Towards Autonomous Machine Learning in Chemistry *via* Evolutionary Algorithms

Gaurav Vishwakarma,[1,*] Mojtaba Haghighatlari,[1] and Johannes Hachmann[1,2,3,†]

[1] *Department of Chemical and Biological Engineering, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[2] *Computational and Data-Enabled Science and Engineering Graduate Program, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States*
[3] *New York State Center of Excellence in Materials Informatics, Buffalo, NY 14203, United States*

(Dated: September 7, 2019)

Machine learning has been emerging as a promising tool in the chemical and materials domain. In this paper, we introduce a framework to automatically perform rational model selection and hyperparameter optimization that are important concerns for the efficient and successful use of machine learning, but have so far largely remained unexplored by this community. The framework features four variations of genetic algorithm and is implemented in the *ChemML* program package. Its performance is benchmarked against popularly used algorithms and packages in the data science community and the results show that our implementation outperforms these methods both in terms of time and accuracy. The effectiveness of our implementation is further demonstrated *via* a scenario involving multi-objective optimization for model selection.

I. INTRODUCTION

Over the past few years, there has been a rising trend in the use of data-driven models in the field of chemistry for the discovery and design of new materials, advancing computational modeling techniques and gaining insight into chemical reactions [1–5]. Machine learning (ML), which is a data-mining process, offers tools that are useful for extracting complex, hidden correlations from chemical data and are far less demanding than the traditional time-intensive computational research (i.e., molecular modeling and simulation). However, ML techniques are yet to be recognized as mainstream tools in chemistry that makes them less preferred over other modeling choices.

One of the major challenges in the application of ML is the selection of a model to use in a given problem setting. Unlike for traditional computational research (i.e., molecular modeling and simulation), there are no decades of experience on which techniques do or do not work in a given situation, nor are there many physical insights to rationalize a particular methodological choice. Fig. 1 shows a typical ML workflow where it is seen that the performance of ML models is dependent not just on the feature representation of the data, but also on the hyperparameters that are selected for the learning algorithm. Hyperparameters for ML models can take on discrete, categorical or continuous values which renders hyperparameter selection to be essentially an optimization problem, however, not a traditional one, because in discrete or even categorical hyperparameter space, gradients (that could be used in gradient descent techniques) are not well defined.

The traditional method of choice for selecting hyperparameters for an ML model has been manual selection by intuition, and more often than not, on an *ad hoc* basis. Since there are no standard guidelines for such an approach, it differs across different applications and thereby cannot be utilized to conduct an automated hyperparameter search and/or over multiple ML models. These limitations highlight the need to conduct a systematic search of the valid hyperparameter space, and the simplest implementation perhaps is just a grid search over this entire space. However, it is readily seen that grid search suffers from the curse of dimensionality and the issues in its usage are further exacerbated for hyperparameters that take on continuous values since the valid hyperparameter space becomes infinite.

In 2012, Bergstra *et al.* [6] showed that random search was computationally more effective for hyperparameter optimization compared to grid search and resulted in better ML models. Although random search is an improvement over grid search, it still is inefficient as information gathered about the hyperparameter space during each of the trials is not used to guide the future trials. Hence, for complex, non-differentiable or high-dimensional space, one needs to look for a solution beyond these primitive optimization approaches.

Optimization of hyperparameters of various ML models has been the focus of a number of studies in data science. Particle swarm optimization [7], a sub-class of Swarm Intelligence methods, has been shown to derive ML models with good prediction accuracy [8–13]. Genetic Algorithm [14], a sub-class of evolutionary algorithms that is inspired from biological evolution, by itself and in combination with other methods, has widely been applied for optimization of hyperparameters [15, 16]. More recently, hyperparameter optimization using Gaussian processes and Bayesian optimization, that do not require the hyperparameter space to be continuously differentiable, have been widely studied and applied for this

