Message Passing Neural Networks for Molecular Property Prediction

by

Kyle Swanson

Submitted to the
Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Computer Science and Engineering

at the

Massachusetts Institute of Technology

June 2019

Author: _____

Department of Electrical Engineering and Computer Science
May 24, 2019

Certified by: _____

Regina Barzilay, Delta Electronics Professor, Thesis Supervisor
May 24, 2019

Accepted by: _____

Katrina LaCurts, Chair, Master of Engineering Thesis Committee

Message Passing Neural Networks for Molecular Property Prediction

by

Kyle Swanson

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2019, in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Computer Science and Engineering

Abstract

Developing new drugs relies heavily on understanding the various molecular properties of potential drug candidates. While experimental assays performed in the lab are the best source of information about molecular properties, these assays are slow and expensive. For this reason, there has been great interest in the potential of machine learning models to predict molecular properties without the need for experimental assays. However, recent literature has not yet clearly determined which machine learning models are optimal for molecular property prediction. In this thesis, I apply the Direct Message Passing Neural Network (D-MPNN) from [47, 48] to 19 publicly available property prediction datasets, and I demonstrate that it consistently outperforms prior machine learning models. Additionally, I introduce several optimizations to the D-MPNN which further enhance its performance and lead to new state-of-the-art results.

Thesis Supervisor: Regina Barzilay
Title: Delta Electronics Professor

Acknowledgments

Thank you to the editors of the Journal for Chemical Information and Modeling for allowing the reproduction of portions of [48] in this thesis. Thank you as well to all of my co-authors on [48] for collaborating on this research project and making it possible. I would like to thank the industry partners of the Machine Learning for Pharmaceutical Discovery and Synthesis consortium for funding this research and for providing useful discussions regarding how to develop and use property prediction models in a real-world setting. I would also like to thank the members of the computer science and chemical engineering groups in the Machine Learning for Pharmaceutical Discovery and Synthesis consortium for their helpful ideas and feedback throughout the research process. Thank you to Zhenqin Wu for his aid in recreating the original data splits of [46] and to Andreas Mayr for helpful suggestions regarding adapting the model from [32] to new classification and regression datasets. Additionally, I would like to thank Lior Hirschfeld for his incredible work building a web-based user interface for the property prediction model [1]. Finally, a huge thank you to Kevin Yang, who was an extraordinary collaborator and friend and helped develop all of the work presented in this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Molecular property prediction is one of the oldest cheminformatics tasks and is crucial to rapidly developing new drugs. Accurate property prediction models allow chemists to rapidly filter through vast libraries of molecules, thereby helping them more quickly identify viable drug candidates. In recent years, machine learning has begun to play an increasingly prominent role in molecular property prediction as it provides a fast, cheap, and accurate framework for learning a property prediction model from prior experimental results.

Machine learning architectures for property prediction have been developed either to operate on fixed molecular descriptors or fingerprints or to learn a task-specific representation by operating directly on the molecular graph [16, 46, 22, 20, 28, 24, 13, 7, 9, 40, 5]. However, while both approaches have shown promise, it is still unclear which of the two approaches is superior.

In this thesis, I aim to determine which machine learning model paradigm is better for property prediction. To this end, I develop and extend the Directed Message Passing Neural Network (D-MPNN) from [47, 48], which belongs to the latter category of models operating on molecular graphs. I introduce three optimizations to the D-MPNN: two standard machine learning optimizations—hyperparameter optimization and ensembling—along with an optimization that augments the D-MPNN with additional molecule-level information. I then extensively test the D-MPNN and its optimizations on 19 publicly available datasets from [32] and [46]. I compare

the performance of the D-MPNN to prior work from [32, 46] and to a number of descriptor/fingeprint-based baselines that I develop.

The results indicate the clear superiority of the D-MPNN to all prior models. The D-MPNN demonstrates strong out-of-the-box performance, and with the additional optimizations that I introduce, it establishes new state-of-the-art performance on a number of datasets. This indicates that machine learning models that operate on molecular graphs are in fact superior to those which utilize fixed descriptors or fingerprints, and so future work should prioritize further developing these machine learning approaches.

Chapter 2

Background

Many current approaches to molecular property prediction rely on standard machine learning models such as random forests [6] and support vector machines [10] which are applied to hand-crafted molecular descriptors, such as the Dragon descriptors [31], or to molecular fingerprints, like Morgan (ECFP) fingerprints [37]. Improvements in property prediction accuracy using these models has primary come through the development of better molecular descriptors [44, 8, 15, 33, 31, 32]. Other approaches have explored the use of 3D atomic coordinates to augment the information provided to these models [46, 39, 25, 17, 18].

Another line of research has instead investigated how to build stronger models. Some work has focused on building better models for descriptors or fingerprints [32, 27], while other work has explored the use of SMILES [45] strings [32] or the molecular graph [16, 46, 22, 20, 28, 24, 13, 9, 7, 40, 5] as input. The latter approach relies on models known as graph convolutional or message passing neural networks. These models operate directly on featurizations of the atoms and bonds in the molecular graph, giving them flexibility to build a molecular representation more relevant to the property or properties of interest [20, 46].

