


A general optimization protocol for molecular property prediction using a deep learning network

Jen-Hao Chen and Yufeng Jane Tseng 

Corresponding author: Yufeng Jane Tseng, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.
Tel: +886-2-33664888#403; Fax: +886-2-23628167; E-mail: yjtseng@csie.ntu.edu.tw

Abstract

The key to generating the best deep learning model for predicting molecular property is to test and apply various optimization methods. While individual optimization methods from different past works outside the pharmaceutical domain each succeeded in improving the model performance, better improvement may be achieved when specific combinations of these methods and practices are applied. In this work, three high-performance optimization methods in the literature that have been shown to dramatically improve model performance from other fields are used and discussed, eventually resulting in a general procedure for generating optimized CNN models on different properties of molecules. The three techniques are the dynamic batch size strategy for different enumeration ratios of the SMILES representation of compounds, Bayesian optimization for selecting the hyperparameters of a model and feature learning using chemical features obtained by a feedforward neural network, which are concatenated with the learned molecular feature vector. A total of seven different molecular properties (water solubility, lipophilicity, hydration energy, electronic properties, blood–brain barrier permeability and inhibition) are used. We demonstrate how each of the three techniques can affect the model and how the best model can generally benefit from using Bayesian optimization combined with dynamic batch size tuning.

Key words: optimization; drug discovery; deep learning; CNN

Introduction

Deep learning has demonstrated its power in a wide range of application fields and has achieved breakthrough results, especially in computer vision [1, 2], video understanding [3, 4] and natural language processing [5, 6]. Such success has led to attempts to apply deep learning to other fields, including pharmaceutical domains. The continuing increase in chemical data and databases, especially those in the public domain, has also stimulated concurrent growth in deep learning methods in pharmaceutical research [7].

Although directly applying existing deep learning models can achieve fair performance, there are many optimization methods that can help to boost the performance and truly demonstrate

the true power of learning. These optimization methods are presented in different works for different scenarios, and each method was shown to improve the performance of the models. It is possible that they have the potential to improve the learning results for the data commonly seen in pharmaceutical research.

Among those optimization methods, some methods are established technologies whose effectiveness has been verified and has been widely used in many fields. For example, ensemble methods [8], which combine the predictions of multiple independently trained models, have been proven to generally produce more accurate predictions than nonensemble models. Coley *et al.* [9] used a graph-based convolutional neural network (CNN) for molecular embedding to predict aqueous solubility, octanol solubility, melting point and toxicity. Fivefold

Jen-Hao Chen is a PhD student in the Department of Computer Science and Information Engineering, National Taiwan University, and he is an engineer with Chunghwa Telecom Co., Ltd., Taipei, Taiwan. His research interests include machine learning and its applications in cheminformatics.

Yufeng Jane Tseng is a professor in the Department of Computer Science and Information Engineering, National Taiwan University. Her research interests include computational chemistry, toxicology, bioinformatics, cheminformatics and health informatics.

Submitted: 22 June 2021; Received (in revised form): 17 August 2021

© The Author(s) 2021. Published by Oxford University Press.

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited. For commercial re-use, please contact journals.permissions@oup.com

cross-validation was used to build five models, and the predictions were made by the consensus of all five models, weighted by the performance on the internal validation dataset. The ensemble model in this work achieved generally better prediction than individual models. Lusci et al. [10] used an ensemble of recursive neural networks associated with all possible vertex-centered acyclic orientations of the molecular graph on aqueous solubility and tested them on four benchmark datasets. Additionally, 20 models with a different number of hidden units and features were used as the ensemble.

Another common practice is using data augmentation [11] to increase the amount of data using information only from the given data as regularization to achieve better performances. Kimber et al. [12] demonstrated the synergistic effect between CNNs and the multiplicity of the simplified molecular-input line-entry system (SMILES) on a total of 18 datasets (9 datasets for the regression tasks and 9 datasets for the classification tasks). They increased the SMILES notation by 10 times and 25 times, allowing their model to learn more about the global structure of molecules. Their best results were obtained when augmentation on both the training and testing sets was implemented. Schwaller et al. [13] treated chemical reaction prediction as a machine translation problem between SMILES strings of reactants, reagents and their products. This work doubled the training data by generating a copy of every reaction in the training set, and they found that SMILES data augmentation leads to a significant increase in accuracy.

Hyperparameter optimization [14] is the other common optimization approach, which heavily influences the behavior of the learned model and is crucial for the success of neural network architectures. Wang et al. [15] proposed a multichannel substructure-graph gated recurrent unit architecture for molecular property prediction and used grid search for hyperparameter optimization. Duvenaud et al. [16] introduced a CNN that operates directly on graphs, generalizing standard molecular feature extraction methods based on circular fingerprints. Duvenaud et al. compared the performance of standard circular fingerprints on a variety of domains, including solubility, drug efficacy and organic photovoltaic efficiency. In their studies, a random search algorithm to optimize hyperparameters by 50 trials for each cross-validation fold was used.

Last, transfer learning [17] trains a neural network on a larger database and uses it to aid learning on a smaller dataset. Hu et al. [18] pretrained an expressive model at the level of individual nodes as well as entire graphs. Their pretraining strategy avoids negative transfer and improves generalization for molecular property prediction and protein function prediction, giving significantly better predictive performance than nonpretrained models. Goh et al. [19] used rule-based knowledge to train their model, which learns in a weakly supervised manner from large unlabeled chemical databases and is a transferable and generalizable deep neural network for chemical property prediction. This transfer method works effectively across network architectures and data modalities.

In this work, three high-performance optimization methods developed in the past decade that are relevant to the above established optimization methods were selected and analyzed. First, augmented data by the enumeration of SMILES representation of each compound gives the characteristic of redundancy, which helps to maintain the generalization performance as the small batch size while enjoying the benefit of a large batch size. We analyzed the effect of a dynamic batch size for optimized enumeration ratios of the SMILES representation of the compounds. This technique was proposed by Hoffer et al. [20], who performed

batch augmentation on image classification tasks with a larger batch composed of original samples in batches augmented with different transforms. They found that using larger augmented batches, computational resources can be better utilized without the cost of additional input and output (I/O). It is even possible to achieve a better generalization accuracy while incorporating existing learning rate schedules. We did not find related work in the chemistry field that applies this method.

Second, in hyperparameter optimization, engineers are often faced with myriad choices that are often complex and high dimensional, with interactions that are difficult to understand. This overwhelming number of design choices must be tuned manually, which is too vast for anyone to effectively navigate. In the past, the most widely used strategies for hyperparameter optimization were grid search and random selection. Bayesian optimization has emerged as a powerful solution for these varied design problems and promises greater automation to increase both product quality and human productivity [21]. In particular, Bayesian optimization prescribes a prior belief over the possible objective functions and then sequentially refines this model by Bayesian posterior updating as data are observed. The Bayesian posterior represents the updated beliefs on the likely objective function on which the model is optimized. Goh et al. [22] built a deep recurrent neural network (RNN) that automatically learns features from SMILES to predict chemical properties. They used a Bayesian optimizer to optimize the hyperparameters related to the neural network topology. Schwaller et al. [23] cast the reaction prediction task as a translation problem by introducing a template-free sequence-to-sequence model. They performed Bayesian hyperparameter optimization by a gradient-boosted-tree regression tree search, training 100 models for 30 epochs. The best model was further trained to 80 epochs to improve its final accuracy.

Finally, transfer learning is used. Transfer learning has been proposed to pass the knowledge from a related task that has been learned. For the same reasons, encoding relevant domain-specific information into the model can be used to bootstrap the training of a deep neural network and increase the overall model accuracy. It is likely that the additional representations helped, providing chemical information that is not discernible from raw representation, and some of that information appears to be related to the prediction target. Additionally, incorporating relevant domain-specific information does not need to learn the representations for these basic features but instead will be able to direct more of its learning capacity. We review the model leveraging multiple molecular representations as input. Paul et al. [24] utilized a vector input in the form of molecular fingerprints and a sequence input in the form of SMILES strings to develop neural networks for predicting chemical properties and found that it outperformed other deep learning models. Yang et al. [25] incorporated global molecular features into their model concatenated with the features learned from molecular graphs. They found that it is highly dataset dependent and can be further optimized by selecting different features more relevant to the task of interest. We tested these two kinds of hybrid representations.

