually the SMILES strings are first converted with one-hot encoding. The categorical distribution for each element can then be produced by the generative models. Fingerprints are another vital group of molecular representations. Molecular Access System (MACCS) fingerprint[60] has 166 binary keys, each of which indicates the presence of one of the 166 MDL MACCS structural keys calculated from the molecular graph. Fingerprints can be calculated through different approaches. By enumerating circular fragments, linear fragments, and tree fragments from the molecular graph, Circular[61], Path, and Tree fingerprints[62] can be created. Using fingerprints as representations may suffer from inconvertibility in that the complete structure of a molecule cannot be reconstructed directly from the fingerprints[63]. To have fingerprints calculated for a large enough compounds library to function as a look-up index may be a compromised solution[64]. Despite this difficulty, fingerprints are popular among ML classification models for tasks like distinguishing active compounds from inactive ones for a given target.

Table 1. Well established cheminformatics databases available for drug discovery

Database	Description	Web linkage	Examples of usage
UniProt[48]	The Universal Protein Resource (UniProt) is a resource for protein sequence and annotation data	https://www.uniprot.org	Protein sequence homology search, alignment, and protein ID retrieving especially for structural-based drug discovery
RCSB PDB[47]	The Protein Data Bank (PDB) provides access to 3D structure data for large biological molecules, including protein, DNA, and RNA.	https://www.rcsb.org	Protein 3D structures are fundamental for hot spot identification, docking simulation, and molecular dynamics simulation in structural-based drug discovery.
PDBbind[65]	PDBbind provides a collection of the experimentally measured binding affinity data for all types of biomolecular complexes deposited in the PDB.	http://www.pdbbind.org.cn	The receptor-ligand binding data for resolved protein structures can function as the benchmark to evaluate future simulations
PubChem[49]	PubChem is the world's largest collection of chemical information.	https://pubchem.ncbi.nlm.nih.gov	To acquire comprehensive chemical information ranging from NMR spectra, physical-chemical properties, to biomolecular interactions.
ChEMBL[50]	ChEMBL is a manually curated database of bioactive molecules with drug-like properties.	https://www.ebi.ac.uk/chembl/	To collect cheminformatics data of reported molecules for a given target. A high-quality compound collection is the key to the ligand-based drug discovery
SureChEMBL[66]	SureChEMBL is a resource containing compounds extracted from patent literature.	https://www.surechembl.org/search/	Compound-patent associations
BindingDB[67]	BindingDB is a database of measured binding affinities for the interactions of protein considered to be drug-targets with small, drug-like molecules.	https://www.bindingdb.org/bind/index.jsp	To retrieve compound sets for a specific target similar to ChEMBL but with the focus on experimental binding affinities.
DrugBank[51]	The DrugBank database combines detailed drug data with comprehensive drug target information	https://www.drugbank.ca	Drug repurposing study for existing drugs. On-target and off-target analysis for a compound.
ZINC[52]	Zinc is a database of commercially-available compounds	https://zinc.docking.org	Zinc database is good for virtual screening on hit identification as the compounds are commercially available for quick biological validations afterwards.
Enamine	Enamine provides an enumerated database of synthetically feasible molecules	https://enamine.net	The establishment of a target-specific compound library. Fragment-based drug discovery.
ASD[53]	Allosteric Database (ASD) provides a resource for structure, function, disease and related annotation for allosteric macromolecules and allosteric modulators	http://mdl.shsmu.edu.cn/ASD/	To facilitate the research on allosteric modulation with enriched chemical data on allosteric modulators.
GDB[68]	GDB databases provide multiple subsets of combinatorially generated compounds following chemical stability and synthetic feasibility rules	http://gdb.unibe.ch/downloads/	Using combinatorial chemistry is a good way to largely expand the chemical space.

Table 2. Examples of commonly used molecular representations

Representation		Description
SMILES[56]		The simplified molecular-input line-entry system (SMILES) is a specification in the form of a line notation for describing the structure of chemical species using short ASCII strings.
Canonical SMILES		Canonicalization is a way to determine which of all possible SMILES will be used as the reference SMILES for a molecular graph.
InChI[69]		The International Chemical Identifier (InChI) is a textual identifier for chemical substances, designed to provide a standard way to encode molecular information.
InChI Key		The condensed, 27 character InChI Key is a hashed version of the full InChI.
Fingerprints	MACCS Keys[60]	MACCS keys are 166 bit structural key descriptors in which each bit is associated with a SMARTS pattern.
	Circular[61, 70]	Circular fingerprints are created by exhaustively enumerating all circular fragments grown radially from each heavy atom of the molecule up to the given radius.
	Path[62]	Path fingerprints are created by exhaustively enumerating all linear fragments of a molecular graph up to a given size.
	Tree[62]	Tree fingerprints are generated by exhaustively enumerating all tree fragments of a molecular graph up to a given size.
	Atom Pair[71]	Atom Pair fingerprints encode each atom as a type, enumerates all distances between pairs, and then hashes the results.

After collecting the high-quality data and transforming the data into the appropriate format, it is time to apply data science to the development of the predictive models. **Table 3** illustrates examples of frequently considered cheminformatics toolkits and machine learning packages. RDKit, Open Babel[72], and CDK[73] are cheminformatics toolkits that are comprised of a set of libraries with source codes for various functions, such as chemical files I/O formatting, substructure and pattern search, and molecular representations generation. The typical applications of deploying these toolkits can contribute to virtual screening, structural similarity search, structure-activity relationship analysis, etc[74]. The workflow environment is not unique to the cheminformatics research, but can facilitate the automation of data processing with a user-friendly interface. The workflow systems like KNIME[75, 76] can execute tasks in succession and perform recurring tasks efficiently, such as iterative fingerprints calculation for a compound library. The strategy of integrating cheminformatics toolkits as nodes into a workflow and connecting them with edges is gaining popularity and is increasingly employed[77-79]. When it comes to ML and DL modeling, TensorFlow[80], CNTK[81], Theano[82], and PyTorch[83] are well-recognized packages for employment. These packages handle low-level operations including tensor manipulation and differentiation. In contrast, Keras[84] is a model-level library that deals with tasks in a modular way. As a high-level API, Keras is running on top of TensorFlow, CNTK, and Theano. Scikit-Learn[85] is an efficient and straightforward tool for predictive data analysis. It is known more for its role in conventional ML modeling as

the library comprehensively integrates algorithms like Support Vector Machine (SVM), Random Forest (RF), Logistic regression, Naïve Bayes (NB), etc.

Table 3. Commonly used cheminformatics and machine learning packages

Package	Description	Web linkage
RDKit[58]	RDKit is an open-source toolkit for cheminformatics. Features include 2D and 3D molecular operations, descriptor generation, molecular database cartridge, etc.	https://www.rdkit.org
Open Babel[72]	Open Babel is an open chemical toolbox to search, convert, analyze, or store data from molecular modeling, chemistry, solid-state materials, biochemistry, or related areas.	http://openbabel.org/wiki/Main_Page
CDK[73]	The Chemistry Development Kit (CDK) is a collection of modular Java libraries for processing cheminformatics.	https://cdk.github.io
KNIME[75]	KNIME is a workflow environment in data science that can be integrated to automate certain cheminformatics operations.	https://www.knime.com
TensorFlow[80]	TensorFlow is an open-source platform for machine learning. It has a set of tools, libraries, and community resources that enable researchers to build and deploy ML applications.	https://www.tensorflow.org
CNTK[81]	The Cognitive Toolkit (CNTK) is an open-source toolkit for commercial-grade distributed deep learning. It describes neural networks as a series of computational steps via a directed graph.	https://github.com/microsoft/CNTK
Theano[82]	Theano is a Python library for defining, optimizing, and evaluating mathematical expressions.	http://deeplearning.net/software/theano/
PyTorch[83]	PyTorch is an open-source machine learning library based on the Torch library.	https://pytorch.org
Keras[84]	Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation.	https://keras.io
Scikit-Learn[85]	Scikit-learn is a free software machine learning library for the Python programming language.	https://scikit-learn.org/stable/

4. GENERATIVE CHEMISTRY WITH THE RECURRENT NEURAL NETWORK (RNN)

RNN is a widely used neural network architecture in generative chemistry for proposing novel structures. As a type of powerful generative model especially in natural language processing, RNNs usually use sequences of words, strings, or letters as the input and output[44, 86-88]. In this case, the SMILES strings are usually employed as a molecular representation. Different from ANNs and CNNs which do not have memories, RNNs iteratively process sequences and store a state holding current information. On the contrary, ANNs and CNNs process each input independently without stored information between them. When describing an RNN, it can be considered as a network with an internal loop that loops over the sequence elements instead of processing in a single step (**Fig. 2a**). The state that stored information will be updated during each loop. For simplicity, the process of

computing the output y can follow the equation: $y =$ activation $(W_o x + U_o h + b_o)$, where W_o and U_o are weight matrices for the input x and state h, and b_o as a bias vector. **Figure 2a** can represent the structure of a simple RNN model. However, this structure can suffer severely from the vanishing gradient problem which makes neural networks untrainable after adding more layers. Even though the state h is supposed to hold the information from the sequence elements previously seen, the long-term dependencies make the learning process impossible[89, 90]. The Long Short-Term Memory (LSTM) algorithm[45] was developed to overcome this shortcoming. The LSTM layer attaches a carry track to carry information across the learning process to counter the loss of signals from gradual vanishing (**Fig. 2b**). With this carry track, the information learned from each sequence element can be loaded, and the loaded information can be transported and accessed at a later stage. The process of computing the output y for LSTM is similar with the previous equation but adding the contribution of the carry track: $y =$ activation $(W_o x + U_o h + V_o c + b_o)$, where W_o, U_o, and V_o are weight matrices for the input x, state h, and carry c, and b_o as a bias vector. In certain cases, multiple recurrent layers in a model can be stacked to enhance representational power.

