ChemML: A Machine Learning and Informatics Program Package for the Analysis, Mining, and Modeling of Chemical and Materials Data

Mojtaba Haghighatlari,[1,*] Gaurav Vishwakarma,[1] Doaa Altarawy,[2,3] Ramachandran Subramanian,[4,5] Bhargava Urala Kota,[4,5] Aditya Sonpal,[1] Srirangaraj Setlur,[4,5,6] and Johannes Hachmann[1,7,8,†]

[1] *Department of Chemical and Biological Engineering, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[2] *The Molecular Sciences Software Institute, Virginia Tech, Blacksburg, VA 24060, United States*
[3] *Computer and Systems Engineering Department, Alexandria University, Alexandria 21544, Egypt*
[4] *Department of Computer Science and Engineering, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[5] *Center for Unified Biometrics and Sensors, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[6] *Center of Excellence for Document Analysis and Recognition, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[7] *Computational and Data-Enabled Science and Engineering Graduate Program, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[8] *New York State Center of Excellence in Materials Informatics, Buffalo, NY 14203, United States*

ChemML is an open machine learning and informatics program suite that is designed to support and advance the data-driven research paradigm that is currently emerging in the chemical and materials domain. *ChemML* allows its users to perform various data science tasks and execute machine learning workflows that are adapted specifically for the chemical and materials context. Key features are automation, general-purpose utility, versatility, and user-friendliness in order to make the application of modern data science a viable and widely accessible proposition in the broader chemistry and materials community. *ChemML* is also designed to facilitate methodological innovation, and it is one of the cornerstones of the software ecosystem for data-driven *in silico* research outlined in Ref. [1].

I. INTRODUCTION

Since the early days of the digital and information age, data science has been transforming every domain discipline, and the chemical and materials fields are no longer an exception [2]. Data-driven research and machine learning (ML) have emerged as promising new thrusts in materials and chemical research, in particular in the wake of the 2011 White House Materials Genome Initiative (MGI) [3] and the 2017 NSF Data-Driven Discovery Science in Chemistry (D3SC) initiative [4], respectively. This new paradigm is causing much excitement for its unique potential to unravel hidden structure-property relationships that govern the behavior of chemical and materials systems, as well as for its ability to yield data-derived surrogate models that are dramatically more efficient than traditional physics-based models [5]. However, as data-driven research is still a young and less-well-established approach, there is a distinct infrastructure gap (especially in chemistry) [6]. Tools and techniques that could facilitate it are often in their early stages or lacking altogether.

As outlined in Ref. [1], we have been developing a software ecosystem that combines computational modeling, virtual high-throughput screening, and big data analytics to tackle this issue. It is comprised of *ChemLG* [7], a library generator for the definition and enumeration of chemical and materials spaces; *ChemHTPS* [8], an automated high-throughput *in silico* screening platform for chemical and materials data generation; *ChemBDDB* [9], a big data database template for chemical and materials data storage; and *ChemML* [10] , an ML and informatics program package for chemical and materials data mining.

The paper at hand provides a detailed introduction of *ChemML* and our development efforts since 2014. In Sec. II, we will explain the purpose and design philosophy that *ChemML* is built upon and Sec. III discusses the classes of core tasks at its heart. Sec. IV covers *ChemML*'s architecture and implementation, Sec. V provides a description of methodological advancements and summarizes applications of *ChemML*, and Sec. VI offers conclusions and an outlook on future developments.

II. DESIGN PHILOSOPHY

ChemML is designed as a toolbox for the use of data science in the chemical and materials domain. Its capabilities include the validation, analysis, mining, and modeling of chemical and materials data sets (both of modest and large size, including those generated by *ChemHTPS*) using state-of-the-art ML and informatics methodology. Its primary purpose is to extract hidden structure-property relationships, which are a prerequisite for the generation of data-derived prediction models, hyperscreening studies, accelerated discovery, rational de-

* mojtabah@buffalo.edu
† hachmann@buffalo.edu

sign, and inverse engineering.

While many individual components of data science workflows are openly available, there is currently no comprehensive end-to-end solution that is ready for use in real-world, every-day work of chemists and material scientists, in particular not without expert training and coding efforts. *ChemML* adapts, interfaces, and integrates techniques as well as expertise from chemistry, materials, and data science to create a value-added new tool that enables and advances data-driven research. A key consideration in *ChemML*'s design is to prioritize accessibility to non-expert users, so that ML methodology can readily be employed by interested research practitioners. *ChemML* provides abstracted ML workflows for general-purpose use, broad scope, and wide range of applicability combined with automation and user-friendliness. At the same time, we allow all settings to be tailored and customized to keep *ChemML* relevant for more experienced users. *ChemML* is open source and distributed under the permissive 3-clause BSD license.