While individual optimization methods from different past works outside the pharmaceutical domain each succeeded in improving the model performance, better improvement may be achieved with optimized combinations of these methods and practices. Compared to existing studies, there was no work that analyzed these optimization methods in detail. Most of the works applied these methods as an addition, while the complete practice procedures and the configurations of the optimization

methods are unknown. Unfortunately, without careful configurations and examinations, these optimization methods might even reduce the performance of the model.

For the dynamic batch size method, the Hoffer et al.'s approach simply augments the batch size by the ratio of the augmented data, while our experiment suggests that the smaller augmentation ratio for the batch size is better. For the hyperparameter optimization method, both Goh et al.'s and Schwaller et al.'s work did not provide practical protocol to follow. We did find that the work by Yang et al. [25] used the hyperparameter optimization method to determine the hyperparameters. In the work, 20 iterations of Bayesian optimization on 10 randomly seeded data were split to determine the best hyperparameters based on validation set performance. When we tried to repeat the protocol, we found the performance of the different seeded data splits is similar to each other. It was hard to determine whether the results benefit from the hyperparameters or due to different splitting of data. We therefore modified the approaches with same data splits and randomly seeded on the Bayesian optimization framework. For the hybrid representations, the Paul et al.'s work reported dramatic performance increase with the hybrid representations, while Yang et al. reported the effect on performance was dataset dependent. Since both groups used different additional representations with the raw representations, here we tested the representations, molecular fingerprints and features and their effects on performance.

We aim to thoroughly analyze the detailed configurations of the optimization methods and their combination effects, providing a general optimization protocol for others to directly apply. Our experiments show that the best model can benefit from using Bayesian optimization combined with dynamic batch size tuning.

The rest of this article is organized as follows: FRAMEWORK describes the overview of the procedure for employing the optimization methods to the model. METHODS explains the detailed configuration and implementation of the optimization methods we used in this work. EXPERIMENTS compares the performance of different combinations of the different optimization methods. DISCUSSION AND CONCLUSION discusses the findings and advantages of the optimization methods in this work.

Methods

ConvS2S model

This work is based on the fully convolutional sequence-to-sequence (ConvS2S) deep learning model illustrated in [26], and we refer readers to the original publication [27] for its architectural details.

Briefly, the model encodes the input molecule as an encoded representation, and an attention mechanism in the decoder network was used, which results in dot products between decoder context representations and encoded representations. The model is based entirely on CNN.

The overall processing flow of the model is shown in the first branch of Figure 1. The SMILES notation is transformed to numeric by a count-based dictionary, and then, it will be mapped to embedding vectors. The encoder network consists of repeated blocks, including a convolution network, gated linear units and a fully connected network used for mapping. An encoded representation for the input SMILES notation is generated at the end of the encoder network. The decoding network has a similar structure to that of the encoding network except that it has an attention mechanism, and the initial input is redundant

data. The decoded result in the decoder network layer first uses the dot product of the encoded representation to obtain the attention map, and the layer output is computed by a multiple encoded representation, which is augmented by adding position embedding to the attention map. The output of the final decoder network layer will go through some mapping to generate the final prediction. Note that the residual nets are employed in both the encoder and decoder networks to forward the result from the previous layer.

Dynamic batch size

We used the default batch size setting 4000 tokens as the base batch size. In [20], Hoffer et al. performed batch augmentation on image classification tasks that augmented the data size in each batch by the augmentation ratio. Specifically, for weight w_t at epoch t , learning rate η , batch size B , M instances of the same input sample by applying the transformation T_i , the learning rule can be denoted as follows:

$$w_{t+1} = w_t - \eta \frac{1}{M \cdot B} \sum_{i=1}^M \sum_{n \in B(k(t))} \nabla_w \ell(w_t, T_i(x_n), y_n)$$

where $\ell(w, x_n, y_n)$ is the loss function, $\{x_n, y_n\}_{n=1}^N$ is a dataset of N sample-target pairs and ∇_w is the gradient of weights. For simplicity, it is assumed that B divides N , $k(t)$ is sampled from $[N/B] \triangleq \{1, \dots, N/B\}$ and $B(k)$ is the set of samples in batch k . Note that the subscript $i \in [M]$ of T_i highlights the fact that the samples are different from one another.

Due to the large default batch size, we doubled the batch size by 10 times the SMILES enumeration ratio; i.e. the batch token sizes are 8000 and 16 000 for 10 \times and 100 \times enumerations, respectively. We slightly modified the rule denoted above:

$$w_{t+1} = w_t - \eta \frac{1}{r^{\log R} \cdot B} \sum_{i=1}^{r^{\log R}} \sum_{n \in B(k(t))} \nabla_w \ell(w_t, T_i(x_n), y_n)$$

where R is the SMILES enumeration ratio, r is the ratio for batch size augmenting and $k(t)$ here is sampled from the augmented dataset. The batch size at each step uses a larger size $r^{\log R} \cdot B$.

Bayesian optimization

We perform hyperparameter optimization by Bayesian optimization using the Hyperopt [28] Python library. Specifically, we follow the practice in [25], using 20 iterations of Bayesian optimization on 10 random seeds to determine the best hyperparameters and selecting hyperparameters based on the performance of the validation set.

The criterion to be optimized in Bayesian optimization is the expected improvement (EI), which can be expressed as:

$$EI_{y^*}(x) = \int_{-\infty}^{\infty} \max(y^* - y, 0) p_M(y|x) dy$$

where y^* is the best value found after observing \mathcal{H} : $y^* = \min\{f(x_i), 1 \leq i \leq n\}$ and p_M is the posterior model of \mathcal{H} , for which we used the tree-structured Parzen estimator [29] in this work. Unlike the Gaussian process-based approach that models $p(y|x)$ directly, the

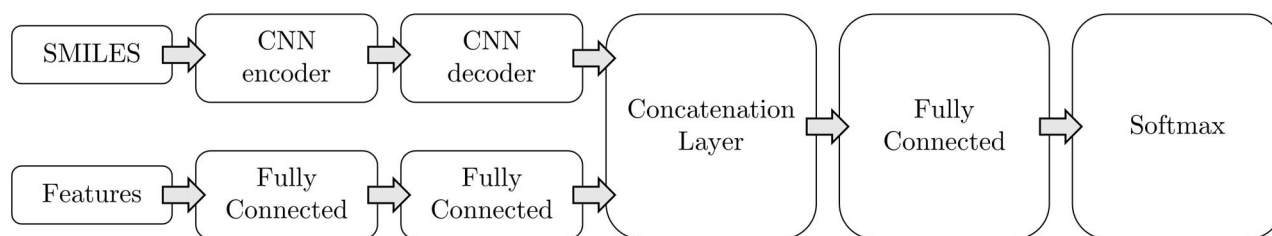


Figure 1. Detailed framework of the deep learning model. Two branches of representation learning are employed in the model. One branch uses raw input as the SMILES notation for the CNN encoder and CNN decoder to learn representations. The other branch uses computed molecular features with a fully connected network to learn the representations. These two branches of representation are concatenated, and another fully connected network is used to make the prediction.

Table 1. Bayesian optimization ranges for hyperparameter tuning

Hyperparameter	Configuration
Embedding size	8, 16, 32, 64, 128, 256
Convolution size	32, 64, 128, 256, 512
Number of encoder layers	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
Number of decoder layers	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Notes: Four hyperparameters, namely, the embedding size, convolution size, number of encoder layers and number of decoder layers, are automatically tuned. The batch size is also tuned in some configurations

tree-structured Parzen estimator model $p(x|y)$ is as follows:

$$p(x|y) = \begin{cases} \ell(x) & \text{if } y < y^* \\ g(x) & \text{if } y \geq y^* \end{cases}$$

where $\ell(x)$ is the density formed by using sets of hyperparameter values associated with the smallest loss function values and $g(x)$ is the density formed by the remaining hyperparameter values. By the above model, we can derive the EI:

$$EI_{y^*}(x) \propto \frac{\ell(x)}{g(x)}$$

which means that to maximize improvement, we would like points x with high probability under $\ell(x)$ and low probability under $g(x)$. The hyperparameters tuned for the model are shown in Table 1. We tuned four parameters, namely, the embedding size, convolution size, number of encoder layers and number of decoder layers. The embedding size is the vector size that embeds each token in the SMILES notation. The convolution size is the size of the vector for each input position in the CNN. The numbers of encoder and decoder layers are the layer counts in each network.