A typical framework on generative modeling for molecule generation applying LSTM algorithm (**Fig. 2c**) starts with the collection of training molecules. The RNN model can be fine-tuned through the transfer learning that first accumulates knowledge from the large compound datasets and then produces the novel structures by learning smaller focused datasets. When the collections of training molecules (for large sets or small focused sets) are ready, SMILES strings can be calculated for each molecule. One-hot encoding is a regular operation for processing the molecular representations. In one-hot encoding, a unique integer index i is assigned to every character in the SMILES string. Then a binary vector can be constructed of size C (the number of unique characters in the string) with all zeros but for the ith entry which is one. For instance, there are four ($C = 4$) unique characters, "C", "N", "c", and "1" in SMILES strings, input "C" is transferred to (1, 0, 0, 0), "N" to (0, 1, 0, 0), "c" to (0, 0, 1, 0), and "1" to (0, 0, 0, 1) after one-hot encoding. In practice, usually an additional starting character like "G" and an ending character like "E" will be added to the SMILES to denote a complete string. The neural network with LSTM layer(s) can be trained to predict the $n+1$th character given the input of string with n characters. The probability of distribution for the $n+1$th character is calculated as the loss to evaluate the model performance. With the trained model, the sampling process can start with the starting character or certain SMILES strings of molecular fragments to sample the next character until the ending character is hit. The SMILES strings are reversed from the generated binary matrices according to the previous one-hot encoding to construct the molecular graphs as the output for this generative model.
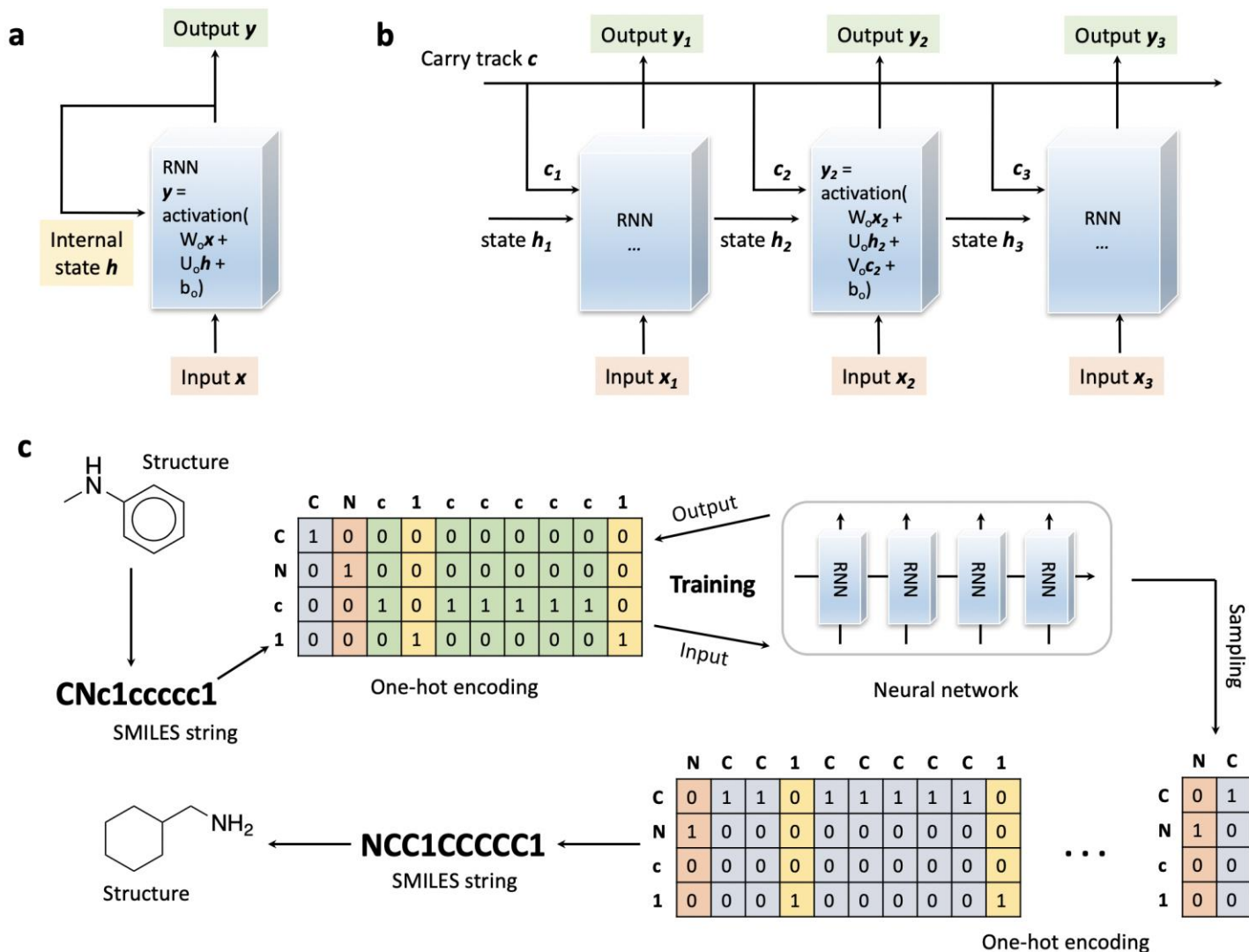
Figure 2. The RNN, the LSTM, and their application in generative chemistry. a. The schematic illustration of the RNN, the neural network with an internal loop. b. The schematic illustration of data processing with the LSTM. c. The typical framework on building generative models applying RNN for molecules generation.

Representative case studies are discussed in this paragraph. All the case applications covered in this review are summarized in **Table 4**. Anvita Gupta et al. trained an LSTM-based generative model with transfer learning to generate libraries of molecules with structural similarity to known actives for PPARγ and trypsin[91]. The model was first trained with 550,000 SMILES strings of active compounds from ChEMBL and further fine-tuned with SMILES strings for 4,367 PPARγ ligands and 1,490 trypsin inhibitors. Among the valid generated molecules, around 90% are unique from the known ligands and are different from each other. The proposed model was assessed for fragment-based drug discovery as well. In fragment-based drug discovery, fragment growing is a strategy for novel compounds generation with the identified fragment lead. Substitutions can be added to the identified fragment with the consideration of pharmacophore features and proper physical-chemical properties to enhance the receptor-ligand interactions[92]. Instead of using the starting character to

initiate the generative process, the SMILES string of the molecular fragment can be read and extended by calculating the probability of distribution for the next character. Marwin H. S. Segler et al. also reported their application of LSTM-based generative models for structure generation with transfer learning[93]. There was a good correlation between the generated structures and the molecules used for training. Notably, the complete *de novo* drug design cycle can be achieved with target prediction models for scoring. As the target prediction model can be a molecular docking algorithm or even robot synthesis and bio-testing system, the drug design cycle does not require known active compounds to start. Chemical Language Model (CLM) proposed by Michael Moret et al. is another example of applying LSTM-based generative models to work with chemical SMILES strings with transfer learning processes[94]. This approach enables the early stage molecular design in a low data regime. When it comes to real-world validation, Daniel Merk et al. published their prospective study with experimental evaluations[95]. Using the SMILES strings as the input, the LSTM-based generative model was trained and fine-tuned with the transfer learning process for the peroxisome proliferator-activated receptor. Five top-ranked compounds designed by the model were synthesized and tested. Four of them have nanomolar to low micromolar activities in cell-based assays. Besides using the LSTM algorithm, some other RNN architectures such as implementing Gated Recurrent Unit[96] (GRU) can also have promising applications. GRU layers work with the same principle as LSTM layers but may have less representational power. Shuangjia Zheng et al. developed a quasi-biogenic molecular generator with GRU layers[97]. As biogenic compounds and pharmaceutical agents are biologically relevant, over 50% of existing drugs result from drug discovery campaigns starting with biogenic molecules. Their generative model is an effort to explore greater biogenic diversity space. Similarly, focused compound libraries can be constructed with transfer learning processes.