* gvishwak@buffalo.edu
† hachmann@buffalo.edu

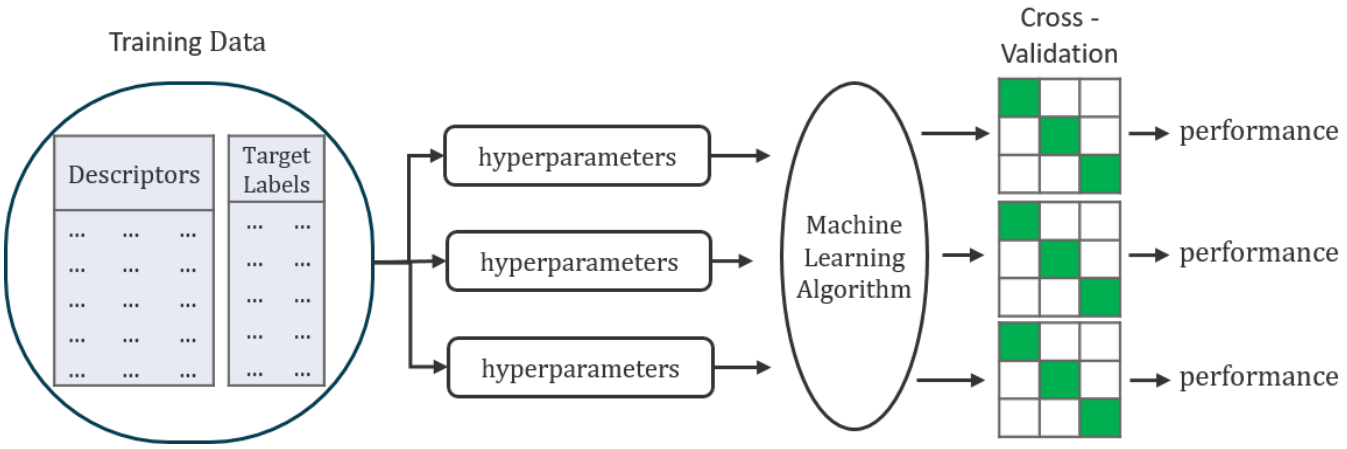

FIG. 1: A typical machine learning (ML) workflow where an ML model receives its hyperparameters and the training data as inputs and its performance is assessed based on its cross-validation score.

task and it has been shown that models with very high prediction accuracy can be obtained [17–21]. Complementary methods such as meta-learning have also been used on occasion to guide the initial guesses for these optimization methods thereby decreasing the computational overhead involved in such calculations [22, 23].

However, an extremely important criterion that has determined the popularity or usage of such advanced optimization methods in the past is the trade-off between the improvement in the model predictions *vs* the total time required for the optimization. This trade-off becomes more pronounced when the objective function being optimized is computationally expensive.

Given the number of studies conducted in data science involving different algorithms, one would naturally assume that these methods can be applied to problems pertaining to other domains as well. However, transferability of these approaches from data science to the chemical/materials domain in particular is not guaranteed, since the optimization is subject to the data used for training the model, and there exist some key differences between the data in the two domains. Variations in the volume of data sets, representation of data (descriptors), generalizability and veracity of the data are few of the major points of concern regarding chemical data sets as highlighted in the NSF report by Hachmann and co-workers [2]. Hence, this work investigates the optimization of hyperparameters for an ML model in the chemical and materials domain and presents one of the first proof-of-principle applications for a chemical data set.

In this work, we introduce a framework based on genetic algorithm (GA) that performs rational model selection and automatically optimizes the hyperparameters for a given ML model. Within this framework, we explore different strategies for GA and investigate their impact on the performance of the algorithm with a significant focus on the computation time. This paper further aims to assess and benchmark our genetic algorithm framework against two of the commonly used optimization algorithms within the data science community, to obtain fully optimized deep neural networks for predicting refractive indices of small organic molecules. In Sec. II, we formally present our implementation of a real-encoded genetic algorithm within the group's machine learning development platform, *ChemML* [24], along with the background of this work and other computational details. In Sec. III, we present and discuss results for hyperparameter optimization using our GA framework as well as two algorithms commonly used in data science. Our findings are summarized in Sec. IV.

II. BACKGROUND, METHODS, AND COMPUTATIONAL DETAILS

A. Method Development

We have developed an open-source machine learning and informatics program suite, *ChemML*, that makes machine learning tools more accessible to expert as well as non-expert users in the chemical and materials domain. *ChemML* has been designed to accomplish a two-fold agenda: to provide a testbed for the systematic and efficient evaluation of the performance of existing ML tools and techniques, and to serve as a development platform for any new methods that are developed within the group. The multi-objective GA module that this work focuses on has been developed as part of the *ChemML* package in order to automate the process of identification of the best ML approach for any given data set.