ChemML is written in Python and its design and implementation follows that of an open-software infrastructure, allowing for a sustainable development, evolution, and deployment. It is strictly modular and object-oriented, and it takes advantage of available open-source libraries to avoid 'reinventing the wheel' and to harness their excellent performance. In addition to serving as a production-level tool, *ChemML* is also designed as a development platform and testbed for ML methods and techniques. It allows us to systematically, rapidly, and efficiently test the efficacy and performance of available methodology in the chemical context as well as of original contributions that we are introducing in the course of our work.

With this overall design philosophy we seek to reproduce the game-changing influence that the rise of computational chemistry program packages with similar characteristics have had on making modeling and simulation such ubiquitous techniques in today's research landscape.

III. CORE TASKS

A typical supervised ML scenario, which *ChemML* is designed to tackle, is the creation of a data-derived prediction model [5]. The latter can be thought of as a complex mathematical operation $f : X \to Y$ that learns a mapping from input $x \in X$ onto an output $y \in Y$, where x in this context is the feature representation of a chemical or materials system and y is its target property (or other characteristic). If the variable y is continuous (numerical), then the mapping is a regression; if it is discrete (categorical), then it is a classification. An ML algorithm defines model f and optimizes all its parameters to approximate the output y for a given input x, i.e., the algorithm minimizes the prediction error of a model by training it on available data. We can also express the feature representation as a function $g : M \to X$. It maps a compound's basic chemical representation $m \in M$ to the feature input x (i.e., a set of descriptors). A typical ML workflow encompasses a number of steps [11], including the parsing, cleaning, and preprocessing of a chemical data set $\{M, Y\}$; the generation of an appropriate feature representation *via* g; and the training, validation, and evaluation of model f. All these basic components and more sophisticated ones are provided by *ChemML* in the form of canned methods for ease of use, similar to domain-independent libraries such as scikit-learn [12].

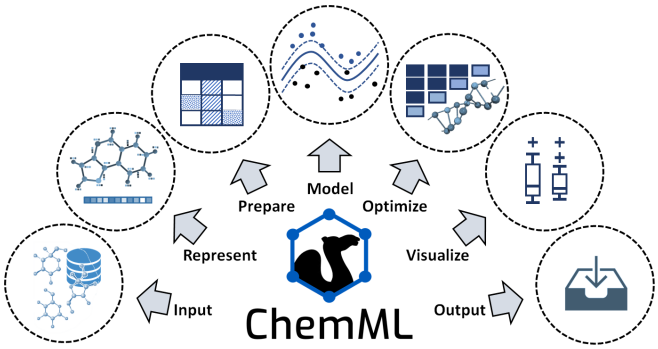

FIG. 1. *ChemML* provides a multitude of methods as part of seven core task classes to conduct data mining projects (such as the creation of data-derived machine learning (ML) surrogate models).

In the following, we give a brief overview of the types of methods that are covered by each of the core task classes shown in Fig. 1, and we mention notable innovation as well as key third-party libraries *ChemML* utilizes in these tasks:

Input/Output (I/O): *ChemML* provides several methods to read and parse input files (e.g., containing data sets, libraries of molecules, or trained models) in various formats and store the outputs of *ChemML*'s operations. *ChemML* uses Numpy and Pandas libraries for the majority of basic data manipulation tasks. It provides three main classes to manipulate the chemical information of molecular or crystal structures. These classes are supported by the popular cheminformatics codes OpenBabel [13], RDKit [14], and Magpie [15].

Represent: The choice of feature representation is one of the most important aspects of an ML approach and an obvious place to infuse the physics of a given data set into the desired model. *ChemML* includes implementations of numerous feature representation methods, including Coulomb matrix [16], bag-of-bonds [17], local atomic and bond features for deep learning (DL) [18], interfaces for RDKit fingerprints and molecular descriptors from Dragon [19], as well as a Python reimplementation of all composition-based and crystal-structure-based features available in the Magpie library. It also covers our original work on fingerprint and feature engineering.

Prepare: The data preparation techniques in *ChemML* allow us to alleviate issues associated with one-

to-many and many-to-one mappings. They also allow us to remove redundant or otherwise irrelevant features using feature transformation and feature selection techniques.

Model: To date, our focus in developing *ChemML* has been on supervised ML techniques. For core ML tasks in the creation of these models, we utilize popular and efficient libraries including scikit-learn, Tensorflow [20], and Keras [21]. This task class also includes crucial facilities for model assessment, validation, and evaluation. The original model contributions developed and maintained within *ChemML* include physics-informed DL architectures and pre-tuned ML models for the prediction of particular molecular properties.