5. GENERATIVE CHEMISTRY WITH THE VARIATIONAL AUTOENCODER (VAE)

The principle aim of an autoencoder (AE) is to construct a low-dimensional latent space of compressed representations that each element can be reconstructed to the original input (**Fig. 3a**). The module that maps the original input data, which is in high-dimension, to a low-dimensional representation is called the encoder, while the module that realizes the mapping and reconstructs the original input from the low-dimensional representation is called the decoder[41, 98]. The encoder and the decoder are usually neural networks with RNN and CNN architectures as SMILES strings and molecular graphs are commonly used molecular representations. With the molecular representations calculated, a typical data processing procedure with AE on molecule generation starts with encoding the input into a low-dimensional latent space. Within the latent space, the axis of variations from the input can be encoded. Using the variation of molecular weight (M.W.) as an example, while in practice the features learned can be highly abstractive as the M.W. is used here for simplified illustration, the points along this axis are embedded representations of compounds with different M.W. These variations are termed concept vectors. With an identified vector, it makes the molecular editing possible by

exploring the representations in a relevant direction. The encoded latent space with compressed representations can then be sampled with the decoder to map them back to molecular representations. Novel structures alongside the original input can be expected.
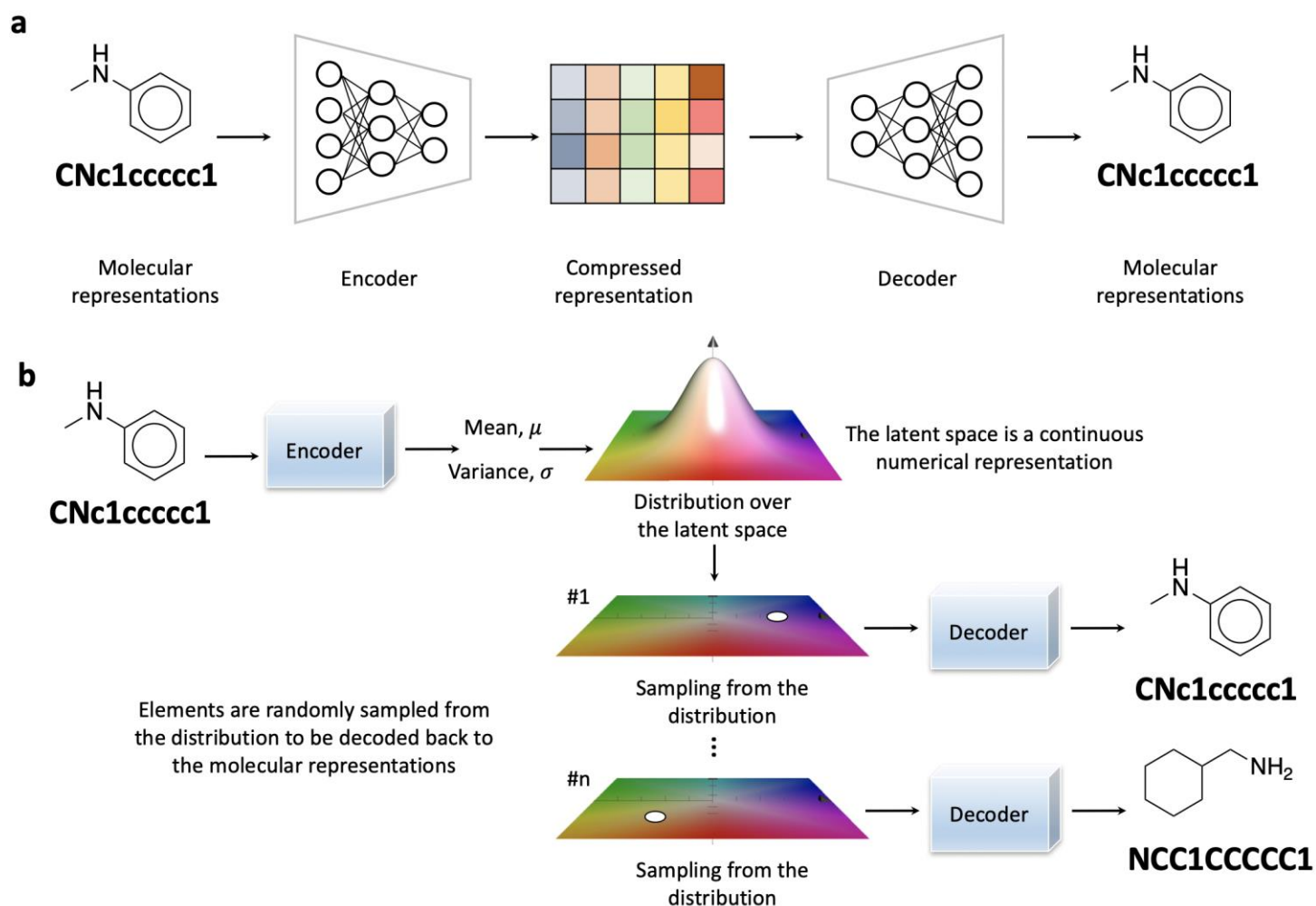

Figure 3. The autoencoder and the variational autoencoder. a. An autoencoder encodes input molecules into compressed representations and decodes them back. b. A variational autoencoder maps the molecules into the parameters of a statistical distribution as the latent space is a continuous numerical representation.

The concept of VAE was first proposed by Kingma and Welling at the end of 2013[99, 100]. This technique quickly gained popularity in building robust generative models for images, sounds, and texts[101-103]. The AE compresses a molecule x into a fixed code in the continuous latent space z, and trends to summarize the explicit mapping rules as the number of adjustable parameters is often much more than the number of training molecules. These explicit rules make the decoding of random points in the continuous latent space challenging and sometimes impossible[26]. Instead, VAE maps the molecules into the parameters of a statistical distribution (**Fig. 3b**). With $p(z)$ describing the distribution of prior continuous latent space, the probabilistic encoding

distribution is $q_\varphi(z|x)$ and the probabilistic decoding distribution is $p_\theta(x|z)$. The training iterations with back propagation will gradually optimize the parameters of both $q_\varphi(z|x)$ and $p_\theta(x|z)$. VAE is fundamentally a latent variable model $p(x,z) = p_\theta(x|z)p(z)$. The stochasticity of the training process enables the latent space to encode valid representations, which further results in a structured latent space[100]. Both the reconstruction loss and the regularization loss are often used for parameter optimization during the training process. The reconstruction loss evaluates whether the decoded samples match the input while the regularization loss investigates whether the latent space is overfitting to the training data.

Applications of VAE for generating chemical structures started in 2016 as Rafael Gómez-Bombarelli et al. developed a VAE-based automatic chemical design system[104]. In their practice, the ZINC database and QM9 dataset were referred to as the sources for collecting molecules. The QM9 dataset archives small molecules following three rules: (1) no more than 9 heavy atoms, (2) with 4 distinct atomic numbers, and (3) with 4 bond types. Canonical SMILES strings were calculated as the molecular representation. The encoder maps input SMILES strings into the continuous real-valued vectors, and the decoder reconstructs molecular representations from these vectors. The encoder was formed with three convolutional layers and one fully connected dense layer while the decoder contained three GRU layers. The architectures of CNNs and RNNs were compared for string encoding and convolutional layers achieved superior performance. The last layer of the decoder would report a probability distribution for characters of the SMILES string at each position. This stochastic operation allowed the same point in the latent space to have different decoded outcomes. Besides, they added one additional module for property prediction. An MLP was jointed to predict the property values from the continuous representation created by the encoder in order to optimize the desired properties for the new molecules. Thomas Blaschke et al. tested various generative AE models including VAE for compound design targeting dopamine receptor 2 (DRD2)[105]. Their study showed that the generated latent space preserved the chemical similarity principles. The generated molecules similar to known active compounds can be observed. In their VAE model, CNN layers were used for the encoder for pattern recognition and the RNN layers of GRU cells were adapted for the decoder. The ChEMBL database functioned as the data source for molecular structures. Canonical SMILES were prepared as the molecular representation. Similarly, an SVM classification model trained with extensive circular fingerprint (ECFP) of active and inactive DRD2 ligands was integrated to investigate the newly generated molecules. Boris Sattarov et al. combined a sequence-to-sequence VAE model with generative topographic mapping (GTM) for molecular design[106]. Both the encoder and the decoder were RNN models containing two LSTM layers in their practice. SMILES strings with one-hot encoding for molecules from the ChEMBL database were prepared prior to the training. Their GTM module contributed to the selection of sampling points in the VAE latent space, which facilitated the generation of a focused library of compounds with desired properties.

Besides the use of SMILES strings, molecular graphs have also been applied as a type of molecular representation to feed the VAE models. Bidisha Samanta et al. proposed NeVAE, a VAE-based compound generative model employing molecular graphs[107]. The molecular structures are usually not grid-like and come with an inconsistent number of nodes and edges, which impedes the use of molecular graphs as representations. In their work, the molecular graphs were prepared for drug-like compounds collected from the ZINC database and QM9 dataset. The nodes and edges in the graph represent atoms and bonds respectively. The node features are types of atoms with one-hot encoding and the edge weights are bond types (saturated bonds, unsaturated double/triple bonds, etc.). The purpose of training is to enable the VAE to create credible molecular graphs including node features and edge weights. Another example is GraphVAE. Martin Simonovsky et al. proposed GraphVAE to facilitate the compound design using molecular graphs[108]. Their central hypothesis was to decode a probabilistic fully-connected graph in which the existence of nodes, edges, and their attributes are independent random variables. The encoder was a feed-forward network with convolutional layers and the architecture for the decoder was an MLP. The model training and evaluation involved the molecules from the ZINC database and QM9 dataset. Some other generative applications can switch the topic to lead optimizations with methods such as scaffold hopping, substitutions design, and fragment-based approaches. One example is the DeLinker which was proposed by Fergus Imrie et al. to incorporate two fragments into a new molecule[109]. This method is VAE-based, using molecular graphs as the input. The design process heavily relied on 3D structural information that considers relative distance and orientation between the starting fragments.