The model used in this thesis, the Directed Message Passing Neural Network (D-MPNN) [47, 48], builds o of the message passing paradigm. However, rather than propagating information along atoms as in [20, 46], the D-MPNN propagates information along directed bonds, which gives the model greater control over the flow

of information, thus resulting in better molecular representations and more accurate property predictions. In this thesis, I further enhance the D-MPNN using both standard machine learning techniques and by incorporating computable molecule-level information, and I extensively test this model against prior work.

Chapter 3

Methods

The core model used throughout this work is the Directed Message Passing Neural Network (D-MPNN) [48], a recent model I helped develop that builds off of the message passing neural network (MPNN) framework established by [20]. In this chapter, I'll first describe MPNNs and the D-MPNN in particular, and then I'll describe several optimizations I developed to improve the performance of the D-MPNN.

3.1 Message Passing Neural Networks

An MPNN is a type of neural network model which is specifically designed to operate on graphs. The input to an MPNN is an undirected graph G with node features x_v and bond features e_{vw}. In chemistry, the graph is a molecule with atoms as nodes and bonds as edges.

With this featurization as input, the MPNN then operates in two phases. First is a *message passing phase*, which propagates information across the graph in order to build a neural representation of the whole graph. Second is the *readout phase*, where the neural representation of the graph is then used to make predictions. In this work, the goal is to use a molecular graph to predict a property or properties of interest.

The message passing phase consists of T steps of information propagation. Hidden states h_v^t and messages m_v^t are associated with each vertex v, and on each time step t, those states are updated using a message function M_t and a vertex updated function

U_t according to:

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw})$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

where $N(v)$ is the set of neighbors of v in graph G, and h_v^0 is some function of the initial atom features x_v.

In the readout phase, a readout function R is applied to the set of final hidden states h_v^T to make the prediction as follows:

$$\hat{y} = R(\{h_v^T | v \in G\}).$$

The readout function typically works by first building a single representation h of the whole graph by summing the final hidden states:

$$h = \sum_{v \in G} h_v^T.$$

Then, a feed-forward neural network f is applied to h to output

$$\hat{y} = f(h).$$

The MPNN is trained end-to-end, with the gradient of the loss backpropagated through both the readout and message passing phases. The MPNN can be trained in either a regression or a classification setting by modifying the loss function appropriately.

3.2 Directed MPNN

The primary difference between the D-MPNN and regular MPNNs is in the nature of the messages being passed through the molecule during the message passing phase. While the general MPNN framework assumes messages m_v^t are centered on atoms, the

D-MPNN instead centers messages on bonds. Specifically, the D-MPNN maintains two representations for the message centered on the bond between atoms v and w: one from atom v to atom w (m^t_{vw}) and one from atom w to atom v (m^t_{wv}). Consequently, rather than aggregating information from neighboring atoms, the D-MPNN aggregates information from neighboring bonds. Each bond's message is updated based on all incoming bond messages m^t_{kv} where $k \in \{N(v) \setminus w\}$. Due to this structure, with messages centered on bonds and a distinction between the two directions of bond messages, the D-MPNN has greater control over the flow of information across the molecule and can therefore build more informative molecular representations. More details of the D-MPNN architecture are provided in [47, 48].

3.3 D-MPNN Optimizations

Although the D-MPNN is a powerful model on its own, there is still room to increase the model's predictive capacity. To this end, I explored three strategies for optimizing the D-MPNN's performance on property prediction tasks. The first—adding computed molecule-level features—leverages the power of cheminformatics packages such as RDKit [26] to add information that a D-MPNN might not be able to learn e□ciently during training. The other two strategies—hyperparameter optimization and ensembling—are standard machine learning techniques known to improve the performance of a wide variety of models.

3.3.1 Computed Molecule-Level Features

Although an MPNN should ideally be able to extract *any* information about a molecule that might be relevant to predicting a given property, two limitations may prevent this in practice. First, many property prediction datasets are very small, i.e., on the order of only hundreds or thousands of molecules. With so little data, MPNNs are unable to learn to identify and extract all features of a molecule that might be relevant to property prediction, and they are susceptible to overfitting to artifacts in the data. Second, most MPNNs use fewer message passing steps than the diameter of