Hybrid representation

We introduced the hybrid representation to the model, which is shown in Figure 1. The first branch of representation is learned from the raw input of the SMILES notation by the CNN encoder and CNN decoder. The decoded representation can be utilized directly to predict the classification results. Another branch of representation is employed in the model, and it is concatenated with the decoded representation to perform the final prediction. This branch of representation can be learned from the molecular fingerprint or molecular features. More concretely, we modify the readout phase of the CNN model to apply fully connected

network f to the concatenation of the learned molecule feature vector h and the computed molecular features h_f : $y = f(\text{concatenate}(h, h_f))$.

We analyzed the hybrid representation of the maximum auto and cross-correlation (MACC) molecular fingerprint and 200 molecular features computed by RDKit. The 200 molecular features we used are the same as [25]. We transform all the features by quantile transformation to ensure that the differing magnitudes of different features do not cause certain features with large ranges dominating smaller ranged features, as well as putting all features into the same distribution.

Experimental procedure

The overall framework is shown in Figure 2. The dynamic batch technique is employed on the enumerated dataset. The enumerated dataset is input into the model with different batch sizes determined by its enumeration ratio in the training process. For hyperparameter optimization, we employed a Bayesian optimization technique to automatically tune the hyperparameters within a given range by the history of the optimization to make the search efficient. Another branch of the fully connected network with molecular features is introduced in the model. This branch of the model is concatenated with the learned molecule feature to make the final prediction.

The combinations of the three methods and different settings are tested, and the results of which are shown in Table 2. We first analyzed the enumeration strategy for the regression datasets and classification dataset denoted as 'Enu'. Then, since a larger batch size can reduce the training time, we investigated the better setting for the dynamic batch size, which was built upon the 'Enu' methods and is denoted as 'Enu_DB'. Afterwards, we performed Bayesian optimization to optimize the hyperparameters of the model. We tested several cases to identify the best practice and denoted it as Enu_DB_BO. Finally, we compared the model performance with that of two kinds of additional features, i.e. a simpler MACC molecular fingerprint (MA) and complex RDKit-computed molecular features (RD), which are denoted as Enu_DB_BO_MA and Enu_DB_BO_RD.

Experiments

We utilized optimization methods to analyze the impact on the performance of the model. The analysis was performed using PyTorch 1.8.1. We employed the Nesterov optimizer [30] for early stopping, and we followed the default configuration setting learning rate shrink factor with 0.1, learning rate 0.25 and minimum learning rate with 10^{-5} . All the reported models and configurations were trained 10 times and reported with 95% confidence intervals by t-tests.

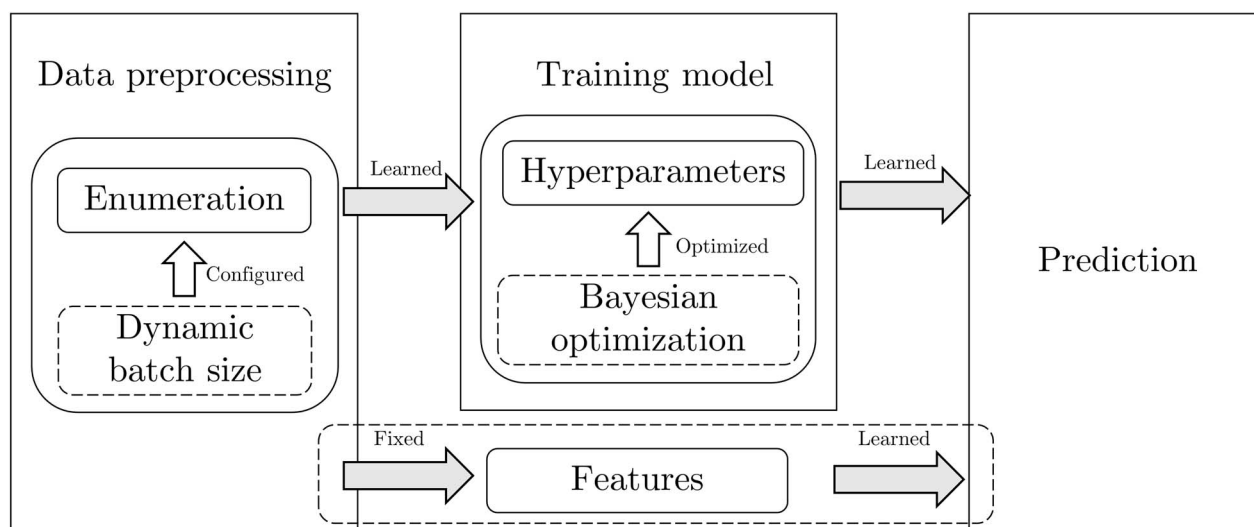


Figure 2. The optimization framework. The data preprocessing step prepares a batch of samples used at each step for gradient computation. The deep learning model, whose performance greatly depends on the hyperparameters, is trained on the data to make the prediction. Three optimization techniques were employed in the framework. The dynamic batch size strategy sets the batch size depending on the enumeration ratio. Bayesian optimization is used for hyperparameter tuning. Another branch of representation learning is performed by computed molecular features.

Table 2. The evaluated combinations of the optimization methods

Optimization methods	Notation	Configuration
Enumeration	Enu	Original dataset (1×) 10× 100× 1000×
Dynamic batch size	DB	Enu_DB(2x) Enu_DB(Fix2x)
Bayesian optimization	BO	Enu_DB_BO(1x) Enu_DB_BO(Enu)
Hybrid representation	MA (MACC fingerprint)	Enu_DB_BO_MA
	RD (RDKit features)	Enu_DB_BO(MA)_MA Enu_DB_BO_RD Enu_DB_BO(RD)_RD

Notes: The combinations of enumeration (Enu), dynamic batch size (DB), Bayesian optimization (BO), hybrid representation with the MACC fingerprint (MA) and hybrid representation with 200 RDKit features (RD) are tested. The best configuration of the optimization methods is denoted by its notation in further analysis. The condition each optimization method performs under is denoted in the parentheses

Dataset

We selected four datasets from all the categories (quantum mechanics, physical chemistry, biophysics and physiology) in MoleculeNet [31] to find the best optimization configuration. The four datasets are ESOL, QM7, BBBP and HIV, where ESOL and QM7 are regression tasks, and BBBP and HIV are classification tasks. Three more datasets are tested for the comparison. The three datasets are FreeSolv, Lipophilicity and BACE, where FreeSolv and Lipophilicity are regression tasks, and BACE is classification tasks. Since the DeepChem project [32] has integrated MoleculeNet as part of the package, we used the project to generate the datasets.

By testing the performance of the baseline model GraphConvModel [16] provided by DeepChem, we found that there is a large discrepancy between the performance of the baseline model in different versions of DeepChem, which is shown in Table 3. The baseline models are trained over 1000 epochs. The data generated by the two versions of DeepChem are the same in the

datasets, and the discrepancies result from the implementation of the split methods. We used the latest version 2.5 of DeepChem to generate the dataset and build the baseline model in our experiments.

After finding the best optimization configuration, we further introduce the state-of-the-art model GCN [33] and random forest [34] to compare with our model, and the three more datasets are tested in the comparison.

For quantum mechanics, the QM7 dataset is a subset of the GDB-13 database, providing electronic properties given stable conformational coordinates. For physical chemistry, ESOL is a small dataset consisting of water solubility data for 1128 compounds. For biophysics, the HIV dataset tests the ability of over 40 000 compounds to inhibit HIV replication. For physiology, the blood-brain barrier penetration (BBBP) dataset models barrier permeability. Three more datasets for comprehensive comparison are the FreeSolv, Lipophilicity and BACE datasets. FreeSolv is the Free Solvation Database that provides experimental and calculated hydration free energy of small molecules in water.