6. GENERATIVE CHEMISTRY WITH THE ADVERSARIAL AUTOENCODER (AAE)

The architecture of the AAE is comparatively similar to the VAE except the appending of the additional discriminator network[110]. An AAE trains three modules, an encoder, a decoder, and a discriminator (**Fig. 4**). The encoder learns the input data and maps the molecule into the latent space following the distribution of $q_\phi(z|x)$. The decoder reconstructs molecules through sampling from the latent space following the probabilistic decoding distribution of $p_\theta(x|z)$. And the discriminator distinguishes the distribution of the latent space $z \sim q_\phi(z)$ from the prior distribution $z' \sim p(z)$. During the training iterations, the encoder is modified consistently to have the output, $q_\phi(z|x)$, follow a specific distribution, $p(z)$, in an effort to minimize the adversarial cost of the discriminator. A simplistic prior, like Gaussian distribution, is assumed in VAE, while alternative priors can exist in real-world practices[111]. The AAE architecture with the additional discriminator module demonstrates improved adaptability.
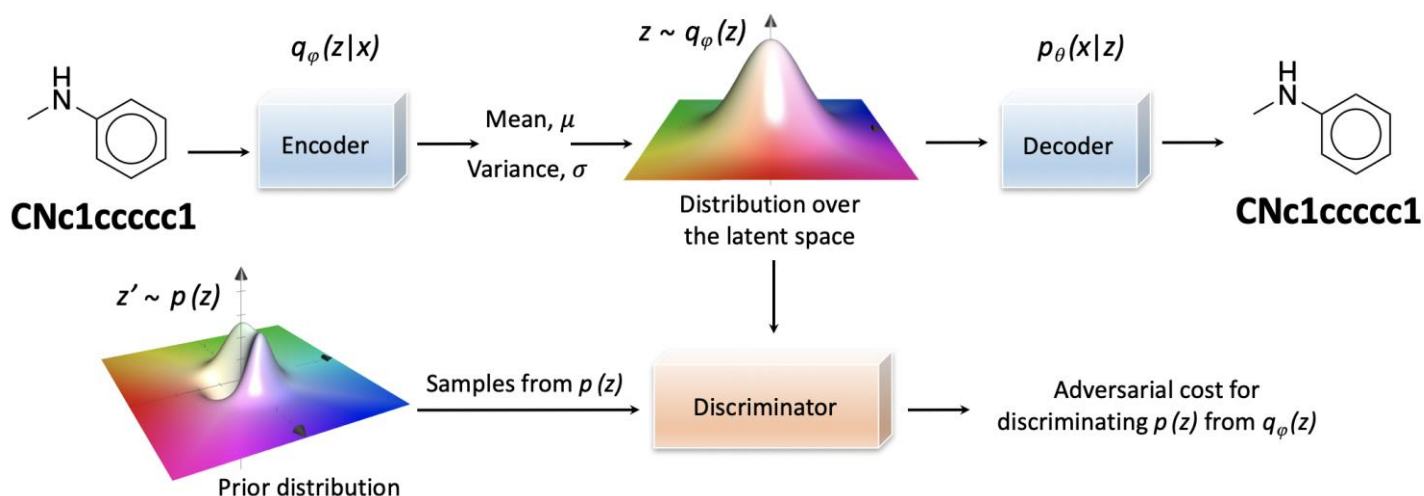
Figure 4. The illustrated architecture of an adversarial autoencoder. A discriminator network is appended to calculate the adversarial cost for discriminating *p(z)* from $q_\varphi(z)$. As a result, the outcome latent space from the encoder is driven to follow the prior distribution.

Thomas Blaschke et al. summarized a three-step training process in their compound design practice with AAE: (1) The simultaneous training of both the encoder and the decoder to curtail the reconstruction loss of the decoder; (2) The training of the discriminator to distinguish the distribution of the latent space, $q_\varphi(z)$, from the prior distribution *p(z)* effectively; (3) The training of the encoder to minimize the adversarial cost for discriminating *p(z)* from $q_\varphi(z)$[105]. The training iterations continue until the reconstruction loss converges. Artur Kadurin et al. proposed the method of using a generative adversarial autoencoder model to identify fingerprints of new molecules with potential anticancer properties[111]. The input molecules come from a small data set of compounds profiled on the MCF-7 cell line. The MACCS fingerprints were used as the molecular representation and two fully connected dense layers with different dimensions were used as the network architecture for the encoder, decoder, and the discriminator. One notable modification in this study was the removal of the batch normalization layers for the discriminator. Batch normalization is an optimization method that reduces the covariance shift among the hidden units and allows each layer to learn more independently. In the authors' opinion, the noise from the generator can be masked into target random noise with the batch normalization layers, which prohibits the training of the discriminator. As each bit of the MACCS fingerprints represents certain substructure features, the learned structural information by machine can be beneficial to the design of chemical derivatives for identified leads. Daniil Polykovskiy et al. reported their work on building a conditional AAE for molecule design targeting Janus kinase 3 (JAK3)[112]. The contributions from a set of physical-chemical properties including bioactivity, solubility, and synthesizability were considered and the model was conditioned to produce molecules with specified properties. Clean lead molecules were collected from the ZINC database and encoded as SMILES strings. The LSTM layers are adapted for building the

encoder and the decoder networks. Both *in silico* method (molecular docking) and *in vitro* assay (inhibition of JAK2 and JAK3) were conducted as the evaluation for the newly generated molecules. Rim Shayakhmetov et al. reported a bidirectional AAE model that generates molecules with the capacity of inducing a desired change in gene expression[113]. The model was validated using LINCS L1000, a database that collects gene expression profiles. The molecular structures *x* and induced gene expression changes *y* contributed to a joint model *p(x,y)*. In this specific conditional task, there is no direct association between *x* and *y* as certain changes at the gene expression are irrelevant to the drug-target interactions. The proposed bidirectional AAE model then learned the joint distribution and decomposed objects into shared features, exclusive features to *x*, and exclusive features to *y*. Therefore, the discriminator that divides the latent representations into shared and exclusive sections was constructed to secure the conditional generation to be consequential.

Table 4. Representative applications of generative chemistry covered in this review

#	Generative architecture	Neural networks involved	Data source	Molecular representation	Note	Ref.
1	RNN	LSTM	ChEMBL	SMILES	The application was extended to fragment-based drug design.	91
2	RNN	LSTM	ChEMBL	SMILES	The design-synthesis-test cycle was simulated with target prediction models for scoring.	93
3	RNN	LSTM	ChEMBL	SMILES	A chemical language model (CLM) in low data regimes.	94
4	RNN	LSTM	ChEMBL	SMILES	A prospective application with experimental validations of top-ranking compounds.	95
5	RNN	GRU	ZINC ChEMBL	SMILES	The generative model explored greater biogenic diversity space.	97
6	VAE	Encoder: CNN Decoder: GRU	ZINC QM9	SMILES	An MLP model was jointed to predict property values.	104
7	VAE	Encoder: CNN Decoder: GRU	ChEMBL	SMILES	An SVM classification model was added to evaluate the outcome.	105
8	VAE	Encoder: LSTM Decoder: LSTM	ChEMBL	SMILES	A sequence-to-sequence VAE model was combined with generative topographic mapping (GTM) for molecular design.	106
9	VAE	Encoder: CNN Decoder: CNN	ZINC QM9	Molecular graph	The nodes and edges in the graph of NeVAE represent atoms and bonds respectively.	107
10	VAE	Encoder: CNN Decoder: MLP	ZINC QM9	Molecular graph	The central hypothesis of GraphVAE was to decode a probabilistic fully-connected graph.	108

11	VAE	Encoder: GGNN[#] Decoder: GGNN	ZINC CASF[*]	Molecular graph	DeLinker was designed to incorporate two fragments into a new molecule.	109
12	AAE	Encoder: MLP Decoder: MLP Discriminator: MLP	MCF-7[^]	MACCS fingerprints	Fingerprints cannot be directly converted to structures but can provide certain substructure information.	111
13	AAE	Encoder: LSTM Decoder: LSTM Discriminator: MLP	ZINC	SMILES	The generated molecules targeting JAK3 were evaluated with *in silico* and *in vitro* methods.	112
14	AAE	Encoder: GRU Decoder: GRU Discriminator: MLP	LINCS[&] ChEMBL	SMILES	The combination of molecules and gene expression data were analyzed.	113
15	GAN	Discriminator: CNN Generator: LSTM	ZINC	SMILES	Sequence generation with objective-reinforced generative adversarial networks (ORGAN).	114
16	GAN	Discriminator: MLP Generator: MLP	ZINC	Molecular graph	The model operated in the latent space trained by the Junction Tree VAE.	115
17	GAN	Discriminator: MLP Generator: MLP	LINCS[&]	SMILES	The compound design was connected to the systems biology.	116
18	GAN	Encoder: LSTM Decoder: LSTM Discriminator: MLP Generator: MLP	ChEMBL	SMILES	The concept of the autoencoder and the generative adversarial network was combined to propose a latentGAN.	117

[#]GGNN represents the gated graph neural network. [*]CASF is also known as PDBbind core set. [^]MCF-7 represents a small data set of compounds profiled on the MCF-7 cell line. [&]LINCS represents the LINCS L1000 dataset that collects gene expression profiles.