Inspired from biological evolution, a typical GA workflow begins with initializing a population of candidate solutions for the desired objective. The candidates are evaluated based on a certain metric and a subset of them is selected (*via* a selection operator) to breed new mem-

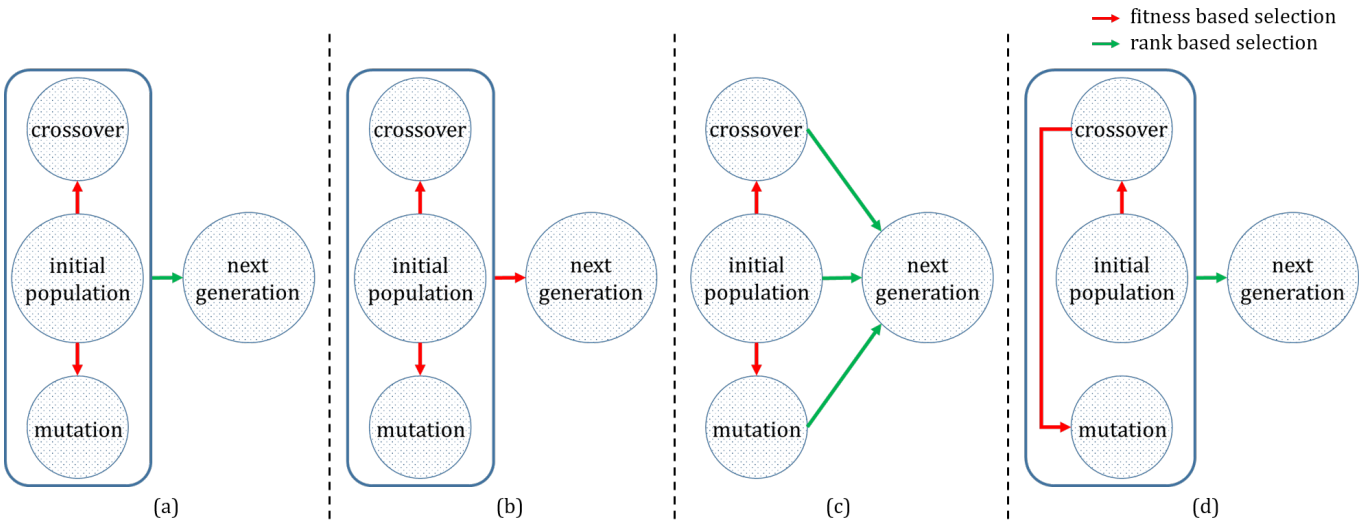

FIG. 2: Four different genetic algorithm selection schemes implemented in *ChemML*.

bers to form the next generation of candidate solutions. The breeding is performed *via* two other genetic operators, crossover and mutation. The crossover operation involves swapping a subset of the vector representation of one candidate solution (i.e., chromosome of the candidate) with the same positional subset of a second vector. The mutation operator involves randomly selecting elements of the vector representation and changing their values in a valid and predefined range. Once the breeding process is completed, individuals in the new generation are evaluated and the entire process is repeated again until a convergence criteria is satisfied.

The GA module in *ChemML* implements four different schemes for selecting individuals for future generations as shown in Fig. 2. These different schemes are intended to introduce various aspects of biological evolution apart from those derived from the Darwinian theory such as survival of low and average performing individuals, ensuring diversity in future generations and allowing for randomness in this process. The following elaborates on the selection process at every generation of GA for each of the four methods shown in the figure.

- *Method 1:* Members for crossover and mutation are selected from the initial population *via* fitness-based selection. The best n individuals are then selected for the next generation from the overall pool of crossover, mutation and initial population.
- *Method 2:* With a slight difference from the first method, this scheme selects individuals for the next generation using fitness-based selection, to allow for a higher representation of members with lower relative fitness values.
- *Method 3:* Individuals for crossover and mutation are selected in the same fashion as the first method. However, the selection process of members for the next generation follows a user-defined percentage of best individuals from each of the three pools, i.e., initial population, crossover and mutation, to ensure diversity in the population at each generation.
- *Method 4:* This method differs from the first method in the selection of individuals for mutation from the crossover pool, instead of the initial population, to introduce a random factor in the inherited genes.

In most optimization problems, the desired result involves optimization of more than one variable where each variable is optimized independent of the other variables. Thus, we also demonstrate the efficiency of our GA module in handling such cases of multi-objective optimization in Sec. III.