Table 3. The model performance of GraphConv on two DeepChem versions 2.3 and 2.5 with the same split methods and dataset

DeepChem version	ESOL (RMSE)	QM7 (MAE)	BBBP (ROC-AUC)	HIV (ROC-AUC)
2.5	1.178	191.4	0.687	0.732
2.3	0.483	85.0	0.895	0.747

Table 4. Descriptions of the public dataset from MoleculeNet

Category	Dataset	Type	Compounds	Rec – split	Rec – metric
Quantum mechanics	QM7	Regression	7165	Stratified	RMSE
Physical chemistry	ESOL	Regression	1128	Scaffold	MAE
Biophysics	HIV	Classification	41 127	Scaffold	ROC-AUC
Physiology	BBBP	Classification	2039	Scaffold	ROC-AUC
Physical chemistry	FreeSolv	Regression	642	Random	RMSE
Physical chemistry	Lipophilicity	Regression	4200	Scaffold	RMSE
Biophysics	BACE	Classification	1513	Scaffold	ROC-AUC

Notes: We tested all dataset categories by the recommended split from DeepChem and metric from MoleculeNet. Two datasets are selected for both regression type and classification type. Three more datasets, FreeSolv, Lipophilicity and BACE, are added for comparison with other related works

Lipophilicity is an important feature of drug molecules that affects both membrane permeability and solubility. The BACE dataset provides quantitative and qualitative binding results for a set of inhibitors of human β -secretase 1.

We split these datasets into training, validation and test subsets following a 80/10/10 ratio by using the split method recommended by DeepChem, and we evaluate them by the metrics recommended by MoleculeNet, which are shown in Table 4.

The classification datasets suffer from imbalanced data; for example, the HIV dataset is highly imbalanced, having only 3% active data. To address the negative effect of an imbalanced training dataset, sampling techniques are used widely in the context of machine learning models. One of the easiest and most effective methods is resampling (e.g. undersampling majority data and oversampling minority data). In [35], oversampling methods outperformed undersampling methods. Therefore, we utilized oversampling in this work.

For the enumerated dataset, we left out molecules that could not be parsed by RDKit [36]. The oversampling method was used to increase the amount of minority data in each dataset split so that the amount of minority data equaled the amount of majority data. To maintain the distribution of the minority class, we equally oversampled the compounds.

We comprehensively evaluated different combinations of the optimization methods by the ESOL, QM7, BBBP and HIV datasets and determined the best practice for the methods. Then, we used the best practice on all the datasets for our model to evaluate whether the best optimization methods applied on datasets can perform better than other related works.

Loss function and metric

We evaluate the regression performance for prediction using common metrics based on root mean squared error (RMSE), which for a given dataset of n samples is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2}$$

where Y_i is the predicted value and \hat{Y}_i is the measured value.

We evaluated the classification performance using common metrics based on the negative log-likelihood L :

$$L = -\log(y)$$

where y is the target class. The evaluation metric reported in our paper is the area under the receiver operating characteristic curve (ROC-AUC).

The best SMILES enumeration ratios are different in regression and classification datasets

In the field of chemical science, the common practice of data augmentation is SMILES enumeration, which randomizes atom ordering to enumerate the SMILES notations [37, 38].

We used the tool developed by Bjerrum [38], which generates different SMILES notations by converting the SMILES notations to a Molfile format and randomly changing the atom order. Then, the SMILES notation is generated using RDKit. The process restricts the SMILES notations by prioritizing the sidechains when traversing a ring and preventing SMILES substrings such as c1cc(c(cc1)) since the SMILES notation with unrestricted random atom order has worse performance for the model than restricted order [37].

From [26], with a higher enumeration ratio of the regression dataset, more improvement in the performance of the model can be obtained. We enumerated all the molecules in all the data split. For datasets in both the regression and classification types, we tried the original dataset and the enumerated datasets with three enumeration ratios, i.e. $10\times$, $100\times$ and $1000\times$; the results are shown in Table 5.

For regression tasks, we observed the same results where the best performances were obtained by a $1000\times$ enumeration ratio. However, for classification tasks, the best models were trained by the dataset with a $10\times$ enumeration ratio. We performed oversampling on the minority data in the classification tasks to address the imbalanced data issues, and the minority data were augmented. In the enumeration process, the minority data were augmented one more time, causing the model to focus on minority data and cannot be generalized well to test data. We used a $1000\times$ enumeration ratio for regression datasets and a $10\times$

Table 5. Comparisons of the model performance on the ESOL, QM7, BBBP and HIV datasets with enumeration ratios of 10 \times , 100 \times and 1000 \times of the dataset

Enumeration ratio	ESOL (RMSE)	QM7 (MAE)	BBBP(ROC-AUC)	HIV (ROC-AUC)
Original dataset	2.130 \pm 0.195	182.1 \pm 1.0	0.686 \pm 0.005	0.712 \pm 0.019
10 \times	1.203 \pm 0.167	175.3 \pm 0.6	0.690 \pm 0.008	0.731 \pm 0.015
100 \times	0.989 \pm 0.052	169.3 \pm 4.0	0.683 \pm 0.009	0.725 \pm 0.010
1000 \times	0.930 \pm 0.020	159.7 \pm 2.8	0.677 \pm 0.011	0.685*

Note: The results marked with a single asterisk are averaged over three runs

enumeration ratio for classification datasets in the remaining experiments and denoted this setting as ‘Enu’.

Since the training overhead is very large for the HIV dataset with a large enumeration ratio and the same trend can be observed in the 100 \times enumeration ratio case, we reported the average results over three iterations of the 1000 \times enumeration ratio case.

The evaluated combinations of the optimization methods

We comprehensively tested the combinations of dynamic batch size (DB), Bayesian optimization (BO), hybrid representation with MACC fingerprint (MA) and hybrid representation with 200 RDKit features (RD).

We first tested the effect of dynamic batch size with two configurations. One configuration doubles the batch size when the enumeration ratio is multiplied by 10 times, denoted as Enu_DB(2x). Another configuration always uses the doubled batch size for the enumerated dataset, denoted as Enu_DB(Fix2x). The best configuration is denoted as DB in further analysis.

Since the overhead of Bayesian optimization on a larger dataset is very large, we also tested whether the hyperparameters found by BO on a 1 \times dataset can be applied to an enumerated dataset. We denote the condition BO optimized on in the brackets. The conditions Enu_DB_BO(1x) and Enu_DB_BO(Enu) are tested in this case. The best configuration is denoted as BO in further analysis.

There are two kinds of hybrid representations that we tested in this work with the following conditions: Enu_DB_BO_MA and Enu_DB_BO_RD. To find the best hyperparameters in hybrid representation cases, Enu_DB_BO(MA)_MA and Enu_DB_BO(RD)_RD were also analyzed.

Results

Dynamic batch size has to be carefully configured

Table 6 demonstrates the performance of different configurations of the dynamic batch size. In the Enu_DB(2x) cases, all the model performances of the classification datasets BBBP and HIV were improved. This shows that increasing the batch size can benefit the model training in the enumeration dataset. However, the model performances of the regression datasets ESOL and QM7 were decreased. Since the enumeration ratio in regression datasets is 1000 \times and the batch sizes are enlarged eight times, we conjectured that the batch should be carefully increased instead of being simply proportional to the enumeration ratio.

We tested the Enu_DB(Fix2x) configuration that simply uses the doubled batch size for the dataset enumerated by a 1000 \times ratio. The results showed that the model performance for the

ESOL dataset increased in this configuration. The model performance of the QM7 dataset was still decreased, which may result from the default batch size already being too large for the QM7 dataset. In the classification cases, since the selected enumeration ratio is 10 \times , the dynamic batch size configuration Enu_DB(Fix2x) is the same as Enu_DB(2x). By the above observation, we selected DB(Fix2x) as the default DB configuration.