Optimize: This task class provides methods to quantify and improve both the accuracy and reliability of predictions. We can optimize a given model in hyperparameter space using a coarse grid search or evolutionary algorithm. In addition, ML design methodologies such as active learning (AL) and transfer learning (TL) are available to enhance the efficiency of the exploration of compound space. (We detail the implementation of these techniques as examples for methodological innovation in the Sec. V.) The available optimization methods link to other elements of an ML workflow and *ChemML* allows us to automatize the modeling of specified search spaces.

Visualize: Data visualization methods that help facilitate a better comprehension of the modeling results are available *via* a separate module, which builds on the Matplotlib and Seaborn libraries for key visualization elements. These elements are accessible as part of any ML workflow.

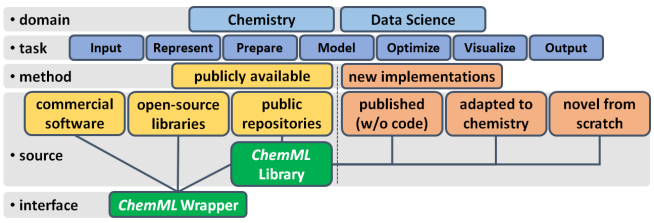

FIG. 2. Design scheme and architecture of the *ChemML* program package, consisting of *ChemML* Library and *ChemML* Wrapper. The overview shows the seven task classes and the methods available within them, as well as the six categories we distinguish with regards to the method sources.

IV. ARCHITECTURE AND IMPLEMENTATION

ChemML incorporates a host of methods to support the core tasks introduced in the previous section and it stages them as constituent elements of ML workflows. As shown in Fig. 2, we distinguish methods for which an implementation is publicly available and those for which it is not. The former include commercial software, open-source libraries, and public code repositories, and they are either directly embedded (in particular in case of the core libraries) or accessed through plug-in interfaces. The latter include methods by others that are published in the literature, but for which no (public) implementation is available, as well as new methods that are being developed by us as original contributions. In the original contributions we distinguish between established methods that we adapt to the chemistry context and entirely novel methods we develop from scratch. All newly implemented code forms *ChemML* Library. *ChemML* Library also includes public code repositories that do not feature a standard structure and are thus not readily amenable for bindings. Parallel processing is available for the majority of these methods to leverage multiprocessors hardware architectures and accelerate large-scale studies. *ChemML* Library and all available third-party libraries and codes are tied together *via* *ChemML* Wrapper, which serves as the overarching framework and driver for the *ChemML* package. *ChemML* Wrapper enables us to build and run arbitrarily complex workflows comprised of the available methods. Its workflow prototyping capability is comparable to the KNIME platform [22], however, it also provides a high-level interface to many libraries (including *ChemML*), and it is specifically tailored for chemical and materials research. In contrast to common symbolic programming approaches (e.g., the Tensorflow architecture), *ChemML* Wrapper does not require the user to code.

As shown in Fig. 3, the ML workflows correspond to computation graphs with nodes and edges. The nodes represent computation units (i.e., the available methods), and the edges the flow of data between the connected nodes *via* embedded input/output tokens. The computation graphs can be generated by means of input files or a Jupyter notebook graphical user interface (GUI). *ChemML* Wrapper also provides a library of abstracted workflow templates for common ML problem settings and of previous studies by us and others. These templates can be used as included or as starting points for the creation of modified and customized workflows. *ChemML* Wrapper offers safeguards, sanity checks, and warnings if a new workflow is inconsistent with best practices or inappropriate for a given data set, or if a trained model is applied to make predictions outside its applicability domain. *ChemML* Wrapper also provides statistical analyses of prediction results and meta-ML facilities to establish guidelines and the foundations for an expert recommendation system for a field that cannot draw on decades of experience (and thus often resorts to suboptimal *ad hoc* choices).

As *ChemML* Wrapper is tasked with providing extensibility, consistency, and flexible glue code functionality, it faces two main challenges, i.e., licenses and maintenance. To address the former, *ChemML* Wrapper only binds to the front-end of external packages (e.g., Dragon, RDKit, OpenBabel) and provides rea-