B. Data representation and model selection

In this work, we use a data set from our previous study on small organic molecules with a high refractive index (greater than 1.8) [25]. A subset of 20,000 molecules is selected randomly from this data that contains their refractive indices as the target property. The feature space for these molecules is generated using the Dragon 7 software where we only select the 1893 two-dimensional descriptors. The choice of feature representation, although simple, is computationally efficient and competitive with more demanding representations [25].

The data is initially split into a 90% training and 10% test set, where the hyperparameters are optimized for a five-fold cross-validation on the training set (i.e., validation step). The hold-out test set is eventually used to assess the performance of the model defined with the best set of hyperparameters and trained on the entire training set (i.e., test step). A standard deep neural network

(DNN) is used to fit the data, and its performance is assessed based on the prediction accuracy measured by the mean absolute error (MAE) of the trained models. We use Scikit-Learn[26] implementation of DNN as it is compatible with other essential methods in the library, and comparably efficient to run on CPUs.

We use the MAE as the objective function (i.e., the cost function) in all the hyperparameter selection strategies. In addition, the time taken to arrive at the final set of hyperparameters is monitored to arrive at the most cost-effective selection strategies. The computational time reported in this study for any method refers to the time taken to complete execution of the code running parallel on the same 24 core computation node. The regularization parameter, activation function, number of hidden layers and the number of neurons per hidden layer are the hyperparameters of interest for a DNN in our study [27]. The bounds for the hyperparameter space explored in this work are specified as:

- regularization parameter – (0.0001, 0.1)
- activation function – (logistic, identity, tanh, relu)
- number of hidden layers – (1, 3)
- number of neurons per hidden layer – (1, 250)

The 'iterations' in this work refer to the total number of times the objective function is evaluated or equivalently, the number of times the ML model is trained. The number of iterations that any algorithm is allowed to run for is set at a maximum value of 1000. For all calculations involving GA, the population size is held constant at 20, and the algorithm is allowed to run for a maximum of 50 generations. However, GA is allowed to stop at an earlier stage if the algorithm selects the same best individual for more than 10 generations consecutively, at which point we say that the algorithm has converged.

III. RESULTS AND DISCUSSION

We present the results for optimization of hyperparameters for two scenarios – optimization of a single objective and a multi-objective cost function.

A. Single Objective Optimization

Here we show the performance of random search, genetic algorithm (all four methods discussed in Sec. II) and the tree of parzen estimators (TPE) algorithm from the Hyperopt package [28] in finding the best set of hyperparameters for a neural network. The MAE for prediction of the refractive index of small organic molecules is the objective/cost function that is being minimized.

The results for this optimization are shown in Fig. 3, which plots the validation MAE for each of the algorithms along with the total computation time required for it. It is readily seen that even after a thousand iterations, random search isn't able to yield the best results (MAE) relative to GA and TPE. The least relative MAEs are obtained for the four methods in our GA implementation.

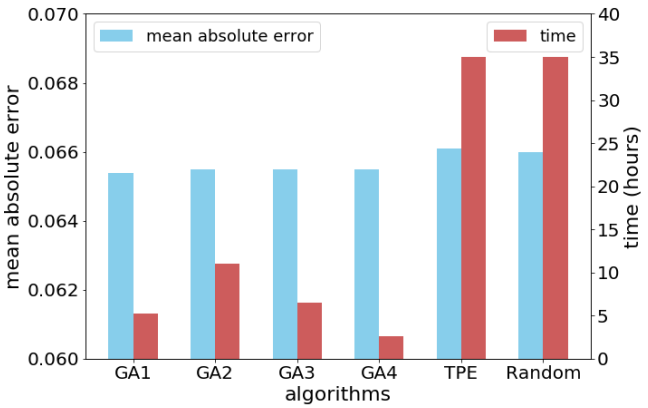

FIG. 3: Comparing the performance of genetic algorithm (GA), tree of parzen estimators (TPE) and random search for a single-objective optimization of hyperparameters of a deep neural network.

TABLE I: Evaluation metrics for single objective optimization of a neural network using genetic algorithm (GA), tree of parzen estimators (TPE) and random search.