We also tried to identify how large the batch size would yield worse results for ESOL and QM7 datasets. The results are shown in Table 7. For the ESOL dataset, other than the batch size 8000 (Fix2x) and 32 000 (2 \times), we also tested other five batch size configurations: 4000, 6000, 10 000, 12 000 and 16 000. The best performance is still obtained with the configuration of the batch size 8000. For the QM7 dataset, we searched the best batch size setting on the original dataset from batch size 500 to 8000. Then, we compared the same batch size range on the enumeration dataset. The best configuration of the batch size on the original dataset is 500, and the best batch size on the enumeration dataset is 1000, which again demonstrate that the Enu_DB(Fix2x) obtained the best performance.

Bayesian optimization must be performed on the same enumerated dataset

Table 8 gives the performance of different configurations of the Bayesian optimization. In the Enu_DB_BO(Enu) configuration cases, the model performances for the QM7, BBBP and HIV datasets were improved. The model performance for the ESOL dataset was slightly decreased, indicating that the default hyperparameters are already good enough for the dataset. The default hyperparameters of the model were originally tailored for the ESOL dataset [26].

Since the overhead is large when conducting Bayesian optimization on the enumerated dataset, we tried to generalize the hyperparameters from the original dataset to the enumerated dataset. In the Enu_DB_BO(1x) cases, unfortunately, all the models cannot achieve better performance than the Enu_DB_BO(Enu) case. The results showed that the hyperparameters of the model obtained from the original dataset cannot be generalized to the enumerated dataset, and the optimized hyperparameters are specific to the dataset configuration.

The metric for the classification dataset in this work is the ROC-AUC, which is not differentiable, and Bayesian optimization may be advantageous for this. We tried Bayesian optimization for the ROC-AUC on the original dataset 1x_BO(1x_AUC), and the results show that performing Bayesian optimization on the ROC-AUC for the classification dataset seems to help in selecting hyperparameters. Based on the observation, we further tested Bayesian optimization for the ROC-AUC on the Enu_DB_BO(Enu_AUC) enumerated dataset. For the BBBP dataset case, the hyperparameters found happen to be the same as Enu_DB_BO(Enu). For the HIV dataset case, the hyperparameters found cannot perform better than the Enu_DB_BO(Enu) case.

Table 6. Comparison of the model performances on the ESOL, QM7, BBBP and HIV datasets with two dynamic batch size strategies Enu_DB(2x) and Enu_DB(Fix2x)

Configuration		ESOL (RMSE)	QM7 (MAE)	BBBP(ROC-AUC)	HIV (ROC-AUC)
Enu		0.930 ± 0.020	159.7 ± 2.8	0.690 ± 0.008	0.731 ± 0.015
Enu_DB(2x)	Batch size	32 000	32 000	8000	8000
	Result	0.989 ± 0.057	161.0 ± 2.9	0.691 ± 0.005	0.745 ± 0.019
Enu_DB(Fix2x)	Batch size	8000	8000	8000	8000
	Result	0.908 ± 0.017	163.2 ± 3.1	0.691 ± 0.005	0.745 ± 0.019

Table 7. Performance comparison of the different batch size on (a) ESOL and (b) QM7 dataset

(a) ESOL (RMSE)							
Batch size	4000	6000	8000	10 000	12 000	16 000	32 000
1000×	0.930 ± 0.020	0.915 ± 0.026	0.908 ± 0.017	0.928 ± 0.029	0.945 ± 0.03	0.999 ± 0.061	0.989 ± 0.057
(b) QM7 (MAE)							
Batch size	500	1000	2000	4000	8000		
Original dataset	180.6 ± 3.0	187.1 ± 6.9	185.5 ± 4.2	182.1 ± 1.0	275.4 ± 90.7		
1000×	166.8 ± 8.9	155.8 ± 1.3	158.0 ± 3.0	159.7 ± 2.8	163.2 ± 3.1		

Notes: The best performance in each dataset is highlighted with boldface. The batch sizes from 4000 to 32 000 are tested on the ESOL dataset, while the batch sizes from 500 to 8000 are tested on the QM7 dataset

Table 8. Comparison of the model performance on the ESOL, QM7, BBBP and HIV datasets with four Bayesian optimization strategies: Enu_DB_BO(1x), Enu_DB_BO(Enu), 1x_BO(1x_AUC) and Enu_DB_BO(Enu_AUC)

Configuration	ESOL (RMSE)	QM7 (MAE)	BBBP (ROC-AUC)	HIV (ROC-AUC)
Enu_DB	0.908 ± 0.017	163.2 ± 3.1	0.691 ± 0.005	0.745 ± 0.019
Enu_DB_BO(1x)	0.937 ± 0.042	172.1 ± 2.8	0.691 ± 0.004	0.734 ± 0.016
Enu_DB_BO(Enu)	0.917 ± 0.011	154.2 ± 2.3	0.694 ± 0.004	0.749 ± 0.012
1x_BO(1x_AUC)			0.694 ± 0.013	0.733 ± 0.015
Enu_DB_BO(Enu_AUC)			0.694 ± 0.004	0.742 ± 0.011

This could be caused by the loss cross entropy we used in the training process, which is different in nature from the ROC-AUC. Some other differentiable loss functions are closer to the ROC-AUC [39]; we left these for future work. The best practice we found for Bayesian optimization is Enu_DB_BO(Enu), and we used this configuration as the default Bayesian optimization setting BO.

The effects of hybrid representation are dataset dependent

Table 9 demonstrates the performances of different configurations of the hybrid representation. Both the Enu_DB_BO_MA and Enu_DB_BO_RD configurations increased the model performance in the BBBP dataset while decreasing the model performance in the ESOL, QM7 and HIV datasets.

We further performed Bayesian optimization on all datasets to find the best hyperparameters in the hybrid representation setting. The results of the Enu_DB_BO(MA)_MA and Enu_DB_BO(RD)_RD configurations showed that the model performance of the BBBP and HIV datasets benefited from the setting while the ESOL and QM7 datasets did not vary by much.

Other configuration effects on model performance seem to be dataset dependent. The Enu_DB_BO(RD)_RD configuration achieves better performance on the HIV dataset and the Enu_DB_BO(MA)_MA on the BBBP dataset, while they both have worse performance in another configuration. Observed

from the performed experiment, Enu_DB_BO is the most stable configuration to optimize the model, which we used as the final evaluation configuration.

Discussion and Conclusion

Enumeration cannot be replaced by an optimization technique

Table 10 demonstrates the minimal configuration required to apply the optimization techniques. For example, to apply the dynamic batch size technique, the dataset has to be enumerated at least 10 times. Compared with the simplest model not applying optimization techniques, the performances of the model applying optimization techniques are better in some cases.

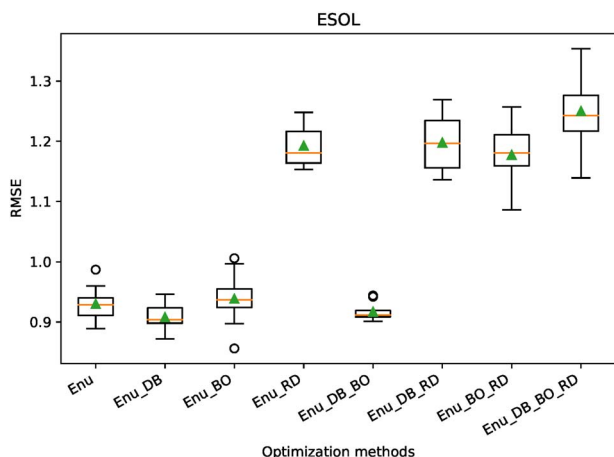
In the dynamic batch size case, the model that applied the technique on the BBBP and HIV datasets outperformed the baseline model. In the Bayesian optimization case, the model that applied the technique on the ESOL, QM7 and HIV datasets outperformed the simple model. In the hybrid representation case, the model that applied the technique on the ESOL and HIV datasets outperformed the simple model. However, none of the three techniques can replace the benefit of the enumeration, and a good strategy to integrate these optimization techniques into the enumerated dataset is crucial to increase the model performance.