7. GENERATIVE CHEMISTRY WITH THE GENERATIVE ADVERSARIAL NETWORK (GAN)

The architecture of the convolutional neural network[43] (CNN) is briefly covered in this section as the convolutional layers are widely used in GAN modeling. The implementation of convolutional layers can also be found in case studies discussed above among autoencoder models. A convolutional layer does not learn an input globally but focuses on the local pattern within a receptive field, the kernel (**Fig. 5a**). The low-level patterns learned in a prior layer can then be concentrated on the high-level features at the subsequent layers[118, 119]. This characteristic allows the CNN to learn and summarize abstract patterns with complexity. Another characteristic that comes out from the local pattern learning is that the learned features can be recognized anywhere[118]. It enables the CNN to process input data with efficiency and powerfulness even with a smaller number of input sample representations. Meanwhile, multiple feature maps (filters) can be stacked to encode different aspects of the input data. Applying several filters capacitates a CNN model to detect distinct features anywhere among the input data. The pooling operation on the other hand subsamples the feature map to reduce the number of parameters and eventually, the computational load[120]. Using a max-pooling layer as one example, only the max input value in that pooling kernel will be kept. Alongside with dropout layers and regularization penalties, the

pooling layers also contribute to confronting the overfitting issues. Putting together, the convolutional layers, pooling layers, and dense layers are carefully selected and arranged to construct a sophisticated CNN architecture.
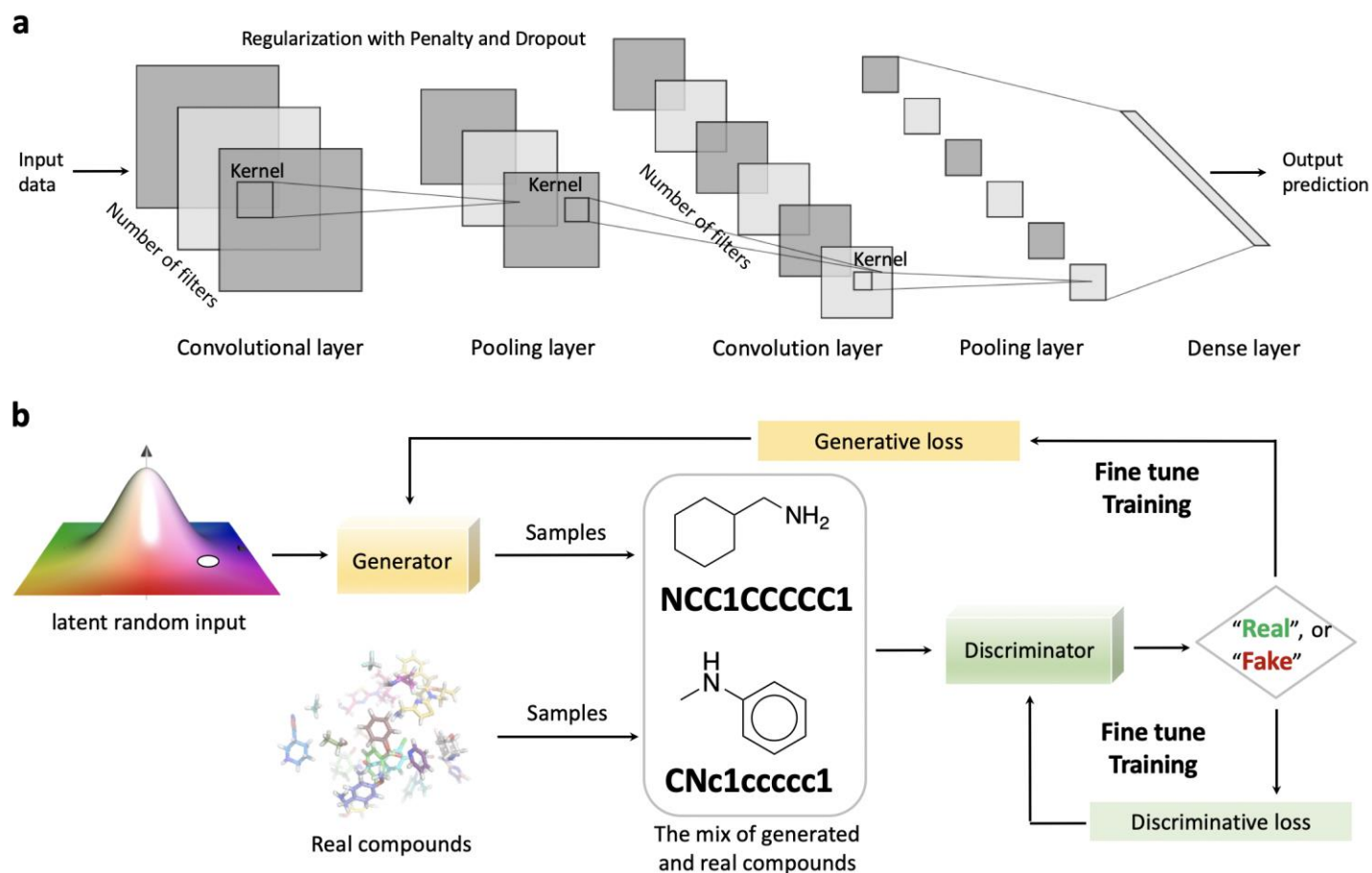

Figure 5. Sample architecture of the convolutional neural network and the framework of a generative adversarial network. a. The careful selection and arrangement of convolutional layers, pooling layers, and dense layers, etc. constitute a convolutional neural network. b. The generative adversarial network comprises two modules, the generator and the discriminator. Both the generative loss and discriminative loss are monitored during the training process.

The concept of the GAN was first raised by Ian Goodfellow in 2014[46]. The method quickly gained popularity on generative tasks regarding image, video, and audio processing and related areas[121-123]. Two models, the discriminator and the generator are trained iteratively and simultaneously during the adversarial training process[63]. The discriminator is supposed to discover the hidden patterns behind the input data and to make accurate discrimination of the authentic data from the ones generated by the generator. The generator is trained to keep proposing compelling data to fool the well-trained discriminator by consistently optimizing the data sampling process. The training process is a zero-sum noncooperative game with the purpose of achieving

the Nash equilibrium by the discriminator and the generator. In generative chemistry, the generator generates SMILES strings, molecular graphs, or fingerprints, depending on the selection of the molecular representation, using the latent random inputs (**Fig. 5b**). The generated molecules are mixed with the samples of real compounds to feed the discriminator after correct labeling. The discriminative loss is calculated to evaluate whether the discriminator can distinguish the real compounds from the generated ones, while the generative loss is computed to assess whether the generator can fool the discriminator by generating undistinguishable molecules. The constringency of both loss functions after the iterative training indicates that even a well-established discriminator can be misled to classify generated molecules as real, which further reflects that the generator has learned and accumulated authentic data patterns to create captivating compounds. However, it is worth mentioning that the simultaneous optimization of both loss functions is challenging as the instability can lead to the gradient of one part instead of both being favored (results in a stronger discriminator or generator, but not both). Another limitation may come from the restricted chemical space that is being covered by the generated molecules[64]. To confront the discriminator and minimize the generative loss, the generator can only explore a limited chemical space defined by the real compounds.

Gabriel Guimaraes et al. presented a sequence-based GAN framework termed objective-reinforced generative adversarial network (ORGAN)[114] that includes domain-specific objectives to the training process besides the discriminator reward. The discriminator drove the generated samples to follow the distribution of the real data and the domain-specific objectives secured that the traits maximizing the specific heuristic would be selected. The drug-like and nondrug-like molecules were collected from ZINC databases. SMILES strings were calculated as the molecular representations. A CNN model was designed as the discriminator to classify texts, and an RNN model with LSTM units was used as the generator. Łukasz Maziarka et al. introduced Mol-CycleGAN for derivatives design and compound optimization[115]. The model could generate structures with high similarity to the original input but improved values on considered properties. Molecular graphs of compounds extracted from the ZINC database were used as the molecular representation. The model operated in the latent space trained by the Junction Tree VAE. Dense layers and fully connected residual layers constituted the generator and the discriminator. Oscar Méndez-Lucio et al. reported a GAN model to connect the compound design with systems biology[116]. They have shown that active-like molecules can be generated given that the gene expression signature of the selected target is supplied. The architectures of both the discriminator and the generator were composed with dense layers. There were two stages of training: in stage I, the random noise was taken as the input, while in stage II, the output from stage I and the gene expression signature were taken. Oleksii Prykhodko et al. combined the concept of AE with GAN and proposed a latent vector-based GAN model[117]. A heteroencoder mapped one-hot encoded SMILES strings into the latent space and the generator and discriminator would directly use the latent vector to focus on the optimization of the sampling process. A pre-

trained heteroencoder was then used to transfer the generated vectors back to molecular structures. Both general drug-like compounds and target-biased molecules were generated as applications of the method.