	Validation MAE	Test MAE	Time (hours)
GA - Method 1	0.0654	0.0661	5.29
GA - Method 2	0.0655	0.0657	11.04
GA - Method 3	0.0655	0.0656	6.47
GA - Method 4	0.0655	0.0675	2.65
TPE	0.0661	0.0665	35.05
Random	0.0660	0.0659	35.01

Although the difference between the MAEs from the various algorithms is insignificant, the computation time that is required to obtain such performance is the critical source of difference. With respect to the time taken for this optimization, methods from GA show remarkable performance as compared to the other two algorithms. GA methods 1, 3 and 4 converged according to the criteria mentioned earlier. However, method 2 did not converge on account of its inherent design and thus the ML model was evaluated a thousand times. It should also be noted that the average time for all four GA methods is nearly one-fifth of the time taken by the other algorithms.

Table I summarizes the cross-validation and test MAE for the best combination of hyperparameters obtained from each of the algorithms tested in this work.

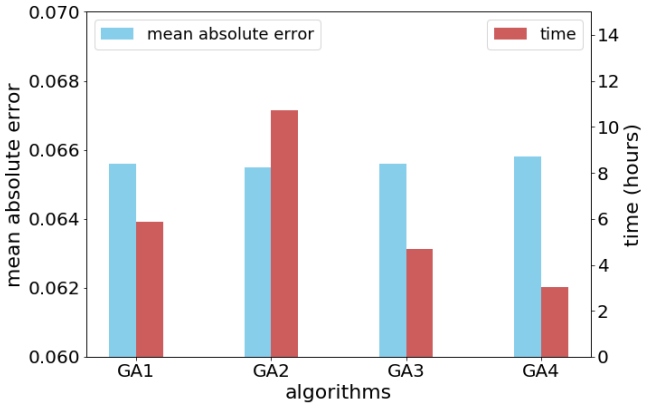

FIG. 4: Comparing the performance of the four strategies of genetic algorithm implemented in *ChemML* for optimization of hyperparameters *via* a multi-objective cost function.

TABLE II: Hidden layer sizes of a deep neural network architecture for single and multi-objective optimization using genetic algorithm (GA).

	Hidden Layer Sizes	
Algorithms	Single Objective	Multi - Objective
GA - Method 1	180, 20, 200	20, 100
GA - Method 2	120, 160, 80	100, 80, 100
GA - Method 3	160, 20, 180	120
GA - Method 4	60, 120, 100	40, 40

B. Multi-Objective Optimization

In the previous scenario, the neural network could get significantly large since the number of neurons in each of the three hidden layers could go up to a maximum of 250. Such a large neural network requires a considerable amount of computation time for training. Generally, it has been observed for neural networks that increasing the number of neurons at each hidden layer improves the model performance, however, at the expense of a proportional increase in the training time. In this scenario, in addition to minimizing the MAE, we seek to minimize the number of neurons in the neural network architecture. Shrinking of these models leads to a decrease in the number of tunable parameters. Thus, we can train low-variance models faster, and simultaneously make a trade-off for the MAE values to keep the performance as good as highly parameterized models.

Since random search and TPE algorithm do not support multi-objective optimization directly, we perform this calculation on just the four methods implemented in *ChemML* - GA. The results are summarized in Tables II and III which show the hidden layer sizes for single and multi-objective optimization, and the validation and test MAE and the time required for the computation respectively. It is noted that all the four GA methods are able to shrink the neural network model to a large extent.

TABLE III: Evaluation metrics for multi-objective optimization of a deep neural network using genetic algorithm (GA).

	Validation MAE	Test MAE	Time (hours)
GA - Method 1	0.0656	0.0658	5.87
GA - Method 2	0.0655	0.0655	10.73
GA - Method 3	0.0656	0.0666	4.7
GA - Method 4	0.0658	0.067	3.03

Fig. 4 plots the validation MAE and the computational time for the four GA methods, and it is seen that GA methods 1, 3 and 4 converged while method 2 did not converge. The plot also shows an interesting result where there is no significant change in the MAE of the four methods compared to the single objective scenario. This indicates that the model is quite insensitive to the number of neurons at each hidden layer. Such an insight into the neural network architecture (that can vary across different datasets) is hard to obtain *a priori* or with the traditional/primitive methods for selection of hyperparameters. This scenario brings to the foreground the significance of multi-objective optimization for ML models.

The results for the two scenarios presented in this work indicate that among the four methods for GA, the fastest convergence is achieved *via* method 4. However, a faster convergence does not necessarily guarantee the best possible model predictions since there is a good chance of the model getting stuck in a local minima. On the other hand, non-convergence of a method, as in the case of method 2, is also undesirable in most cases since time is an extremely important factor in such calculations. Methods 1 and 3 fall somewhere in between in terms of convergence but have nearly the same prediction accuracy as methods 2 and 4.