Table 9. Comparison of the model performances on the ESOL, QM7, BBBP and HIV datasets with four hybrid representation strategies: Enu_DB_BO_MA, Enu_DB_BO_RD, Enu_DB_BO(MA)_MA and Enu_DB_BO(RD)_RD

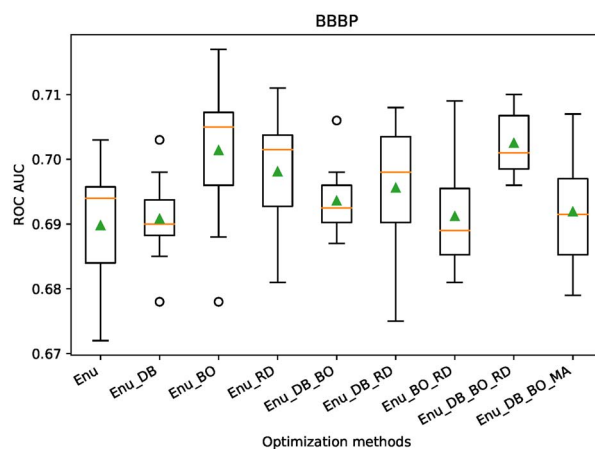
Configuration	ESOL (RMSE)	QM7 (MAE)	BBBP(ROC-AUC)	HIV (ROC-AUC)
Enu_DB_BO(Enu)	0.917 \pm 0.011	154.2 \pm 2.3	0.694 \pm 0.004	0.749 \pm 0.012
Enu_DB_BO_MA	1.222 \pm 0.166	312.8 \pm 4.0	0.695 \pm 0.006	0.727 \pm 0.012
Enu_DB_BO_RD	1.243 \pm 0.041	303.0 \pm 8.6	0.695 \pm 0.007	0.724 \pm 0.011
Enu_DB_BO(MA)_MA	1.119 \pm 0.049	308.6 \pm 5.5	0.703 \pm 0.004	0.746 \pm 0.013
Enu_DB_BO(RD)_RD	1.250 \pm 0.045	344.7 \pm 34.1	0.692 \pm 0.006	0.749 \pm 0.013

Table 10. Comparison of the model performances on the ESOL, QM7, BBBP and HIV datasets with four optimization techniques, i.e. 10x_DB(2x), 1x_BO(1x), 1x_MA and 1x_RD, to examine if enumeration can be replaced

Configuration	ESOL (RMSE)	QM7 (MAE)	BBBP(ROC-AUC)	HIV (ROC-AUC)
Original dataset	2.130 \pm 0.195	182.1 \pm 1.0	0.686 \pm 0.005	0.712 \pm 0.019
10x	1.203 \pm 0.167	175.3 \pm 0.6	0.690 \pm 0.008	0.731 \pm 0.015
10x_DB(2x)	1.764 \pm 0.126	175.3 \pm 1.3	0.691 \pm 0.005	0.745 \pm 0.019
1x_BO(1x)	1.902 \pm 0.248	180.1 \pm 2.0	0.675 \pm 0.01	0.725 \pm 0.011
1x_MA	2.061 \pm 0.065	285.5 \pm 54.3	0.680 \pm 0.006	0.726 \pm 0.008
1x_RD	2.069 \pm 0.029	293.8 \pm 27.4	0.669 \pm 0.017	0.714 \pm 0.010

**Figure 3.** Boxplot depicting the performance of the original model and the models employing the optimization methods with 10 random seeds on the ESOL dataset. The x-axis shows the optimization techniques, and the y-axis represents the RMSE. The green triangles are the average of the performance for the configurations.

To test the effectiveness of these three modules, we performed the ablation experiments on ESOL dataset and BBBP dataset, and the results are shown in Figures 3 and 4. In Figure 3, the hybrid representations always reduced the performance on the ESOL dataset. The Bayesian optimization may reduce performance in some cases (Enu and Enu_BO), while benefit the model performance in other cases. The dynamic batch cannot benefit the model in some cases (Enu_BO_RD and Enu_DB_BO_RD), but it also benefits the model performances in other cases. The best performance is obtained by the Enu_DB_BO configuration. In Figure 4, all the optimization methods benefit the model performance (Enu, Enu_DB, Enu_BO and Enu_RD). Among all the combination cases from two optimization methods, only the Enu_DB_BO configuration can be beneficial (Enu_DB and Enu_DB_BO). Although the configurations with all the optimization methods achieved the best performance, this is not true when using molecular fingerprint as the hybrid representation. Also, the hybrid representation with molecular features on the

**Figure 4.** Boxplot depicting the performance of the original model and the models employing the optimization methods with 10 random seeds on the BBBP dataset. The x-axis shows the optimization techniques, and the y-axis represents the ROC AUC. The green triangles are the average of the performance for the configurations.

BBBP dataset did not achieve better performance in the combination cases from two optimization methods; therefore, the best configuration should still be the Enu_DB_BO.

Optimization techniques must be carefully integrated with enumeration

In light of dynamic batch size, although the redundancy of the enumerated dataset can be employed to enlarge the batch size, excessively enlarging the batch size leads to poorer performance, such as the model on the ESOL and QM7 datasets. To mitigate the negative effect caused by a large batch size, we limit the batch augmentation with the same ratio regardless of the enumeration ratio of the dataset, and the results of the Enu_DB(Fix2x) configurations demonstrated better performance than the Enu_DB(2x) cases.

Since the size of each dataset is different, the base batch size should also be dynamically adjusted, and Bayesian optimization

Table 11. Comparison of the molecular data statuses in the ESOL, QM7, BBBP and HIV datasets

Symbol properties	ESOL	QM7	BBBP	HIV
Average	22.58	33.38	47.09	52.02
Max	98	49	233	405
Min	2	2	4	3

Note: The average, max and min token sizes of the molecules in the datasets are reported