8. CONCLUSION AND FUTURE PERSPECTIVES

Besides the successful generative chemistry stories described above, challenges and opportunities can be found at the following four aspects: (1) the synthetic feasibility of the generated structures, (2) the alternative molecular representations that can better portray a structure, (3) the generation of macro-molecules, and (4) the close-loop automation in combination with experimental validations. Wenhao Gao et al. pointed out that generative models can propose unrealistic molecules even with high performance scores on quantitative benchmarks[124]. Some existing methods of evaluating the synthesizability are based on synthetic routes and molecular structural data, which require heuristic definition to be complex and comprehensive[125], while the change of one single functional group to a scaffold can cause a distinctive synthetic pathway. The ignorance of synthesizability turns out to be an eminent hindrance of connecting generative models with medicinal chemistry synthesis. The molecular representations such as SMILES strings and molecular fingerprints serve well on describing small molecules at the current stage. However, it will be appealing if the novel representations can be designed to also consider three-dimensional geometry data. Chiral compounds may exhibit divergent activities to the biological system[126], and even the conformational change of the same small molecule can alter the receptor-ligand interactions. The case studies that deployed molecular graphs as the representation illustrate the benefits of working with structures directly[107-109, 115]. The extended consideration of bond type, length, and angles improves the performance of feature extraction on spatial patterns. Peptides possess superior advantage among protein subtype selectivity. The strategy of developing antibodies and peptides as therapeutic agents draws increasing attention from both the academia and industry. Deep learning is data-driven research. Current generative chemistry applications mainly focus on the design of small molecules as there is increased availability of accessing chemical data[127]. As the construction of protein-related databases is rising, the attempts of *de novo* protein generation are expected[128]. Better representations are certainly required for describing protein, as the folding and its conformation are even more critical to determine the functionality. Lastly, it is noteworthy of how to integrate the generative chemistry into the drug design framework to close the loop of this automated process. Marwin H. S. Segler et al. mentioned a design-synthesis-test cycle in their application of using the RNN model to generate molecules[93]. Ideally, the HTS will first recognize some hit compounds for a given target. The identified hits will contribute to the iterative training of a deep learning generative model for novel compounds generation, and a machine learning-based target prediction model for virtual classification. The top molecules will be synthesized and tested with biological assays. The true new actives can then be appended to the identified hits, which closes the loop.

In a nutshell, this paper reviewed the latest advances of generative chemistry that utilizes deep learning generative models to expedite the drug discovery process. The review starts with a brief history of AI in drug discovery to outline this emerging paradigm. Commonly used chemical databases, molecular representations, and operating sources of cheminformatics and machine learning are covered as the infrastructure. The detailed discussions on RNN, VAE, AAE, and GAN are centered, which is followed by future perspectives. As a fast-growing area of research, we are optimistic to expect a boosting number of studies on generative chemistry. We are probably at the corner of an upcoming revolution of drug discovery in the AI era, and the good news is that we are witnessing the change.

9. AUTHOR INFORMATION

Corresponding author

Author to whom correspondence should be addressed: Xiang-Qun Xie

Notes

The authors declare no competing financial interest.

10. ACKNOWLEDGEMENTS

11. REFERENCES

1.   Chan, H. S.; Shan, H.; Dahoun, T.; Vogel, H.; Yuan, S., Advancing drug discovery via artificial intelligence. *Trends in pharmacological sciences* **2019**.
2.   Dickson, M.; Gagnon, J. P., The cost of new drug discovery and development. *Discovery medicine* **2009**, 4, 172-179.
3.   Chen, H.; Engkvist, O.; Wang, Y.; Olivecrona, M.; Blaschke, T., The rise of deep learning in drug discovery. *Drug discovery today* **2018**, 23, 1241-1250.
4.   Broach, J. R.; Thorner, J., High-throughput screening for drug discovery. *Nature* **1996**, 384, 14-16.
5.   Kroemer, R. T., Structure-based drug design: docking and scoring. *Current protein and peptide science* **2007**, 8, 312-328.
6.   Blundell, T. L., Structure-based drug design. *Nature* **1996**, 384, 23.
7.   Bacilieri, M.; Moro, S., Ligand-based drug design methodologies in drug discovery process: an overview. *Current drug discovery technologies* **2006**, 3, 155-165.
8.   Pagadala, N. S.; Syed, K.; Tuszynski, J., Software for molecular docking: a review. *Biophysical reviews* **2017**, 9, 91-102.
9.   Bian, Y.-m.; He, X.-b.; Jing, Y.-k.; Wang, L.-r.; Wang, J.-m.; Xie, X.-Q., Computational systems pharmacology analysis of cannabidiol: a combination of chemogenomics-knowledgebase network analysis and integrated in silico modeling and simulation. *Acta Pharmacologica Sinica* **2019**, 40, 374-386.
10.   Bian, Y.; Feng, Z.; Yang, P.; Xie, X.-Q., Integrated in silico fragment-based drug design: case study with allosteric modulators on metabotropic glutamate receptor 5. *The AAPS journal* **2017**, 19, 1235-1248.

11.   Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A., Development and testing of a general amber force field. *Journal of computational chemistry* **2004**, 25, 1157-1174.

12.   Vanommeslaeghe, K.; Hatcher, E.; Acharya, C.; Kundu, S.; Zhong, S.; Shim, J.; Darian, E.; Guvench, O.; Lopes, P.; Vorobyov, I., CHARMM general force field: A force field for drug‐like molecules compatible with the CHARMM all‐atom additive biological force fields. *Journal of computational chemistry* **2010**, 31, 671-690.

13.   Ge, H.; Bian, Y.; He, X.; Xie, X.-Q.; Wang, J., Significantly different effects of tetrahydroberberrubine enantiomers on dopamine D1/D2 receptors revealed by experimental study and integrated in silico simulation. *Journal of computer-aided molecular design* **2019**, 33, 447-459.

14.   Hajduk, P. J.; Greer, J., A decade of fragment-based drug design: strategic advances and lessons learned. *Nature reviews Drug discovery* **2007**, 6, 211-219.

15.   Yang, S.-Y., Pharmacophore modeling and applications in drug discovery: challenges and recent advances. *Drug discovery today* **2010**, 15, 444-450.

16.   Wieder, M.; Garon, A.; Perricone, U.; Boresch, S.; Seidel, T.; Almerico, A. M.; Langer, T., Common hits approach: combining pharmacophore modeling and molecular dynamics simulations. *Journal of chemical information and modeling* **2017**, 57, 365-385.

17.   Liu, Z.; Chen, H.; Wang, P.; Li, Y.; Wold, E. A.; Leonard, P. G.; Joseph, S.; Brasier, A. R.; Tian, B.; Zhou, J., Discovery of Orally Bioavailable Chromone Derivatives as Potent and Selective BRD4 Inhibitors: Scaffolding Hopping, Optimization and Pharmacological Evaluation. *Journal of Medicinal Chemistry* **2020**.

18.   Hu, Y.; Stumpfe, D.; Bajorath, J. r., Recent advances in scaffold hopping: miniperspective. *Journal of medicinal chemistry* **2017**, 60, 1238-1246.

19.   Muegge, I.; Mukherjee, P., An overview of molecular fingerprint similarity search in virtual screening. *Expert opinion on drug discovery* **2016**, 11, 137-148.

20.   Fan, Y.; Zhang, Y.; Hua, Y.; Wang, Y.; Zhu, L.; Zhao, J.; Yang, Y.; Chen, X.; Lu, S.; Lu, T., Investigation of Machine Intelligence in Compound Cell Activity Classification. *Molecular Pharmaceutics* **2019**, 16, 4472-4484.

21.   Minerali, E.; Foil, D. H.; Zorn, K. M.; Lane, T. R.; Ekins, S., Comparing Machine Learning Algorithms for Predicting Drug-Induced Liver Injury (DILI). *Molecular Pharmaceutics* **2020**.

22.   Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; Aila, T., Analyzing and improving the image quality of stylegan. *arXiv preprint arXiv:1912.04958* **2019**.

23.   Wen, T.-H.; Gasic, M.; Mrksic, N.; Su, P.-H.; Vandyke, D.; Young, S., Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745* **2015**.

24.   Zhavoronkov, A.; Ivanenkov, Y. A.; Aliper, A.; Veselov, M. S.; Aladinskiy, V. A.; Aladinskaya, A. V.; Terentiev, V. A.; Polykovskiy, D. A.; Kuznetsov, M. D.; Asadulaev, A., Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nature biotechnology* **2019**, 37, 1038-1040.

25.   Turing, A. M. Computing machinery and intelligence. In *Parsing the Turing Test*; Springer: 2009, pp 23-65.

26.   Chollet, F., *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP-Verlags GmbH & Co. KG: 2018.

27.   Segler, M. H.; Preuss, M.; Waller, M. P., Planning chemical syntheses with deep neural networks and symbolic AI. *Nature* **2018**, 555, 604-610.

28.   Lipinski, C. A., Rule of five in 2015 and beyond: Target and ligand structural limitations, ligand chemistry structure and drug discovery project decisions. *Advanced drug delivery reviews* **2016**, 101, 34-41.

29.   Bian, Y.; Jing, Y.; Wang, L.; Ma, S.; Jun, J. J.; Xie, X.-Q., Prediction of orthosteric and allosteric regulations on cannabinoid receptors using supervised machine learning classifiers. *Molecular pharmaceutics* **2019**, 16, 2605-2615.

30.   Lo, Y.-C.; Rensi, S. E.; Torng, W.; Altman, R. B., Machine learning in chemoinformatics and drug discovery. *Drug discovery today* **2018**, 23, 1538-1546.

31.   Jing, Y.; Bian, Y.; Hu, Z.; Wang, L.; Xie, X.-Q. S., Deep learning for drug design: an artificial intelligence paradigm for drug discovery in the big data era. *The AAPS journal* **2018**, 20, 58.

32.   Bzdok, D.; Altman, N.; Krzywinski, M., In; Nature Publishing Group: 2018.

33.   Vamathevan, J.; Clark, D.; Czodrowski, P.; Dunham, I.; Ferran, E.; Lee, G.; Li, B.; Madabhushi, A.; Shah, P.; Spitzer, M., Applications of machine learning in drug discovery and development. *Nature Reviews Drug Discovery* **2019**, 18, 463-477.