We have been employing our implementation of GA in a number of real-world application studies, not only for model selection, but also for feature selection and molecular library generation. These studies include discovery and design projects for new high-refractive-index polymers for optical applications [29–35], deep eutectic solvents for supercapacitors [36], and organic semiconductors for photovoltaics and other applications [37] (using data of the Harvard Clean Energy Project [38–42]).

IV. CONCLUSIONS

In conclusion, we have assessed and benchmarked our implementation of Genetic Algorithm within the *ChemML* program suite against two popularly used approaches in the data science community, random search method and the TPE algorithm from Hyperopt package. The four selection schemes implemented in Genetic Algo-

rithm outperform these approaches in terms of both time and accuracy. We have also shown that genetic algorithm is very accurate and efficient at optimizing multiple objectives simultaneously.

This work serves as a proof-of-principle study for a single machine learning model and chemical data set, and can, in-principle, be extended to different classes of chemical problems and types of data sets via a meta machine learning approach, i.e., we subsequently (machine) learn, which ML approach performs best for a given scenario. In this direction of future work, we compile the results of the hyperparameter optimization presented in this paper together with descriptors characterizing the corresponding data sets and chemical problems and mine this data to systematically derive insights and technical guidelines as mentioned before.

The Genetic Algorithm module presented in this paper is available to download from: https://github.com/hachmannlab/chemml.

SUPPLEMENTARY MATERIAL

Electronic supplementary material accompanies this paper and is available through the journal website. It provides details of the computational data displayed in the figures throughout this paper, i.e., , the final set of all the hyperparameters that were obtained from each of the optimization algorithms for both single-objective and multi-objective scenarios.

COMPETING FINANCIAL INTERESTS

The authors declare to have no competing financial interests.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation (NSF) CAREER program (grant No. OAC-1751161), and the New York State Center of Excellence in Materials Informatics (grant No. CMI-1148092-8-75163). Computing time on the high-performance computing clusters '*Vortex*', '*Alpha*', '*Beta*', and '*Gamma*' was provided by the UB Center for Computational Research (CCR). The work presented in this paper is part of GV's MS thesis [34]. 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 [43, 44].

[1] Mojtaba Haghighatlari and Johannes Hachmann. Advances of machine learning in molecular modeling and simulation. *Current Opinion in Chemical Engineering*, 23:51–57, 2019.

[2] Johannes Hachmann, Theresa L Windus, John A McLean, Vanessa Allwardt, Alexandra C Schrimpe-Rutledge, Mohammad Atif Faiz Afzal, and Mojtaba Haghighatlari. Framing the role of big data and modern data science in chemistry. Technical report, 2018. This NSF workshop report compiles the opinions of a group of active researchers in the field regarding the current challenges and future opportunities offered by data-driven approaches in the chemical domain.

[3] Kirstin Alberi, Marco Buongiorno Nardelli, Andriy Zakutayev, Lubos Mitas, Stefano Curtarolo, Anubhav Jain, Marco Fornari, Nicola Marzari, Ichiro Takeuchi, Martin L Green, et al. The 2019 materials by design roadmap. *Journal of Physics D: Applied Physics*, 52(1):013001, 2018.

[4] Matthias Rupp, O Anatole Von Lilienfeld, and Kieron Burke. Guest editorial: Special topic on data-enabled theoretical chemistry, 2018.

[5] Connor W Coley, William H Green, and Klavs F Jensen. Machine learning in computer-aided synthesis planning. *Accounts of chemical research*, 51(5):1281–1289, 2018.

[6] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[7] James Kennedy. *Particle swarm optimization*, pages 760–766. Springer, 2011.

[8] XC Guo, JH Yang, CG Wu, CY Wang, and YC Liang. A novel ls-svms hyper-parameter selection based on particle swarm optimization. *Neurocomputing*, 71(16-18):3211–3215, 2008.

[9] Hugo Jair Escalante, Manuel Montes, and Luis Enrique Sucar. Particle swarm model selection. *Journal of Machine Learning Research*, 10(Feb):405–440, 2009.