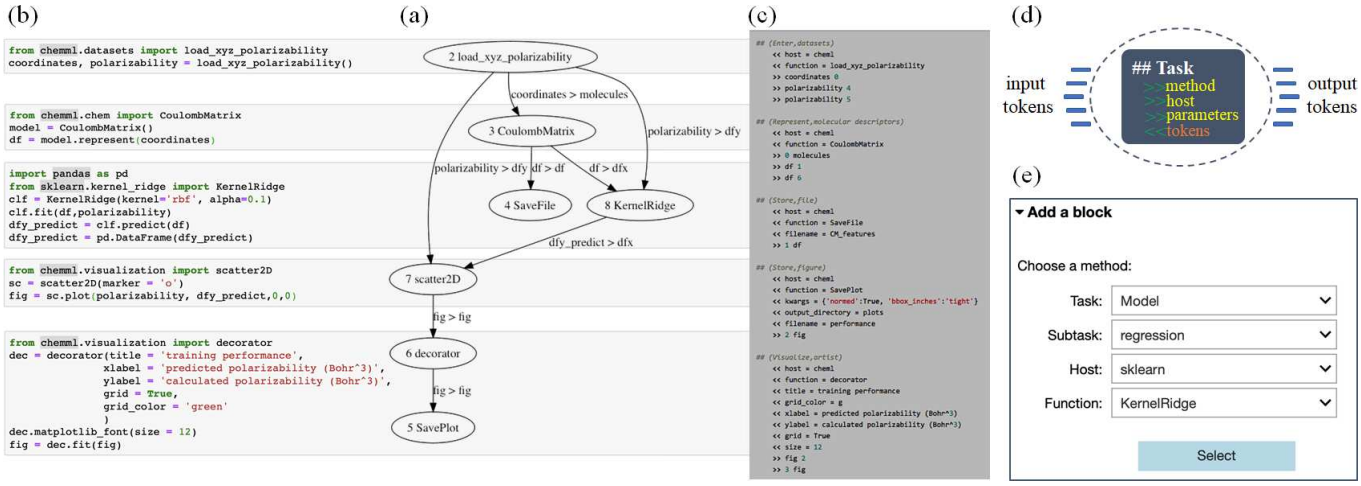

FIG. 3. Toy workflow/computation graph used in *ChemML*. (a) Plot of the computation graph with corresponding Python code (b) and input file (c); (d) shows the structure of a computation unit and (e) an example of its implementation in the Jupyter notebook graphical user interface (GUI). We stress that actual ML workflows are considerably more involved than this simplistic toy example.

sonable automation and support for licensing, installation, and citation issues. We tackle the maintainability challenge by bundling the external open-source libraries, and creating a central database of method and interface attributes (i.e., the meta data) to (semi-)automatically modify *ChemML* Wrapper (including its documentation pages and web application widgets) after each update of the dependencies. Moreover, all methods accessible by *ChemML* Wrapper are tagged by their package and task names, and they are imported on demand.

ChemML's modular structure enables its role as a development platform and testbed for methodological innovation. *ChemML* Library makes it easy to add new methods, and *ChemML* Wrapper's flexibility allows us to bypass the tedious process of testing new ideas by writing standalone prototype implementations. *ChemML* Wrapper offers facilities for the automated testing and benchmarking of these new methods against many existing ones with essentially no overhead in human time. Those that are competitive or that offer other benefits (e.g., interpretability, efficiency) become part of the release branch of *ChemML* Library and are thus directly accessible to the user.

V. METHODOLOGICAL INNOVATION AND APPLICATION STUDIES

Adapting ML techniques and algorithms from other application domains and introducing entirely new approaches for the study of chemical and material problems requires a substantial rethinking and redevelopment of existing ML methodology [23–26]. Our recent development efforts include methods that improve accuracy and efficiency by closing the loop of data modeling and molecular design, i.e., by going beyond the mere creation of ML prediction models. The following examples are three highlights of new classes of methods:

Active learning (AL): *ChemML* provides AL methods [27] that allow us to minimize the data set size needed to train a data-derived prediction model of a given accuracy or maximize its accuracy for a given number of data points. This is an important capability as data generation (either *via* experiments and physics-based modeling) is often expensive, making it a common bottleneck in data-driven research. AL is based on smart and deliberate sampling that identifies data points with the highest information yield and thus impact on the training of an ML model. For this, it assesses the uncertainty of a given model with respect to its training input. *ChemML* provides a number of different model-based AL approaches (including the query-by-committee and expected-model-change strategies [28]) that can be applied on a pool of unlabeled data points (see Fig. 4). The *ChemML* implementation supports any type of DL model that is built using the Keras interface. It also supports batch selection of data points based on the estimated impact of previously selected data with no extra query or training overhead. An interactive implementation allows the user to deposit or ignore queries based on direct feedback from experiment or simulation. The available AL approaches can be used simultaneously with no extra cost.