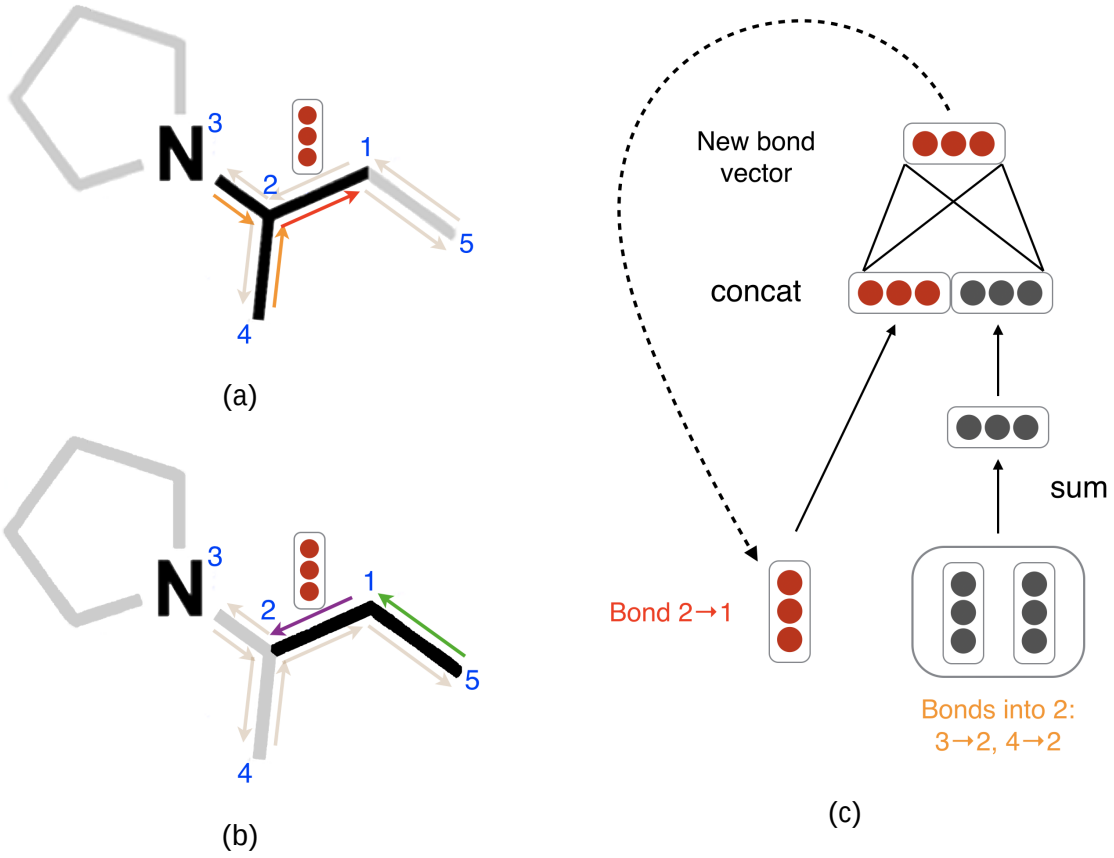


Figure 3-1: Illustration of bond-level message passing in a D-MPNN. (a): Messages from the orange directed bonds are used to inform the update to the hidden state of the red directed bond. By contrast, in a traditional MPNN, messages are passed from atoms to atoms (for example, from atoms 1, 3, and 4 to atom 2) rather than from bonds to bonds. (b): Similarly, a message from the green bond informs the update to the hidden state of the purple directed bond. (c): Illustration of the update function to the hidden representation of the red directed bond from diagram (a).

the molecular graph, i.e. $T < \texttt{diam}(G)$, meaning atoms that are a distance of greater than T bonds apart will never receive messages about each other. This results in a molecular representation that is fundamentally local rather than global in nature, meaning the MPNN may struggle to predict properties that depend heavily on global features.

In order to counter these limitations, I developed a variant of the D-MPNN that incorporates 200 global molecular features that can be computed rapidly *in silico* using RDKit. The neural network architecture requires that the features are appropriately scaled to prevent features with large ranges dominating smaller ranged features, as well as preventing issues where features in the training set are not drawn from the same sample distribution as features in the testing set. To prevent these issues, a large sample of molecules was used to fit cumulative density functions (CDFs) to all features. CDFs were used as opposed to simpler scaling algorithms mainly because CDFs have the useful property that each value has the same meaning: the percentage of the population observed below the raw feature value. Min-max scaling can be easily biased with outliers, and Z-score scaling assumes a normal distribution, which is most often not the case for chemical features, especially if they are based on counts.

The CDFs were fit to a sample of 100k compounds from the Novartis internal catalog[1] using the distributions available in the scikit-learn package [35], a sample of which can be seen in Figure 3-2. One could do a similar normalization using publicly available databases such as ZINC [21] and PubChem [23]. scikit-learn was used primarily due to the simplicity of fitting and the final application. However, more complicated techniques could be used in the future to fit to empirical CDFs, such as finding the best fit general logistic function, which has been shown to be successful for other biological datasets [38]. No review was taken to remove odd distributions. For example, azides are hazardous and rarely used outside of a few specific reactions, as reflected in the fr_azide distribution in Figure 3-2. As such, since the sample data was primarily used for chemical screening against biological targets, the distribution

[1]Thank you to Brian Kelley from Novartis for suggesting the normalization strategy and for providing the data and fitting the CDFs.

used here may not accurately reflect the distribution of reagents used for chemical synthesis. For the full list of calculated features, please refer to Appendix G.