[10] Pablo Ribalta Lorenzo, Jakub Nalepa, Michal Kawulok, Luciano Sanchez Ramos, and José Ranilla Pastor. Particle swarm optimization for hyper-parameter selection in deep neural networks. In *Proceedings of the genetic and evolutionary computation conference*, pages 481–488. ACM, 2017.

[11] David JJ Toal, NW Bressloff, AJ Keane, and CME Holden. The development of a hybridized particle swarm for kriging hyperparameter tuning. *Engineering optimization*, 43(6):675–699, 2011.

[12] Bruno Feres De Souza, Andre CPLF De Carvalho, Rodrigo Calvo, and Renato Porfirio Ishii. Multiclass svm model selection using particle swarm optimization. In *2006 Sixth International Conference on Hybrid Intelligent Systems (HIS'06)*, pages 31–31. IEEE, 2006.

[13] Pablo Ribalta Lorenzo, Jakub Nalepa, Luciano Sanchez Ramos, and José Ranilla Pastor. Hyper-parameter selection in deep neural networks using parallel particle swarm optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1864–1871. ACM, 2017.

[14] J.H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Bi-*

ology, Control, and Artificial Intelligence. M.I.T.P., 1992. ISBN 9780262581110. URL https://books.google.com/books?id=5EgGaBkwvWcC.

[15] Mingyuan Zhao, Chong Fu, Luping Ji, Ke Tang, and Mingtian Zhou. Feature selection and parameter optimization for support vector machines: A new approach based on genetic algorithm with feature chromosomes. *Expert Systems with Applications*, 38(5):5197–5204, 2011.

[16] Steven R Young, Derek C Rose, Thomas P Karnowski, Seung-Hwan Lim, and Robert M Patton. Optimizing deep learning hyper-parameters through an evolutionary algorithm. In *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments*, page 4. ACM, 2015.

[17] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[18] Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John P Cunningham. Bayesian optimization with inequality constraints. In *ICML*, pages 937–945, 2014.

[19] Katharina Eggensperger, Matthias Feurer, Frank Hutter, James Bergstra, Jasper Snoek, Holger Hoos, and Kevin Leyton-Brown. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In *NIPS workshop on Bayesian Optimization in Theory and Practice*, volume 10, page 3, 2013.

[20] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pages 2171–2180, 2015.

[21] Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast bayesian optimization of machine learning hyperparameters on large datasets. *arXiv preprint arXiv:1605.07079*, 2016.

[22] Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. Initializing bayesian hyperparameter optimization via meta-learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[23] James Bergstra, Daniel Yamins, and David Daniel Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. 2013.

[24] Mojtaba Haghighatlari, Gaurav Vishwakarma, Doaa Altarawy, Ramachandran Subramanian, Bhargava Urala Kota, Aditya Sonpal, Srirangaraj Setlur, and Johannes Hachmann. Chemml: A machine learning and informatics program package for the analysis, mining, and modeling of chemical and materials data. *ChemRxiv*, page 8323271, 2019. doi:10.26434/chemrxiv.8323271.v1.

[25] Mojtaba Haghighatlari, Gaurav Vishwakarma, Mohammad Atif Faiz Afzal, and Johannes Hachmann. A Physics-Infused Deep Learning Model for the Prediction of Refractive Indices and Its Use for the Large-Scale Screening of Organic Compound Space. *ChemRxiv*, pages 1–9, 2019.

[26] F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[27] Alexios Koutsoukas, Keith J. Monaghan, Xiaoli Li, and Jun Huan. Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. *Journal of Cheminformatics*, 9(1):42, Jun 2017. ISSN 1758-2946. doi:10.1186/s13321-017-0226-y. URL https://doi.org/10.1186/s13321-017-0226-y.

[28] James Bergstra, Dan Yamins, and David D Cox. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in science conference*, pages 13–20. Citeseer, 2013.

[29] Mohammad Atif Faiz Afzal, C Cheng, and Johannes Hachmann. Combining first-principles and data modeling for the accurate prediction of the refractive index of organic polymers. *The Journal of Chemical Physics*, 148(24):241712, 2018. doi:10.1063/1.5007873.

[30] Mohammad Atif Faiz Afzal and Johannes Hachmann. Benchmarking DFT approaches for the calculation of polarizability inputs for refractive index predictions in organic polymers. *Physical Chemistry Chemical Physics*, 21:4452–4460, 2019. doi:10.1039/C8CP05492D.