Transfer learning (TL): *ChemML* provides TL methods [29] that allow us, e.g., to generate high-quality data-derived prediction models using a combination of lower-quality (cheap) training data and a small set of high-quality (expensive) data that by itself would be insufficient for this task. This is another approach to alleviate the data generation issue as a limiting factor by reducing its cost or the number of data points needed to obtain an ML model of a desired accuracy. TL essentially allows us to learn the mapping from a lower-quality

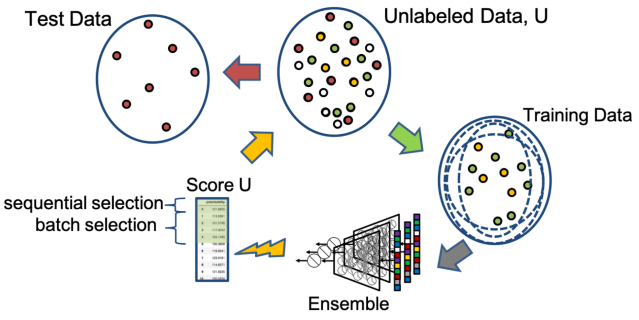

FIG. 4. The process of applying a model-based active learning (AL) strategy to a pool of unlabeled candidates as implemented in *ChemML*. After initializing the training and test sets randomly, an ensemble of deep learning (DL) models score unlabeled candidates in a sequential or batch selection format to query one or several new data points that promise to improve the model the most.

to a high-quality model and transfer the tuned parameters from the former to the latter (see Fig. 5). *ChemML* features an implementation of TL design methodologies for DL models to facilitate this type of task. We employ uncertainty qualification as a measure of prediction confidence and analyses of distribution shifts for the retraining of models.

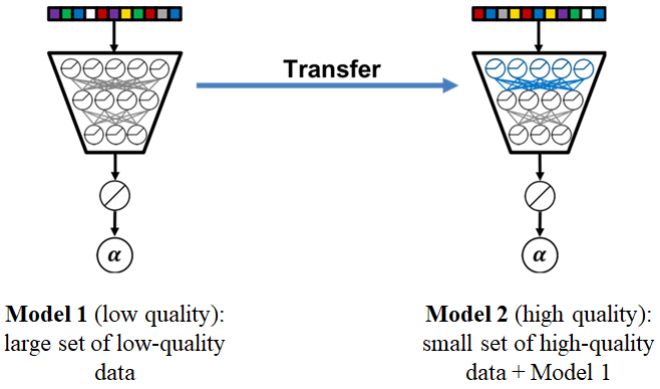

FIG. 5. The process of applying transfer learning (TL) from a DL model 1 for target property α based on a large set of lower-quality data to a more accurate model 2 based on a small set of high-quality data.

Auto machine learning (AML): One of the critical steps in any ML approach is the selection of hyperparameters that define the employed model and its components (cf. Fig. 1). The choice of hyperparameters (in particular those associated with representation, preparation, and modeling) for a given problem setting can have a dramatic impact on the quality of a resulting model's predictive performance. Given that hyperparameters can assume both discrete and continuous values, the hyperparameter space does not have well-defined gradients, which precludes many common optimization algorithms. Primitive techniques such as brute-force grid searches are available in *ChemML*, but will either remain coarse or become prohibitively demanding due to the curse of dimensionality. *ChemML* also features a fully customizable evolutionary algorithm to automatize the hyperparameter selection [30]. It is implemented *via* a real-coded genetic algorithm that can efficiently navigate the complex hyperparameter space to minimize the prediction errors. *ChemML* provides customizations for each of the genetic operators (crossover, mutation, and selection) to give the user enhanced flexibility and to guide the search in a preferred direction. We use AML as the basis for the before-mentioned meta-ML, i.e., we machine learn best (initial) approaches, settings, defaults, and other recommendations for ML work on particular types of application problems and data sets. These insights allow us to better harness the potential of ML, i.e., by pursuing models with the smallest possible prediction errors and computing time demands.

ChemML features other methodological innovation, e.g., in the areas of physics-infused neural network architectures, learned features, local domain models, training set design, on-the-fly assessment of learning curves [31], chemical pattern recognition [32], etc., that we will describe elsewhere.

We have been employing *ChemML* in a number of real-world application studies, both for the creation of data-derived prediction models and chemical pattern recognition. These studies include discovery and design projects for new high-refractive-index polymers for optical applications [30–35], deep eutectic solvents for supercapacitors [36], and organic semiconductors for photovoltaics and other applications [37, 38] (using data of the Harvard Clean Energy Project [39–43]).