34.   Korotcov, A.; Tkachenko, V.; Russo, D. P.; Ekins, S., Comparison of deep learning with multiple machine learning methods and metrics using diverse drug discovery data sets. *Molecular pharmaceutics* **2017**, 14, 4462-4475.

35.   Ma, X. H.; Jia, J.; Zhu, F.; Xue, Y.; Li, Z. R.; Chen, Y. Z., Comparative analysis of machine learning methods in ligand-based virtual screening of large compound libraries. *Combinatorial chemistry & high throughput screening* **2009**, 12, 344-357.

36.   Verma, J.; Khedkar, V. M.; Coutinho, E. C., 3D-QSAR in drug design-a review. *Current topics in medicinal chemistry* **2010**, 10, 95-115.

37.   Fan, F.; Warshaviak, D. T.; Hamadeh, H. K.; Dunn, R. T., The integration of pharmacophore-based 3D QSAR modeling and virtual screening in safety profiling: A case study to identify antagonistic activities against adenosine receptor, A2A, using 1,897 known drugs. *PloS one* **2019**, 14.

38.   Gladysz, R.; Dos Santos, F. M.; Langenaeker, W.; Thijs, G.; Augustyns, K.; De Winter, H., Spectrophores as one-dimensional descriptors calculated from three-dimensional atomic properties: applications ranging from scaffold hopping to multi-target virtual screening. *Journal of cheminformatics* **2018**, 10, 9.

39.   Nguyen, T. T.; Nguyen, N. D.; Nahavandi, S., Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE Transactions on Cybernetics* **2020**.

40.   LeCun, Y.; Bengio, Y.; Hinton, G., Deep learning. *nature* **2015**, 521, 436-444.

41.   Goodfellow, I.; Bengio, Y.; Courville, A., *Deep learning*. MIT press: 2016.

42.   Kleene, S. C. *Representation of events in nerve nets and finite automata*; RAND PROJECT AIR FORCE SANTA MONICA CA: 1951.

43.   LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P., Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **1998**, 86, 2278-2324.

44.   Rumelhart, D. E.; Hinton, G. E.; Williams, R. J., Learning representations by back-propagating errors. *nature* **1986**, 323, 533-536.

45.   Hochreiter, S.; Schmidhuber, J., Long short-term memory. *Neural computation* **1997**, 9, 1735-1780.

46.   Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Advances in neural information processing systems, 2014; 2014; pp 2672-2680.

47.   Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E., The protein data bank. *Nucleic acids research* **2000**, 28, 235-242.

48.   UniProt: the universal protein knowledgebase. *Nucleic acids research* **2017**, 45, D158-D169.

49.   Kim, S.; Thiessen, P. A.; Bolton, E. E.; Chen, J.; Fu, G.; Gindulyte, A.; Han, L.; He, J.; He, S.; Shoemaker, B. A., PubChem substance and compound databases. *Nucleic acids research* **2016**, 44, D1202-D1213.

50.   Gaulton, A.; Hersey, A.; Nowotka, M.; Bento, A. P.; Chambers, J.; Mendez, D.; Mutowo, P.; Atkinson, F.; Bellis, L. J.; Cibrián-Uhalte, E., The ChEMBL database in 2017. *Nucleic acids research* **2017**, 45, D945-D954.

51.   Wishart, D. S.; Feunang, Y. D.; Guo, A. C.; Lo, E. J.; Marcu, A.; Grant, J. R.; Sajed, T.; Johnson, D.; Li, C.; Sayeeda, Z., DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic acids research* **2018**, 46, D1074-D1082.

52.   Irwin, J. J.; Shoichet, B. K., ZINC– a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling* **2005**, 45, 177-182.

53.   Huang, Z.; Mou, L.; Shen, Q.; Lu, S.; Li, C.; Liu, X.; Wang, G.; Li, S.; Geng, L.; Liu, Y., ASD v2. 0: updated content and novel features focusing on allosteric regulation. *Nucleic acids research* **2014**, 42, D510-D516.

54.   Nidhi; Glick, M.; Davies, J. W.; Jenkins, J. L., Prediction of biological targets for compounds using multiple-category Bayesian models trained on chemogenomics databases. *Journal of chemical information and modeling* **2006**, 46, 1124-1133.

55.   Wang, L.; Ma, C.; Wipf, P.; Liu, H.; Su, W.; Xie, X.-Q., TargetHunter: an in silico target identification tool for predicting therapeutic potential of small organic molecules based on chemogenomic database. *The AAPS journal* **2013**, 15, 395-406.

56.   Weininger, D., SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of chemical information and computer sciences* **1988**, 28, 31-36.

57.   OEChem, T., OpenEye Scientific Software. *Inc., Santa Fe, NM, USA* **2012**.

58.   Landrum, G., RDKit: Open-source cheminformatics. **2006**.

59.   O'Boyle, N. M., Towards a Universal SMILES representation-A standard method to generate canonical SMILES based on the InChI. *Journal of cheminformatics* **2012**, 4, 22.

60.   Durant, J. L.; Leland, B. A.; Henry, D. R.; Nourse, J. G., Reoptimization of MDL keys for use in drug discovery. *Journal of chemical information and computer sciences* **2002**, 42, 1273-1280.

61.   Rogers, D.; Hahn, M., Extended-connectivity fingerprints. *Journal of chemical information and modeling* **2010**, 50, 742-754.

62.   Hert, J.; Willett, P.; Wilton, D. J.; Acklin, P.; Azzaoui, K.; Jacoby, E.; Schuffenhauer, A., Comparison of fingerprint-based methods for virtual screening using multiple bioactive reference structures. *Journal of chemical information and computer sciences* **2004**, 44, 1177-1185.

63.   Bian, Y.; Wang, J.; Jun, J. J.; Xie, X.-Q., Deep convolutional generative adversarial network (dcGAN) models for screening and design of small molecules targeting cannabinoid receptors. *Molecular pharmaceutics* **2019**, 16, 4451-4460.

64.   Elton, D. C.; Boukouvalas, Z.; Fuge, M. D.; Chung, P. W., Deep learning for molecular design—a review of the state of the art. *Molecular Systems Design & Engineering* **2019**, 4, 828-849.

65.   Wang, R.; Fang, X.; Lu, Y.; Yang, C.-Y.; Wang, S., The PDBbind database: methodologies and updates. *Journal of medicinal chemistry* **2005**, 48, 4111-4119.

66.   Papadatos, G.; Davies, M.; Dedman, N.; Chambers, J.; Gaulton, A.; Siddle, J.; Koks, R.; Irvine, S. A.; Pettersson, J.; Goncharoff, N., SureChEMBL: a large-scale, chemically annotated patent document database. *Nucleic acids research* **2016**, 44, D1220-D1228.

67.   Gilson, M. K.; Liu, T.; Baitaluk, M.; Nicola, G.; Hwang, L.; Chong, J., BindingDB in 2015: a public database for medicinal chemistry, computational chemistry and systems pharmacology. *Nucleic acids research* **2016**, 44, D1045-D1053.

68.   Ruddigkeit, L.; Van Deursen, R.; Blum, L. C.; Reymond, J.-L., Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *Journal of chemical information and modeling* **2012**, 52, 2864-2875.

69.   Heller, S. R.; McNaught, A.; Pletnev, I.; Stein, S.; Tchekhovskoi, D., InChI, the IUPAC international chemical identifier. *Journal of cheminformatics* **2015**, 7, 23.

70.   Glen, R. C.; Bender, A.; Arnby, C. H.; Carlsson, L.; Boyer, S.; Smith, J., Circular fingerprints: flexible molecular descriptors with applications from physical chemistry to ADME. *IDrugs* **2006**, 9, 199.

71.   Pérez-Nueno, V. I.; Rabal, O.; Borrell, J. I.; Teixidó, J., APIF: a new interaction fingerprint based on atom pairs and its application to virtual screening. *Journal of chemical information and modeling* **2009**, 49, 1245-1260.

72.   O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R., Open Babel: An open chemical toolbox. *Journal of cheminformatics* **2011**, 3, 33.

73.   Willighagen, E. L.; Mayfield, J. W.; Alvarsson, J.; Berg, A.; Carlsson, L.; Jeliazkova, N.; Kuhn, S.; Pluskal, T.; Rojas-Chertó, M.; Spjuth, O., The Chemistry Development Kit (CDK) v2. 0: atom typing, depiction, molecular formulas, and substructure searching. *Journal of cheminformatics* **2017**, 9, 33.

74.   Ambure, P.; Aher, R. B.; Roy, K. Recent advances in the open access cheminformatics toolkits, software tools, workflow environments, and databases. In *Computer-Aided Drug Discovery*; Springer: 2014, pp 257-296.

75.   Arabie, P.; Baier, N. D.; Critchley, C. F.; Keynes, M., Studies in Classification, Data Analysis, and Knowledge Organization. **2006**.

76.   Warr, W. A., Scientific workflow systems: Pipeline Pilot and KNIME. *Journal of computer-aided molecular design* **2012**, 26, 801-804.

77.   Beisken, S.; Meinl, T.; Wiswedel, B.; de Figueiredo, L. F.; Berthold, M.; Steinbeck, C., KNIME-CDK: Workflow-driven cheminformatics. *BMC bioinformatics* **2013**, 14, 257.