[31] Mohammad Atif Faiz Afzal, Mojtaba Haghighatlari, Sai Prasad Ganesh, Chong Cheng, and Johannes Hachmann. 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*, 123:14610–14618, 2019. doi:10.1021/acs.jpcc.9b01147.

[32] Mohammad Atif Faiz Afzal, Aditya Sonpal, Mojtaba Haghighatlari, Andrew J Schultz, and Johannes Hachmann. A Deep Neural Network Model for Packing Density Predictions and its Application in the Study of 1.5 Million Organic Molecules. *ChemRxiv*, page 8217758.v1, 2019. doi:10.26434/chemrxiv.8217758.v1.

[33] Mohammad Atif Faiz Afzal. *From virtual high-throughput screening and machine learning to the discovery and rational design of polymers for optical applications*. PhD thesis, University at Buffalo, 2018.

[34] Gaurav Vishwakarma. Machine Learning Model Selection for Predicting Properties of High-Refractive-Index Polymers. Master's thesis, University at Buffalo, 2018.

[35] Mojtaba Haghighatlari. *Making Machine Learning Work in Chemistry: Methodological Innovation, Software Development, and Application Studies*. PhD thesis, University at Buffalo, 2019.

[36] Aditya Sonpal. Predicting Melting Points of Deep Eutectic Solvents. Master's thesis, University at Buffalo, 2018.

[37] Mojtaba Haghighatlari, Ching-Yen Shih, and Johannes Hachmann. Thinking globally, acting locally: On the issue of training set imbalance and the case for local machine learning models in chemistry. *ChemRxiv*, pages 1–10, 2019.

[38] Johannes Hachmann, Roberto Olivares-Amaya, Sule Atahan-Evrenk, Carlos Amador-Bedolla, Roel S Sánchez-Carrera, Aryeh Gold-Parker, Leslie Vogt, Anna M Brockway, and Alán Aspuru-Guzik. 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*, 2(17):2241–2251, 2011.

[39] Roberto Olivares-Amaya, Carlos Amador-Bedolla, Johannes Hachmann, Sule Atahan-Evrenk, Roel S Sánchez-Carrera, Leslie Vogt, and Alán Aspuru-Guzik. Accelerated computational discovery of high-performance materials for organic photovoltaics by means of cheminformatics. *Energy & Environmental Science*, 4(12):4849–4861, 2011.

[40] C. Amador-Bedolla, R. Olivares-Amaya, J. Hachmann, and A. Aspuru-Guzik. Organic Photovoltaics. In K. Rajan, editor, *Informatics for Materials Science and Engineering Data-driven Discovery for Accelerated Experimentation and Application*, chapter 17, pages 423–442. Butterworth-Heinemann, Oxford, 2013. ISBN 978-0123943996.

[41] Johannes Hachmann, Roberto Olivares-Amaya, Adrian Jinich, Anthony L Appleton, Martin A Blood-Forsythe, Laszlo R Seress, Carolina Roman-Salgado, Kai Trepte, Sule Atahan-Evrenk, Suleyman Er, Supriya Shrestha, Rajib Mondal, Anatoliy Sokolov, Zhenan Bao, and Alán Aspuru-Guzik. Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry - the Harvard Clean Energy Project. *Energy & Environmental Science*, 7(2):698–704, 2014.

[42] Steven A Lopez, Edward O Pyzer-Knapp, Gregor N Simm, Trevor Lutzow, Kewei Li, Laszlo R Seress, Johannes Hachmann, and Alán Aspuru-Guzik. The Harvard organic photovoltaic dataset. *Scientific Data*, 3: 160086, 2016.

[43] Anna Krylov, Theresa L. Windus, Taylor Barnes, Eliseo Marin-Rimoldi, Jessica A. Nash, Benjamin Pritchard, Daniel G.A. Smith, Doaa Altarawy, Paul Saxe, Cecilia Clementi, T. Daniel Crawford, Robert J. Harrison, Shantenu Jha, Vijay S. Pande, and Teresa Head-Gordon. Perspective: Computational chemistry software and its advancement as illustrated through three grand challenge cases for molecular science. *Journal of Chemical Physics*, 149(18):180901, 2018. ISSN 00219606.

[44] Nancy Wilkins-Diehr and T. Daniel Crawford. NSF's inaugural software institutes: The science gateways community institute and the molecular sciences software institute. *Computing in Science & Engineering*, 20(5):26–38, 2018. ISSN 1521-9615.