VI. CONCLUSIONS AND OUTLOOK

Data intensive studies are an increasingly common occurrence in the chemistry and materials domain, and many in the community are interested in testing the utility of ML and in incorporating it into their regular work. However, the majority of tools – as far as these exist – is still technically involved, inaccessible, or otherwise unavailable to the community at large. A related concern is the quality and reproducibility of studies that rely on them. In this paper, we introduced the design and major elements of the open-source ML program package *ChemML* that seeks to overcome some of these shortcomings and limitations, and to facilitate the broader dissemination of cutting-edge techniques in an automated, yet flexible and customizable, format. *ChemML* makes an effort to reach non-expert users for which the novelty of ML research may be daunting, and to help share best practices and guidelines.

Our plans for the future development of *ChemML* include adding new methodological contributions as well as

workflows by us and other; augmenting the access to the domain-independent core libraries and their functionality to reflect the rapid advances in this field; further closing the loop between the diverse elements of ML and the domain science [44]; gaining and encapsulating experience in how to make ML work in the chemical and materials context [45]; and further improving user-friendliness (e.g., *via* an interactive GUI, expanded expert system capabilities, and cloud-based deployment). We will also advance the development of the overarching framework *ChemEco* that binds the different components of our software ecosystem [1] together and allows them to interact directly. Our long-term vision is to enable the fully automated exploration of compound space that supports the accelerated discovery and rational design of next-generation chemistry and materials [46, 47].

COMPETING FINANCIAL INTERESTS

The authors declare to have no competing financial interests.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation (NSF) CAREER program (grant No. OAC-1751161), and the New York State Center of Excellence in Materials Informatics (grants No. CMI-1140384 and CMI-1148092). Early work on *ChemML* was supported by start-up funds provided through the University at Buffalo (UB). The deep eutectic solvent application study was funded by the Army Armament Research, Development and Engineering Center (ARDEC) SBIR program (grant No. W15QKN-17-C-0078), and solubility parameter work by Toyota Motor Engineering and Manufacturing North America. *ChemML* is interfaced with the Open Chemistry platform and the MaDE@UB toolkit, and these efforts are supported by the Department of Energy SBIR program (grant No. DE-SC0017193) and the NSF DIBBs program (grant No. OAC-1640867), respectively. The DIBBs grant also funded the implementation of several methods of particular interest for MaDE@UB into *ChemML*, such as the Magpie library, the meta data parser, and standard DNNs using Keras. Computing time on the high-performance computing clusters '*Rush*', '*Alpha*', '*Beta*', and '*Gamma*' was provided by the UB Center for Computational Research (CCR). The work presented in this paper is a central part of MH's PhD thesis [48]. MH gratefully acknowledges support by Phase-I and Phase-II Software Fellowships (grant No. ACI-1547580-479590) of the NSF Molecular Sciences Software Institute (grant No. ACI-1547580) at Virginia Tech [49, 50]. We thank the other members – past and present – of the Hachmann group as well as Profs. Venugopal Govindaraju and Krishna Rajan (both UB) for valuable discussions and insights that have helped guide the development of *ChemML*.

REFERENCES

[1] Hachmann, J.; Afzal, M.; Haghighatlari, M.; Pal, Y. Building and deploying a cyberinfrastructure for the data-driven design of chemical systems and the exploration of chemical space. *Molecular Simulation* **2018**,

[2] Butler, K. T.; Davies, D. W.; Cartwright, H.; Isayev, O.; Walsh, A. Machine learning for molecular and materials science. *Nature* **2018**, *559*, 547–555.

[3] National Science and Technology Council. Materials genome initiative for global competitiveness. *Technical Report* **2011**,

[4] Hachmann, J.; Windus, T. L.; McLean, J. A.; Allwardt, V.; Schrimpe-Rutledge, A. C.; Afzal, M. A. F.; Haghighatlari, M. *Framing the role of big data and modern data science in chemistry*; 2018.

[5] Haghighatlari, M.; Hachmann, J. Advances of Machine Learning in Molecular Modeling and Simulation. *Current Opinion in Chemical Engineering 23*, 51–57.

[6] Wang, H.; Ji, Y.; Li, Y. Simulation and design of energy materials accelerated by machine learning. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2019**, *0*, e1421.

[7] Afzal, M. A. F.; Vishwakarma, G.; Dudwadkar, J. A.; Haghighatlari, M.; Hachmann, J. *ChemLG* – A Program Suite for the Generation of Compound Libraries and the Survey of Chemical Space. 2019; `https://github.com/hachmannlab/chemlg`.

[8] Pal, Y.; Evangelista, W. S.; Afzal, M. A. F.; Haghighatlari, M.; Hachmann, J. *ChemHTPS* – A General-Purpose Computational Chemistry High-Throughput Screening Platform. 2019; `https://github.com/hachmannlab/chemhtps`.