To incorporate these features, I modified the readout phase of the D-MPNN to apply the feed-forward neural network f to the concatenation of the learned molecular feature vector h and the computed global features h_f:

$$\hat{y} = f(\mathrm{cat}(h, h_f)).$$

This is a very general method of incorporating external information into the model and can be used with any MPNN and with any computed features or descriptors.

3.3.2 Hyperparameter Optimization

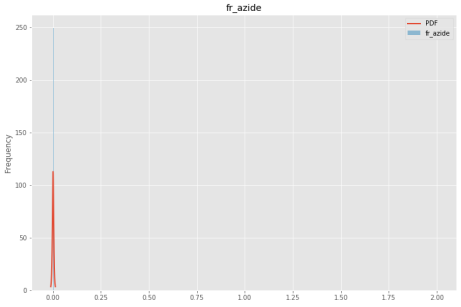
The performance of MPNNs, like most neural networks, can depend largely on the settings of the various model hyperparameters, such as the hidden size of the neural network layers. Thus to maximize performance, I performed hyperparameter optimization via Bayesian Optimization [41] using the Hyperopt[2] Python package. I specifically optimized the model's depth (number of message passing steps), hidden size (size of bond message vectors), number of feed-forward network layers, and dropout probability.

3.3.3 Ensembling

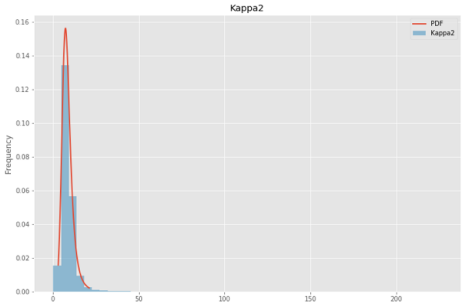
A common technique in machine learning for improving model performance is ensembling, where the predictions of multiple independently trained models are combined to produce a more accurate prediction [14] (see Figure 3-3). I applied this technique by training several copies of our model, each initialized with different random weights, and then averaging the predictions of these models (each with equal weight) to generate an ensemble prediction.

Since prior work did not report performance using ensembling, all direct comparisons I make to prior work use a single D-MPNN model for a fair comparison.

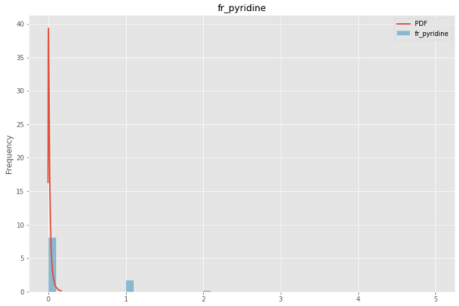
[2] https://github.com/hyperopt/hyperopt



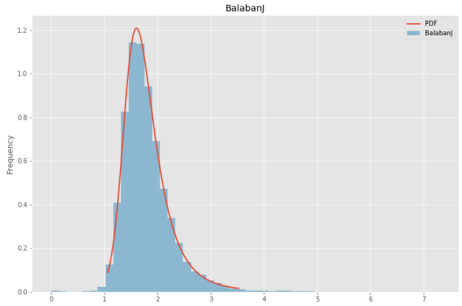
(a) fr_azide



(b) Kappa2



(c) fr_pyridine



(d) BalabanJ

Figure 3-2: Four example distributions fit to a random sample of 100,000 compounds used for biological screening in Novartis. Note that some distributions for discrete calculations, such as fr_pyridine, are not fit especially well. This is an active area for improvement.

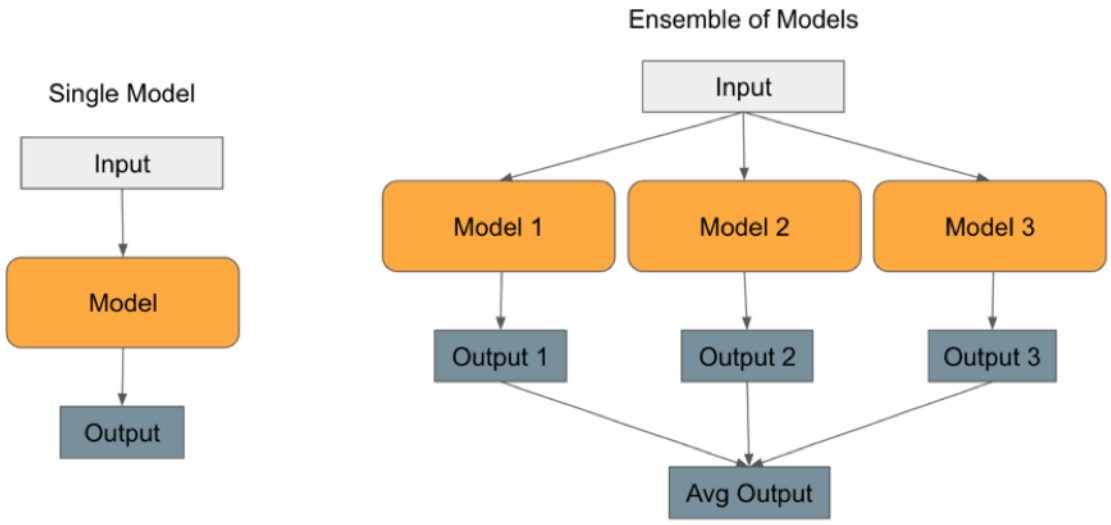


Figure 3-3: An illustration of ensembling models. On the left is a single model, which takes input and makes a prediction. On the right is an ensemble of 3 models. Each model takes the same input and makes a prediction independently, and then the predictions are averaged to generate the ensemble's prediction.