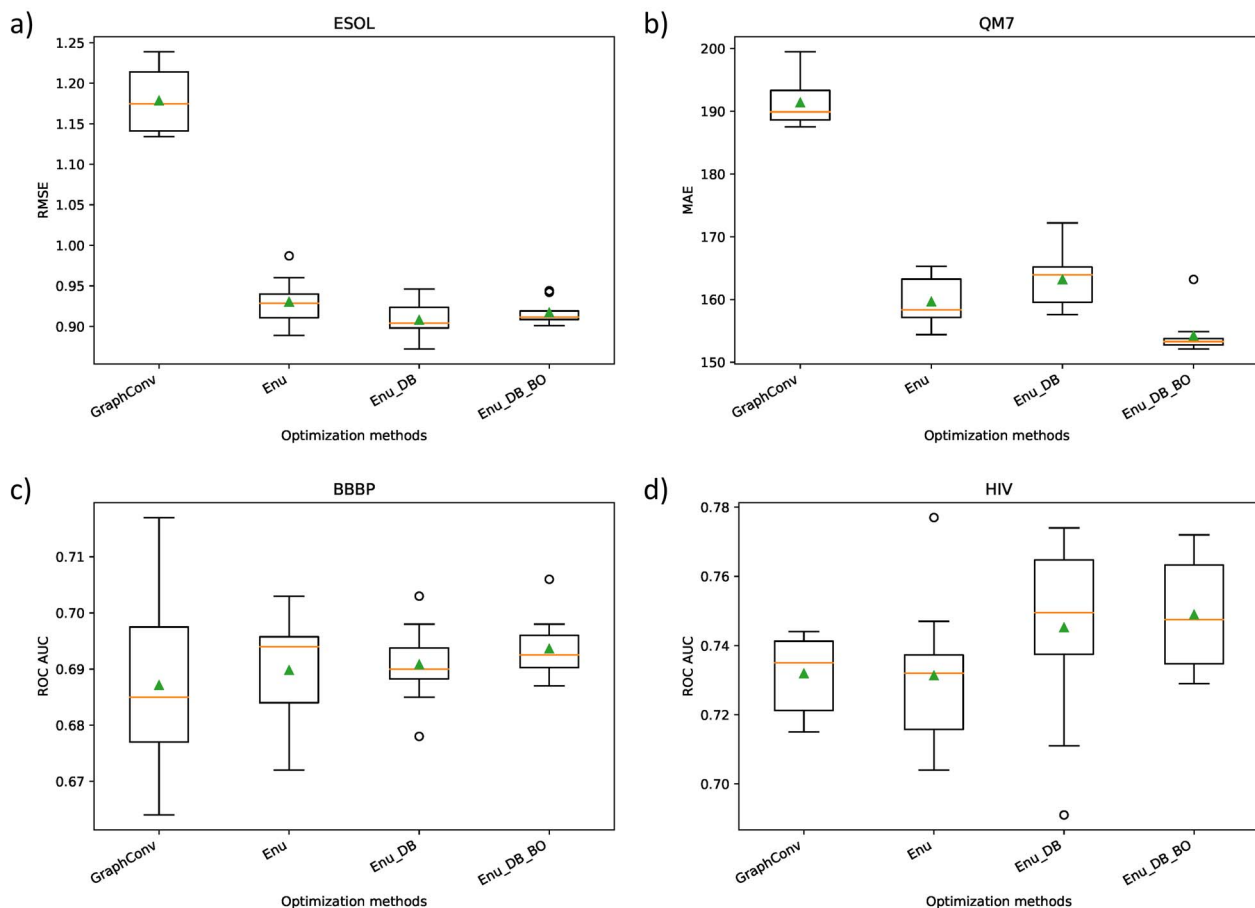


Figure 5. Boxplot of model performances on Enu, Enu_DB and Enu_DB_BO configurations and baseline model GraphConv. The green triangles are the average of the performance for the configurations.

can be a direction to find the right batch size of the base batch size for the original dataset and the augmentation batch size for the enumerated dataset. We leave this extension to future work to find the relation between model performance and the optimized batch size.

An advantage of the dynamic batch size strategy is reducing the training time, allowing the model to have a larger budget to try different designs and hyperparameter configurations. Therefore, we employed this technique first, and other optimization techniques can build on top of it.

For Bayesian optimization, we tried to generalize the optimized hyperparameters of the model on the original dataset to the model on the enumerated dataset but in vain. The results showed that the optimized hyperparameters have to be found on the same configurations, including the enumeration ratio of the dataset. The overhead for finding new hyperparameters on the

enumerated dataset is unavoidable, and the benefits are worth the cost.

Molecular features benefit the model in certain situations

We examined the effect of the hybrid representation on the enumerated dataset. The hybrid representation of the molecular features leverages both the advantages of automatic feature extraction and classical molecular representation, which have been reported to improve the performance of the model in some studies [24, 25].

We analyzed two kinds of features, a simpler MACC molecular fingerprint and complex RDKit-computed molecular features, and a hybrid representation that incorporates each of the classical molecular representations cannot perform better

Table 12. Performance comparison of the related works GraphConv, GCN and random forest on both the (a) regression and (b) classification type datasets**(a) Regression datasets**

Model	ESOL (RMSE)	QM7 (MAE)	FreeSolv (RMSE)	Lipophilicity(RMSE)
GraphConv	1.178 ± 0.028	191.4 ± 2.9	1.249 ± 0.053	$0.777 \pm 0.013^{**}$
GCN	$1.122 \pm 0.105^{**}$	196.6 ± 7.3	$0.701 \pm 0.114^*$	0.907 ± 0.035
Random forest	1.696 ± 0.014	$178.6 \pm 0.3^{**}$	1.379 ± 0.018	0.963 ± 0.004
This work	$0.917 \pm 0.011^*$	$154.2 \pm 2.3^*$	$1.126 \pm 0.047^{**}$	$0.740 \pm 0.003^*$

(b) Classification datasets

Model	BBBP (ROC-AUC)	HIV (ROC-AUC)	BACE (ROC-AUC)
GraphConv	0.687 ± 0.011	0.732 ± 0.008	$0.803 \pm 0.012^{**}$
GCN	0.689 ± 0.013	0.715 ± 0.014	0.759 ± 0.026
Random forest	$0.706 \pm 0.008^*$	$0.774 \pm 0.010^*$	$0.860 \pm 0.006^*$
This work	$0.694 \pm 0.004^{**}$	$0.749 \pm 0.012^{**}$	0.786 ± 0.017

Note: The best performance of all the models in each dataset is marked with a single asterisk, and the second-best performance is marked with a double asterisk

than the simpler setup on ESOL and QM7 datasets. Interestingly, the effect of adding features to the BBBP and HIV datasets is positive in the Enu_DB_BO(MA)_MA and Enu_DB_BO(RD)_RD configurations.

We examined the molecular data in each dataset, which are shown in Table 11. The BBBP and HIV datasets have larger average token lengths of 47 and 52 of all the molecules, and the maximum lengths of the molecules in the dataset are 233 and 405, which are much larger than the ESOL and QM7 datasets with 98 and 49.

There are limitations when utilizing only raw representations to develop models. One limitation is that the models are unable to learn to identify and extract all the features of a molecule due to little training data, and they are susceptible to overfitting to artifacts in the data. We have already addressed this limitation by SMILES notation enumeration.

The other limitation is that the encoding processes can capture only local information and result in a molecular representation that is fundamentally local rather than global in nature, making them struggle to predict properties that depend heavily on the global features. We hypothesize that this is the main reason hybrid representation can achieve better performance on the BBBP and HIV datasets.

Additionally, if relevant domain-specific information were to be provided to the neural network, it would not need to learn the representations for these basic features but instead would be able to direct more of its learning capacity to develop more sophisticated representations and improve its accuracy for predicting complex chemical properties. Although this point of view is not clear from our experimental results, the results of the Enu_DB_BO(RD)_RD configurations on the BBBP dataset perform better than the best optimization configuration Enu_DB_BO, demonstrating the possibility.

The model employing the best optimization methods achieved the best performance compared to related work

We compared the baseline GraphConv model to the model with the Enu, Enu_DB and Enu_DB_BO configurations shown in Figure 5. The ESOL, QM7 and HIV datasets were significantly better than the GraphConv model evaluated by t-test. The GraphConv model on the BBBP dataset has larger variance and has less mean performance than our model. The results also showed marked improvement with optimization techniques.

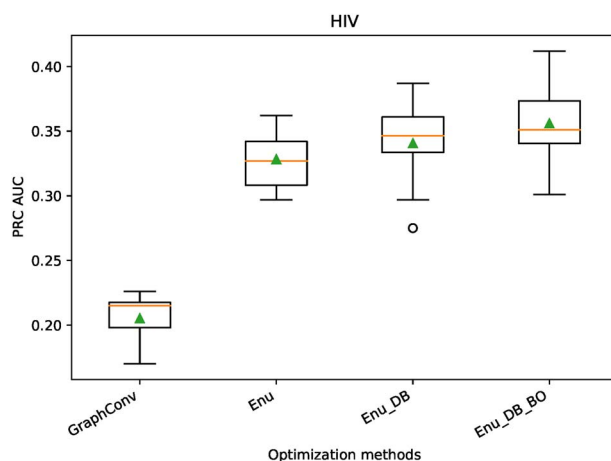


Figure 6. Boxplot of model performances on Enu, Enu_DB and Enu_DB_BO configurations and baseline model GraphConv on the HIV dataset. The x-axis shows the optimization techniques, and the y-axis represents the PRC AUC.

The Enu_DB_BO configuration not only improved the model performance but also largely reduced the variance in the prediction. With the dynamic batch size techniques, the training time is also doubled compared to the Enu configurations.

The Enu_DB and Enu_DB_BO models with the HIV dataset have larger variance than the GraphConv model. The variance may increase when the model tries to reduce the generalization gap, since the model has to struggle with the imbalanced dataset. We also observed that the precision-recall plot is more informative than the ROC plot when evaluating on imbalanced datasets [40]. Therefore, we also reported the results of the PRC-AUC metric, which is shown in Figure 6. The variance of the proposed models in PRC-AUC metrics is largely reduced, and the performance is much better than the GraphConv model, which strengthened our confidence on the proposed optimization protocol.

We compared the model employing the best optimization methods to related work such as GraphConv, GCN [33] and random forest [34] on the same dataset and split, and we found that we achieved the best performance. The results are reported in Table 12, where the best performance of all the models in each dataset is marked with a single asterisk and the second-best performance is marked with a double asterisk. In addition to the ESOL, QM7, BBBP and HIV datasets, we also compared the model

performances on three other datasets, FreeSolv, Lipophilicity and BACE.