[9] Sonpal, A.; Agrawal, S.; Sivaraj, S.; Hachmann, J. *ChemBDDB* – A Big Data Database Toolkit for Chemical and Materials Data Storage. 2019; `https://github.com/hachmannlab/chembddb`.

[10] Haghighatlari, M.; Hachmann, J. *ChemML* – A machine learning and informatics program suite for chemical and materials data mining. 2019; `https://hachmannlab.github.io/chemml`.

[11] Mitchell, J. B. O. Machine learning methods in chemoinformatics. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2014**, *4*, 468–481.

[12] Pedregosa, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.

[13] 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.

[14] Landrum, G. RDKit: Open-source cheminformatics. `http://www.rdkit.org`.

[15] Ward, L.; Agrawal, A.; Choudhary, A.; Wolverton, C. A general-purpose machine learning framework for predicting properties of inorganic materials. *npj Computational Materials* **2016**, *2*, 16028.

[16] Rupp, M.; Tkatchenko, A.; Müller, K.-R.; von Lilienfeld, O. A. Fast and Accurate Modeling of Molecular

Atomization Energies with Machine Learning. *Physical Review Letters* **2012**, *108*, 058301.

[17] Hansen, K.; Biegler, F.; Ramakrishnan, R.; Pronobis, W.; Von Lilienfeld, O. A.; Müller, K. R.; Tkatchenko, A. Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space. *Journal of Physical Chemistry Letters* **2015**, *6*, 2326–2331.

[18] Duvenaud, D.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. Convolutional Networks on Graphs for Learning Molecular Fingerprints. *Advances in Neural Information Processing Systems* **2015**, 2224–2232.

[19] DRAGON (Software for Molecular Descriptor Calculation). 2011; `http://www.talete.mi.it/`.

[20] Martin, A. et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015; `https://www.tensorflow.org/`.

[21] Chollet, F.; Others, Keras. 2015; `https://keras.io`.

[22] Berthold, M. R.; Cebron, N.; Dill, F.; Gabriel, T. R.; Kötter, T.; Meinl, T.; Ohl, P.; Sieb, C.; Thiel, K.; Wiswedel, B. KNIME: The Konstanz Information Miner. Data Analysis, Machine Learning and Applications. Berlin, Heidelberg, 2008; pp 319–326.

[23] Schütt, K. T.; Arbabzadah, F.; Chmiela, S.; Müller, K.-R.; Tkatchenko, A. Quantum-Chemical Insights from Deep Tensor Neural Networks. *Nature Communications* **2017**, *8*, 13890.

[24] Isayev, O.; Oses, C.; Toher, C.; Gossett, E.; Curtarolo, S.; Tropsha, A. Universal Fragment Descriptors for Predicting Electronic Properties of Inorganic Crystals. *Nature Communications* **2017**, *8*, 15679.

[25] Ferré, G.; Haut, T.; Barros, K. Learning molecular energies using localized graph kernels. *The Journal of Chemical Physics* **2017**, *146*, 114107.

[26] Collins, C. R.; Gordon, G. J.; von Lilienfeld, O. A.; Yaron, D. J. Constant size molecular descriptors for accurate machine learning models of molecular properties. *Journal of Chemical Physics* **2017**, 241718.

[27] Settle, B. *Active Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning*; Morgan & Claypool, 2012; pp 1–114.

[28] Cai, W.; Zhang, M.; Zhang, Y. Batch mode active learning for regression with expected model change. *IEEE Transactions on Neural Networks and Learning Systems* **2017**, *28*, 1668–1681.

[29] Pan, S. J.; Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* **2010**, *22*, 1345–1359.

[30] Vishwakarma, G. Machine Learning Model Selection for Predicting Properties of High-Refractive-Index Polymers. M.Sc. thesis, University at Buffalo, 2018.

[31] Afzal, M. A. F.; Sonpal, A.; Haghighatlari, M.; Schultz, A. J.; Hachmann, J. A Deep Neural Network Model for Packing Density Predictions and its Application in the Study of 1.5 Million Organic Molecules. *ChemRxiv* **2019**, 8217758.v1.

[32] Afzal, M. A. F.; Haghighatlari, M.; Ganesh, S. P.; Cheng, C.; Hachmann, J. Accelerated discovery of high-refractive-index polyimides via first-principles molecular modeling, virtual high-throughput screening, and data mining. *The Journal of Physical Chemistry C* **2019**, *123*, 14610–14618.

[33] Afzal, M. A. F.; Cheng, C.; Hachmann, J. Combining first-principles and data modeling for the accurate prediction of the refractive index of organic polymers. *The Journal of Chemical Physics* **2018**, *148*, 241712.