However, I also report results using an ensemble to illustrate the maximum possible performance of the model architecture.

3.4 Implementation

I implemented our model using the PyTorch [34] deep learning framework. All code for the D-MPNN and its variants is available in the chemprop GitHub repository [2][3]. Code for computing and using the RDKit feature CDFs is available in the Descriptastorus package [3][4]. Additionally, a web demonstration of the model's predictive capability on public datasets is available online at http://chemprop.csail.mit.edu [5].

[3] All code was written in conjunction with Kevin Yang.
[4] Code written by Brian Kelley from Novartis.
[5] Website created in conjunction with Lior Hirschfeld.

Chapter 4

Experiments

4.1 Data

I tested the D-MPNN on 19 publicly available datasets from [46] and [32]. These datasets include a wide range of regression and classification targets spanning quantum mechanics, physical chemistry, biophysics, and physiology. The datasets range in size from less than 200 molecules to over 450,000 molecules. Summary statistics for all the datasets are provided in Table 4.1[1]. Additional information on the class balance of the classification datasets is provided in Appendix B[2].

4.2 Experimental Procedure

Cross-Validation and Hyperparameter Optimization. Since many of the datasets are very small (two thousand molecules or fewer), I used a cross-validation approach to decrease noise in the results both while optimizing the hyperparameters and while

[1] For some datasets, the number of compounds in Table 4.1 does not precisely match the numbers from [46] because I removed a small number of molecules which could not be processed by RDKit. Furthermore, I have fewer molecules in QM7 because I used SMILES strings generated by [46] from the original 3D coordinates in the dataset, but the SMILES conversion process failed for ~ 300 molecules. (Note that QM7 and some of the other MoleculeNet datasets are more commonly used for benchmarking models that leverage 3D information, rather than purely SMILES-based methods like mine.)

[2] The MUV dataset is particularly unbalanced, with only 0.2% of molecules classified as positive. This makes D-MPNN training unstable, leading to the wide variation in performance in the subsequent sections.

Category	Dataset	# Tasks	Task Type	# Compounds	Metric
Quantum Mechanics	QM7	1	Regression	6,830	MAE
Quantum Mechanics	QM8	12	Regression	21,786	MAE
Quantum Mechanics	QM9	12	Regression	133,884	MAE
Physical Chemistry	ESOL	1	Regression	1,128	RMSE
Physical Chemistry	FreeSolv	1	Regression	642	RMSE
Physical Chemistry	Lipophilicity	1	Regression	4,200	RMSE
Biophysics	PDBbind-F	1	Regression	9,880	RMSE
Biophysics	PDBbind-C	1	Regression	168	RMSE
Biophysics	PDBbind-R	1	Regression	3,040	RMSE
Biophysics	PCBA	128	Classification	437,928	PRC-AUC
Biophysics	MUV	17	Classification	93,087	PRC-AUC
Biophysics	HIV	1	Classification	41,127	ROC-AUC
Biophysics	BACE	1	Classification	1,513	ROC-AUC
Physiology	BBBP	1	Classification	2,039	ROC-AUC
Physiology	Tox21	12	Classification	7,831	ROC-AUC
Physiology	ToxCast	617	Classification	8,576	ROC-AUC
Physiology	SIDER	27	Classification	1,427	ROC-AUC
Physiology	ClinTox	2	Classification	1,478	ROC-AUC
Physiology	ChEMBL	1,310	Classification	456,331	ROC-AUC

Table 4.1: Summary statistics of the public datasets used in this thesis. Note: PDBbind-F, PDBbind-C, and PDBbind-R refer to the full, core, and refined PDBbind datasets from [46].

determining final performance numbers. For consistency, I maintained the same approach for all of the datasets. Specifically, for each dataset, I used 50 iterations of Bayesian optimization on 3 randomly-seeded 80:10:10 data splits to determine the best hyperparameters. Then I used 10 additional randomly-seeded 80:10:10 data splits to determine the performance of the best set of hyperparameters. Due to computational cost, I used only 1 randomly-seeded data split (instead of 3) during hyperparameter optimization on QM9, MUV, PCBA, and ChEMBL. On PCBA and ChEMBL, I performed 20 iterations of Bayesian optimization and I used 3 data splits (instead of 10) when evaluating at test time with the best hyperparameters.