Among all the models, random forest performed best on the classification datasets; however, it cannot perform well on the regression datasets. The GCN model performed well on some regression datasets such as ESOL and FreeSolv. The GraphConv model performs moderately on both the regression and classification datasets.

Compared to the models from related work, our model achieves the top 2 in six out of seven datasets, demonstrating that our model can succeed on a wide range of datasets. After applying the best optimized method, our model performance is dramatically improved and can achieve the lowest RMSE on the regression tasks and the comparable ROC-AUC on classification tasks. We observed that other models can perform well in one type of task but not in the other. Our model can perform well in both types of tasks.

Key Points

- Different enumeration ratios must be applied to classification tasks (10×) and regression tasks (1000×).
- Three techniques, namely, the dynamic batch size strategy for different enumeration ratios of the SMILES representation of the compounds, Bayesian optimization to select the hyperparameters of the model and hybrid representation, could benefit the deep learning model.
- The best model could benefit from a simple optimized combination Enu_DB_BO rather than a combination of all techniques at the same time.
- The proposed model can succeed on a wide range of datasets and can perform well in both regression and classification types of tasks.

Associated content

The full datasets and code are available at https://github.com/titanda/Learn-it-all/tree/ready_classification_feature.

Acknowledgments

Resources from the Laboratory of Computational Molecular Design and Metabolomics and the Department of Computer Science and Information Engineering of National Taiwan University were used when performing these studies.

Funding

Neurobiology & Cognitive Science Center at NTU (NTU-CC-110L890803, NTU-110L8809); TFDA (grant number MOHW110-FDA-D-114-000611); and the Ministry of Science and Technology (Taiwan) (MOST 109-2823-8-002-010-CV, 109-2320-B-002-040-).

References

- Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*. Association for Computing Machinery, 2012, 1097–105.
- Wang F, Jiang M, Qian C, et al. Residual attention network for image classification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Institute of Electrical and Electronics Engineers, 2017, p. 3156–64.
- Abu-El-Haija S, Kothari N, Lee J, et al. Youtube-8m: a large-scale video classification benchmark. 2016. arXiv:Computer Vision and Pattern Recognition.
- Carreira J, Zisserman A. Quo vadis, action recognition? a new model and the kinetics dataset. In: *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Institute of Electrical and Electronics Engineers, 2017, p. 6299–308.
- Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. 2014. arXiv:Computation and Language.
- Devlin J, Chang M-W, Lee K, et al. Bert: pre-training of deep bidirectional transformers for language understanding. 2018. arXiv:Computation and Language.
- Chen H, Engkvist O, Wang Y, et al. The rise of deep learning in drug discovery. *Drug Discov Today* 2018;23:1241–50.
- Dietterich TG. Ensemble methods in machine learning. In: *International Workshop on Multiple Classifier Systems*. Springer, Berlin, Heidelberg, 2000, 1–15.
- Coley CW, Barzilay R, Green WH, et al. Convolutional embedding of attributed molecular graphs for physical property prediction. *J Chem Inf Model* 2017;57:1757–72.
- Lusci A, Pollastri G, Baldi P. Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules. *J Chem Inf Model* 2013; 53:1563–75.
- Perez L, Wang J. The effectiveness of data augmentation in image classification using deep learning. 2017. arXiv:Computer Vision and Pattern Recognition.
- Kimber TB, Engelke S, Tetko IV, et al. Synergy effect between convolutional neural networks and themultiplicity of SMILES for improvement of molecular prediction. 2018. arXiv:Machine Learning.
- Schwaller P, Laino T, Gaudin T, et al. Molecular transformer for chemical reaction prediction and uncertainty estimation. 2018. arXiv:Chemical Physics.
- Bergstra J, Bengio Y. Random search for hyper-parameter optimization. *J Mach Learn Res* 2012;13:281–305.
- Wang S, Li Z, Zhang S, et al. Molecular property prediction based on a multichannel substructure graph. *IEEE Access* 2020;8:18601–14.
- Duvenaud DK, Maclaurin D, Iparraguirre J, et al. Convolutional networks on graphs for learning molecular fingerprints. In: *Advances in Neural Information Processing Systems*. MIT Press, 2015, 2224–32.
- Oquab M, Bottou L, Laptev I, et al. Learning and transferring mid-level image representations using convolutional neural networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Institute of Electrical and Electronics Engineers, 2014, p. 1717–24.
- Hu W, Liu B, Gomes J, et al. Strategies for pre-training graph neural networks. 2019. arXiv:Machine Learning.
- Goh GB, Siegel C, Vishnu A, et al. Using rule-based labels for weak supervised learning: a ChemNet for transferable chemical property prediction. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery, 2018, p. 302–10.
- Hoffer E, Ben-Nun T, Hubara I, et al. Augment your batch: better training with larger batches. 2019. arXiv:Machine Learning.

21. Shahriari B, Swersky K, Wang Z, et al. Taking the human out of the loop: a review of Bayesian optimization. *Proc IEEE* 2015;**104**:148–75.
22. Goh GB, Hodas NO, Siegel C, et al. Smiles2vec: an interpretable general-purpose deep neural network for predicting chemical properties. 2017. arXiv:Machine Learning.
23. Schwaller P, Gaudin T, Lanyi D, et al. “Found in translation”: predicting outcomes of complex organic chemistry reactions using neural sequence-to-sequence models. *Chem Sci* 2018;**9**:6091–8.
24. Paul A, Jha D, Al-Bahrani R, et al. Chemixnet: mixed dnn architectures for predicting chemical properties using multiple molecular representations. 2018. arXiv:Machine Learning.
25. Yang K, Swanson K, Jin W, et al. Analyzing learned molecular representations for property prediction. *J Chem Inf Model* 2019;**59**:3370–88.
26. Chen J-H, Tseng YJ. Different molecular enumeration influences in deep learning: an example using aqueous solubility. *Brief Bioinform* 2021;**22**(3):bbaa092.
27. Gehring J, Auli M, Grangier D, et al. Convolutional sequence to sequence learning. 2017. arXiv:Sound.
28. Bergstra J, Yamins D, Cox D. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In: *International Conference on Machine Learning*. Association for Computing Machinery, 2013, p. 115–23.
29. Bergstra JS, Bardenet R, Bengio Y, et al. Algorithms for hyperparameter optimization. In: *Advances in Neural Information Processing Systems*. MIT Press, 2011, p. 2546–54.
30. Sutskever I, Martens J, Dahl G, et al. On the importance of initialization and momentum in deep learning. In: *International Conference on Machine Learning*. Association for Computing Machinery, 2013, p. 1139–47.
31. Wu Z, Ramsundar B, Feinberg EN, et al. MoleculeNet: a benchmark for molecular machine learning. *Chem Sci* 2018;**9**:513–30.
32. Ramsundar B, Eastman P, Walters P, et al. *Deep Learning for the Life Sciences: Applying Deep Learning to Genomics, Microscopy, Drug Discovery, and More*. O'Reilly Media, Inc., Sebastopol, California, 2019.
33. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. 2016. arXiv:Social and Information Networks.
34. Breiman L. Random forests. *Mach Learn* 2001;**45**:5–32.
35. Lee YO, Kim YJ. The effect of resampling on data-imbalanced conditions for prediction towards nuclear receptor profiling using deep learning. *Mol Inform* 2020;**39**:1900131.
36. Landrum G, et al. *Rdkit: Open-source cheminformatics*, 2006.
37. Arús-Pous J, Johansson SV, Prykhodko O, et al. Randomized SMILES strings improve the quality of molecular generative models. *J Chem* 2019;**11**:1–13.
38. Bjerrum EJ. SMILES enumeration as data augmentation for neural network modeling of molecules. 2017. arXiv:Machine Learning.
39. Yan L, Dodier RH, Mozer M, et al. Optimizing classifier performance via an approximation to the Wilcoxon-Mann-Whitney statistic. In: *Proceedings of the 20th International Conference on Machine Learning (icml-03)*. Association for Computing Machinery, 2003, p. 848–55.
40. Saito T, Rehmsmeier M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLoS One* 2015;**10**: e0118432.