[34] Afzal, M. A. F.; Hachmann, J. Benchmarking DFT approaches for the calculation of polarizability inputs for refractive index predictions in organic polymers. *Physical Chemistry Chemical Physics* **2019**, *21*, 4452–4460.

[35] Afzal, M. A. F. From virtual high-throughput screening and machine learning to the discovery and rational design of polymers for optical applications. Ph.D. thesis, University at Buffalo, 2018.

[36] Sonpal, A. Predicting Melting Points of Deep Eutectic Solvents. M.Sc. thesis, University at Buffalo, 2018.

[37] Tian, Y. Inheritance of molecular orbital energies from monomer building blocks to larger copolymers in organic semiconductors. M.Sc. thesis, University at Buffalo, 2016.

[38] Shih, C.-Y. Systematic trends in results from different density functional theory models. M.Sc. thesis, University at Buffalo, 2015.

[39] Hachmann, J.; Olivares-Amaya, R.; Atahan-Evrenk, S.; Amador-Bedolla, C.; Sánchez-Carrera, R. S.; Gold-Parker, A.; Vogt, L.; Brockway, A. M.; Aspuru-Guzik, A. The Harvard Clean Energy Project: Large-scale computational screening and design of organic photovoltaics on the world community grid. *The Journal of Physical Chemistry Letters* **2011**, *2*, 2241–2251.

[40] Olivares-Amaya, R.; Amador-Bedolla, C.; Hachmann, J.; Atahan-Evrenk, S.; Sánchez-Carrera, R. S.; Vogt, L.; Aspuru-Guzik, A. Accelerated computational discovery of high-performance materials for organic photovoltaics by means of cheminformatics. *Energy & Environmental Science* **2011**, *4*, 4849–4861.

[41] Amador-Bedolla, C.; Olivares-Amaya, R.; Hachmann, J.; Aspuru-Guzik, A. In *Informatics for Materials Science and Engineering Data-driven Discovery for Accelerated Experimentation and Application*; Rajan, K., Ed.; Butterworth-Heinemann: Oxford, 2013; Chapter 17, pp 423–442.

[42] Hachmann, J.; Olivares-Amaya, R.; Jinich, A.; Appleton, A. L.; Blood-Forsythe, M. A.; Seress, L. R.; Roman-Salgado, C.; Trepte, K.; Atahan-Evrenk, S.; Er, S.; Shrestha, S.; Mondal, R.; Sokolov, A.; Bao, Z.; Aspuru-Guzik, A. Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry - the Harvard Clean Energy Project. *Energy & Environmental Science* **2014**, *7*, 698–704.

[43] Lopez, S. A.; Pyzer-Knapp, E. O.; Simm, G. N.; Lutzow, T.; Li, K.; Seress, L. R.; Hachmann, J.; Aspuru-Guzik, A. The Harvard organic photovoltaic dataset. *Scientific Data* **2016**, *3*, 160086.

[44] Sanchez-Lengeling, B.; Aspuru-Guzik, A. Inverse molecular design using machine learning: Generative models for matter engineering. *Science* **2018**, *361*, 360–365.

[45] Duran-Frigola, M.; Fernndez-Torras, A.; Bertoni, M.; Aloy, P. Formatting biological big data for modern machine learning in drug discovery. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2018**, *0*, e1408.

[46] Xue, D.; Gong, Y.; Yang, Z.; Chuai, G.; Qu, S.; Shen, A.; Yu, J.; Liu, Q. Advances and challenges in deep generative models for de novo molecule generation. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2019**, *9*, e1395.

[47] Jørgensen, P. B.; Schmidt, M. N.; Winther, O. Deep Generative Models for Molecular Science. *Molecular Informatics* **2018**, *37*, 1700133.

[48] Haghighatlari, M. Making Machine Learning Work in Chemistry: Methodological Innovation, Software Development, and Application Studies. Ph.D. thesis, University at Buffalo, 2019.

[49] Krylov, A.; Windus, T. L.; Barnes, T.; Marin-Rimoldi, E.; Nash, J. A.; Pritchard, B.; Smith, D. G.; Altarawy, D.; Saxe, P.; Clementi, C.; Crawford, T. D.; Harrison, R. J.; Jha, S.; Pande, V. S.; Head-Gordon, T. Perspective: Computational chemistry software and its advancement as illustrated through three grand challenge cases for molecular science. *Journal of Chemical Physics* **2018**, *149*, 180901.

[50] Wilkins-Diehr, N.; Crawford, T. D. NSF's inaugural software institutes: The science gateways community institute and the molecular sciences software institute. *Computing in Science & Engineering* **2018**, *20*, 26–38.