When optimizing hyperparameters, I considered it desirable to share a single set of hyperparameters across all of the 80:10:10 test splits, so that it is possible to report a single set of optimized hyperparameters for future use. Therefore, to determine the best hyperparameters, I took the set of hyperparameters with the best average performance on the 3 validation splits. Although the hyperparameter selection may be informed by the full dataset as a result, I minimized this effect by testing on a new set of 10 randomly-seeded splits. When I ran the best model from [32] for comparative purposes, I optimized their model's hyperparameters in the same fashion. I report the best hyperparameters in Appendix E.

Split Type. While I only optimized the models on random splits of the data[3], I evaluated the optimized models on both random and scaffold-based splits, as well as on the original splits from [46] and [32]. The scaffold split I used is similar to that of [46]. Molecules are partitioned into bins based on their Murcko scaffold calculated by RDKit [26]. Any bins larger than half of the desired test set size are placed into the training set, in order to guarantee the scaffold diversity of the validation and test sets. All remaining bins are placed randomly into the training, validation, and test sets until each set has reached its desired size.

[3] I performed hyperparameter optimization on several of the datasets using a scaffold split, but I found that performance was comparable to using the hyperparameters selected from performing optimization on a random split. Due to computation time limitations, I therefore only performed optimization on a random split and used those hyperparameter settings for both random and scaffold splits.

Compared to a random split, a scaffold split is a more challenging and realistic evaluation setting as it more closely approximates the chronological split present in real-world property prediction data. Since chronological information is not available for most public datasets, I used a scaffold-based split for all evaluations except for the direct comparison with the MoleculeNet models from [46], for which I used their original data splits.

Baselines. I compared my model to the following baselines:

- The best model for each dataset from [46].
- The best model from [32], a feed-forward neural network on a concatenation of assorted expert-designed molecular fingerprints.
- Random forest on binary Morgan fingerprints.
- Feed-forward network (FFN) on binary Morgan fingerprints using the same FFN architecture that my D-MPNN uses during its readout phase.
- FFN on count-based Morgan fingerprints.
- FFN on RDKit-calculated fingerprints.

The models in [46] include MPNN [20], Weave [22], GraphConv, kernel ridge regression, gradient boosting [19], random forest [6], logistic regression [12], directed acyclic graph models [29], support vector machines [11], Deep Tensor Neural Networks [40], multitask networks [30], bypass networks [36], influence relevance voting [43], and/or ANI-1 [42], depending on the dataset. Full details can be found in [46]. For the feed-forward network model from [32], I modified the authors' original code with their guidance in order to run their code on all of the datasets, not just on the ChEMBL dataset they experimented with. I tuned learning rates and hidden dimensions in addition to the extensive hyperparameter search already present in their code.

Chapter 5

Results

In this chaper, I analyze the performance of my model on a number of public datasets. All results are displayed as plots showing change relative to a baseline model rather than showing absolute performance numbers. This is because different datasets use different metrics and the scale of the performance numbers can differ drastically between datasets. For regression datasets, the plots show error (either root-mean-square error (RMSE) or mean absolute error (MAE)) relative to the baseline model, meaning lower is better, while for classification datasets, the plots show area under the curve (AUC, either area under the receiver operating characteristic curve (ROC-AUC) or area under the precision recall curve (PRC-AUC)) relative to the baseline model, meaning higher is better. Table 4.1 indicates the metric used for each dataset. Tables showing the exact performance numbers for all experiments can be found in the Appendices. Note that the error bars on all plots show the standard error of the mean across multiple runs, where standard error is defined as the standard deviation divided by the square root of the number of runs.

5.1 Comparison to Baselines

After optimizing my model, I compared my best single (non-ensembled) model on each dataset against models from prior work.

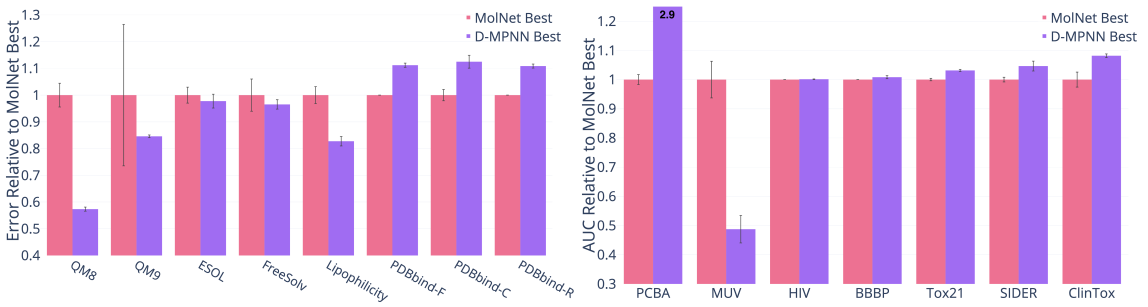
5.1.1 Comparison to MoleculeNet

I first compared the D-MPNN to the best model from MoleculeNet [46] on the same datasets and splits on which [46] evaluated their models. I was unable to reproduce their original data splits on BACE, Toxcast, and QM7, but I have evaluated my model against their original splits on all of the other datasets. The splits are a mix of random, scaffold, and time splits, as indicated in Figure 5-1.