78.   Saubern, S.; Guha, R.; Baell, J. B., KNIME workflow to assess PAINS filters in SMARTS format. Comparison of RDKit and Indigo cheminformatics libraries. *Molecular informatics* **2011**, 30, 847-850.

79.   Roughley, S. D., Five Years of the KNIME Vernalis Cheminformatics Community Contribution. *Current medicinal chemistry* **2019**.

80.   Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M. Tensorflow: A system for large-scale machine learning. In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), 2016; 2016; pp 265-283.

81.   Etaati, L. Deep Learning Tools with Cognitive Toolkit (CNTK). In *Machine Learning with Microsoft Technologies*; Springer: 2019, pp 287-302.

82.   Team, T.; Al-Rfou, R.; Alain, G.; Almahairi, A.; Angermueller, C.; Bahdanau, D.; Ballas, N.; Bastien, F.; Bayer, J.; Belikov, A.; Belopolsky, A.; Bengio, Y.; Bergeron, A.; Bergstra, J.; Bisson, V.; Bleecher Snyder, J.; Bouchard, N.; Boulanger-Lewandowski, N.; Bouthillier, X.; Zhang, Y., Theano: A Python framework for fast computation of mathematical expressions. **2016**.

83.   Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L. PyTorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems, 2019; 2019; pp 8024-8035.

84.   Chollet, F., In; 2015.

85.   Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V., Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* **2011**, 12, 2825-2830.

86.   Mikolov, T.; Karafiát, M.; Burget, L.; Černocký, J.; Khudanpur, S. Recurrent neural network based language model. In Eleventh annual conference of the international speech communication association, 2010; 2010.

87.   Mikolov, T.; Kombrink, S.; Burget, L.; Černocký, J.; Khudanpur, S. Extensions of recurrent neural network language model. In 2011 IEEE international conference on acoustics, speech and signal processing (ICASSP), 2011; IEEE: 2011; pp 5528-5531.

88.   Mikolov, T.; Zweig, G. Context dependent recurrent neural network language model. In 2012 IEEE Spoken Language Technology Workshop (SLT), 2012; IEEE: 2012; pp 234-239.

89.   Hanson, J.; Yang, Y.; Paliwal, K.; Zhou, Y., Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks. *Bioinformatics* **2017**, 33, 685-692.

90.   Cheng, J.; Dong, L.; Lapata, M., Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733* **2016**.

91.   Gupta, A.; Müller, A. T.; Huisman, B. J.; Fuchs, J. A.; Schneider, P.; Schneider, G., Generative recurrent networks for de novo drug design. *Molecular informatics* **2018**, 37, 1700111.

92.   Bian, Y.; Xie, X.-Q. S., Computational fragment-based drug design: Current trends, strategies, and applications. *The AAPS journal* **2018**, 20, 59.

93.   Segler, M. H.; Kogej, T.; Tyrchan, C.; Waller, M. P., Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science* **2018**, 4, 120-131.

94.   Moret, M.; Friedrich, L.; Grisoni, F.; Merk, D.; Schneider, G., Generative molecular design in low data regimes. *Nature Machine Intelligence* **2020**, 2, 171-180.

95.   Merk, D.; Friedrich, L.; Grisoni, F.; Schneider, G., De novo design of bioactive small molecules by artificial intelligence. *Molecular informatics* **2018**, 37, 1700153.

96.   Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y., Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* **2014**.

97.   Zheng, S.; Yan, X.; Gu, Q.; Yang, Y.; Du, Y.; Lu, Y.; Xu, J., QBMG: quasi-biogenic molecule generator with deep recurrent neural network. *Journal of cheminformatics* **2019**, 11, 5.

98.   Kramer, M. A., Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal* **1991**, 37, 233-243.

99.   Kingma, D. P.; Welling, M., Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* **2013**.

100.   Kingma, D. P.; Welling, M., An introduction to variational autoencoders. *arXiv preprint arXiv:1906.02691* **2019**.

101.   Kingma, D. P.; Mohamed, S.; Rezende, D. J.; Welling, M. Semi-supervised learning with deep generative models. In Advances in neural information processing systems, 2014; 2014; pp 3581-3589.

102.   Khemakhem, I.; Kingma, D. P.; Hyvärinen, A., Variational autoencoders and nonlinear ica: A unifying framework. *arXiv preprint arXiv:1907.04809* **2019**.

103.   Pu, Y.; Gan, Z.; Henao, R.; Yuan, X.; Li, C.; Stevens, A.; Carin, L. Variational autoencoder for deep learning of images, labels and captions. In Advances in neural information processing systems, 2016; 2016; pp 2352-2360.

104.   Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A., Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science* **2018**, 4, 268-276.

105.   Blaschke, T.; Olivecrona, M.; Engkvist, O.; Bajorath, J.; Chen, H., Application of generative autoencoder in de novo molecular design. *Molecular informatics* **2018**, 37, 1700123.

106.   Sattarov, B.; Baskin, I. I.; Horvath, D.; Marcou, G.; Bjerrum, E. J.; Varnek, A., De novo molecular design by combining deep autoencoder recurrent neural networks with generative topographic mapping. *Journal of chemical information and modeling* **2019**, 59, 1182-1196.

107.   Samanta, B.; Abir, D.; Jana, G.; Chattaraj, P. K.; Ganguly, N.; Rodriguez, M. G. Nevae: A deep generative model for molecular graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, 2019; 2019; Vol. 33; pp 1110-1117.

108.   Simonovsky, M.; Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. In International Conference on Artificial Neural Networks, 2018; Springer: 2018; pp 412-422.

109.   Imrie, F.; Bradley, A. R.; van der Schaar, M.; Deane, C. M., Deep Generative Models for 3D Linker Design. *Journal of Chemical Information and Modeling* **2020**, 60, 1983-1995.

110.   Makhzani, A.; Shlens, J.; Jaitly, N.; Goodfellow, I.; Frey, B., Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* **2015**.

111.   Kadurin, A.; Nikolenko, S.; Khrabrov, K.; Aliper, A.; Zhavoronkov, A., druGAN: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Molecular pharmaceutics* **2017**, 14, 3098-3104.

112.   Polykovskiy, D.; Zhebrak, A.; Vetrov, D.; Ivanenkov, Y.; Aladinskiy, V.; Mamoshina, P.; Bozdaganyan, M.; Aliper, A.; Zhavoronkov, A.; Kadurin, A., Entangled conditional adversarial autoencoder for de novo drug discovery. *Molecular pharmaceutics* **2018**, 15, 4398-4405.

113.   Shayakhmetov, R.; Kuznetsov, M.; Zhebrak, A.; Kadurin, A.; Nikolenko, S.; Aliper, A.; Polykovskiy, D., Molecular Generation for Desired Transcriptome Changes With Adversarial Autoencoders. *Frontiers in Pharmacology* **2020**, 11, 269.

114.   Guimaraes, G. L.; Sanchez-Lengeling, B.; Outeiral, C.; Farias, P. L. C.; Aspuru-Guzik, A., Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843* **2017**.

115.   Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; Warchoł, M., Mol-CycleGAN: a generative model for molecular optimization. *Journal of Cheminformatics* **2020**, 12, 1-18.

116.   Méndez-Lucio, O.; Baillif, B.; Clevert, D.-A.; Rouquié, D.; Wichard, J., De novo generation of hit-like molecules from gene expression signatures using artificial intelligence. *Nature Communications* **2020**, 11, 1-10.

117.   Prykhodko, O.; Johansson, S. V.; Kotsias, P.-C.; Arús-Pous, J.; Bjerrum, E. J.; Engkvist, O.; Chen, H., A de novo molecular generation method using latent vector based generative adversarial network. *Journal of Cheminformatics* **2019**, 11, 74.

118.   Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K. Q. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2017; 2017; pp 4700-4708.

119.   LeCun, Y.; Bengio, Y., Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* **1995**, 3361, 1995.

120.   Yu, D.; Wang, H.; Chen, P.; Wei, Z. Mixed pooling for convolutional neural networks. In International conference on rough sets and knowledge technology, 2014; Springer: 2014; pp 364-375.

121.   Radford, A.; Metz, L.; Chintala, S., Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* **2015**.

122.   Zhang, H.; Goodfellow, I.; Metaxas, D.; Odena, A., Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318* **2018**.

123.   Li, C.; Wand, M. Precomputed real-time texture synthesis with markovian generative adversarial networks. In European conference on computer vision, 2016; Springer: 2016; pp 702-716.

124.   Gao, W.; Coley, C. W., The synthesizability of molecules proposed by generative models. *Journal of Chemical Information and Modeling* **2020**.

125.   Coley, C. W.; Rogers, L.; Green, W. H.; Jensen, K. F., SCScore: synthetic complexity learned from a reaction corpus. *Journal of chemical information and modeling* **2018**, 58, 252-261.

126.   Vargesson, N., Thalidomide‐induced teratogenesis: History and mechanisms. *Birth Defects Research Part C: Embryo Today: Reviews* **2015**, 105, 140-156.

127.   Polishchuk, P. G.; Madzhidov, T. I.; Varnek, A., Estimation of the size of drug-like chemical space based on GDB-17 data. *Journal of computer-aided molecular design* **2013**, 27, 675-679.

128.   Alley, E. C.; Khimulya, G.; Biswas, S.; AlQuraishi, M.; Church, G. M., Unified rational protein engineering with sequence-based deep representation learning. *Nature methods* **2019**, 16, 1315-1322.