I observed that D-MPNN performs significantly better than the best model for each dataset in MoleculeNet, with two exceptions. The first is the MUV dataset, which is large but extremely imbalanced; only 0.2% of samples are labeled as positives. [46] also encountered great difficulty with this extreme class imbalance when experimenting with the MUV dataset; all other datasets I experiment on contain at least 1% positives (see Appendix B for full class balance information). The second exception is the three variants of the PDBbind dataset, where there is auxiliary 3D information available. The current iteration of the D-MPNN does not use 3D coordinate information, and I leave this extension to future work. Thus it is unsurprising that the D-MPNN underperforms models using 3D information on a protein binding affinity prediction task such as PDBbind, where 3D structure is key. Nevertheless, the D-MPNN outperforms the best graph-based method in MoleculeNet on PDBbind. Moreover, note that on other datasets that provide 3D coordinate information (QM8 and QM9), my model outperforms the best model in MoleculeNet with or without 3D coordinates.

5.1.2 Comparison to Mayr et al

In addition, I compared the D-MPNN to the baseline from [32] in Figure 5-2. I reproduced the features from their best model on each dataset using their scripts or equivalent packages [4]. I then ran their code and hyperparameter optimization directly on the classification datasets, and I modified their code to run on regression datasets with the authors' guidance [4]. On most classification datasets, the D-MPNN matches or outperforms the baseline in [32], using either no human-engineered fea-

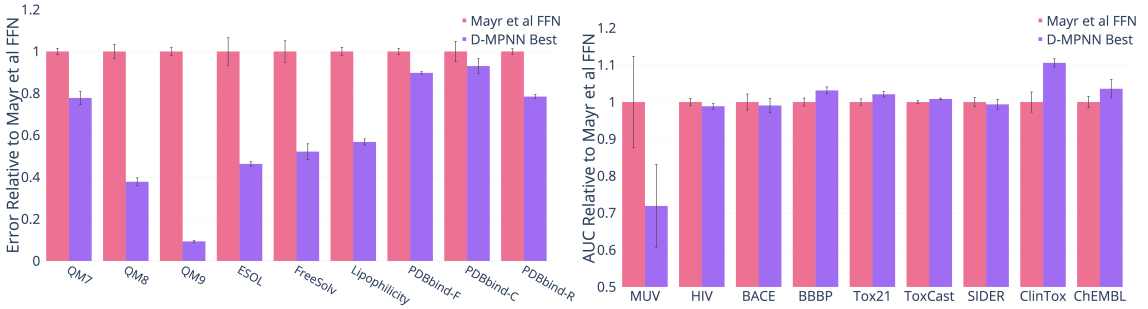


(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-1: Comparison of my best single model (i.e. optimized hyperparameters and optionally RDKit features but without ensembling) to the best models from [46]. Note that D-MPNN significantly outperforms MoleculeNet on PCBA, so much so that the bar extends beyond the top of the chart.

tures (D-MPNN) or a small number of computed molecule-level features (D-MPNN Features). On regression datasets, the baseline from [32] performs poorly in comparison, despite extensive tuning. I hypothesize that this poor performance on regression in comparison to classification is the result of a large number of binary input features to the output feed-forward network; this hypothesis is supported by the similarly poor performance of the Morgan fingerprint FFN baseline. In addition, their method does not employ early stopping based on validation set performance and therefore may overfit to the training data in some cases.

Note that while the D-MPNN outperforms [32]'s method on my sca□old split



(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-2: Comparison of my best single model (i.e. optimized hyperparameters and optionally RDKit features) to the model from . Note that PCBA is omitted as I found [32]'s model to be numerically unstable on this dataset.

of the ChEMBL dataset, the reverse is true on [32]'s original splits, which are also scaffold-based but using a different methodology. Interestingly, while the D-MPNN achieves a lower AUC on [32]'s original splits than on my own splits, [32]'s model achieves a lower AUC on my scaffold splits than on their original splits.

5.1.3 Out-of-the-Box Comparison to Other Baselines

For my final baseline comparison, I evaluated my model's performance "out-of-the-box," i.e. using all the default settings (hidden size = 300, depth = 3, number of feed-forward layers = 2, dropout = 0) without any hyperparameter optimization and without any additional features. For this comparison, I compared to a number of simple baseline models that use computed fingerprints or descriptors:

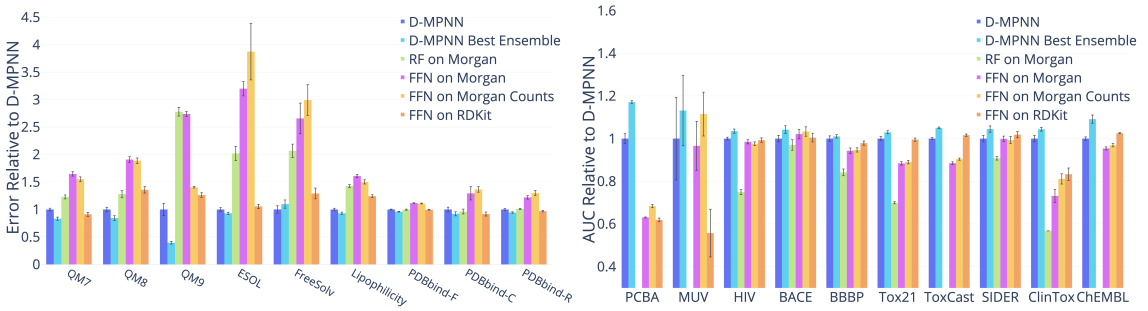
1. Random forest (RF) with 500 trees run on Morgan (ECFP) fingerprints using radius 2 and hashing to a bit vector of size 2048.
2. Feed-forward network (FFN) on Morgan fingerprints.
3. FFN on Morgan fingerprints which use substructure counts instead of bits.
4. FFN on RDKit features.

The parameters of the simple baseline models are also out-of-the-box defaults. I made this comparison in order to demonstrate the strong out-of-the-box performance of my model across a wide variety of datasets. Finally, I include the performance of the optimized version of my model as a reference.

Figure 5-3 shows that even without optimization, the D-MPNN provides an excellent starting point on a wide variety of datasets and targets, though it can be improved further with proper optimization.

5.2 Ablations

Next, I analyze and justify the choice of using directed messages in the D-MPNN.

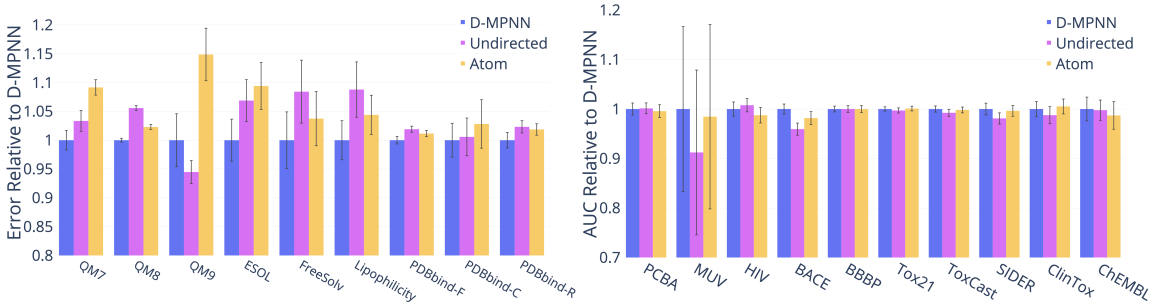


(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-3: Comparison of my unoptimized D-MPNN against several baseline models. I omitted the random forest baseline on PCBA, MUV, Toxcast, and ChEMBL due to its large computational cost.

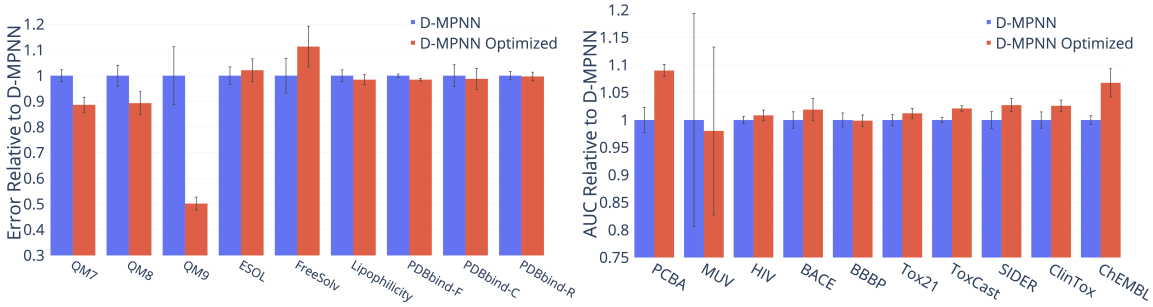
5.2.1 Message Type

The most important distinction between the D-MPNN and related work is the nature of the messages being passed across the molecule. Most prior work uses messages centered on atoms whereas the D-MPNN uses messages centered on directed bonds. To isolate the effect of the message passing paradigm on property prediction performance, I implemented message passing on undirected bonds and on atoms as well, as detailed in Appendix D. Figure 5-4 illustrates the differences in performance between these three types of message passing. While performance is similar on most classification datasets, messages centered on directed bonds outperform messages centered on undirected bonds and on atoms on several of the datasets, indicating the benefit of the D-MPNN's directed bond approach.



(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-4: Comparison of performance of different message passing paradigms.



(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-5: E□ect of performing Bayesian hyperparameter optimization on the depth, hidden size, number of fully connected layers, and dropout of the D-MPNN.

5.3 Model Optimizations

In the following sections, I analyze the impact of each of the model optimizations I introduced.

5.3.1 Hyperparameter Optimization

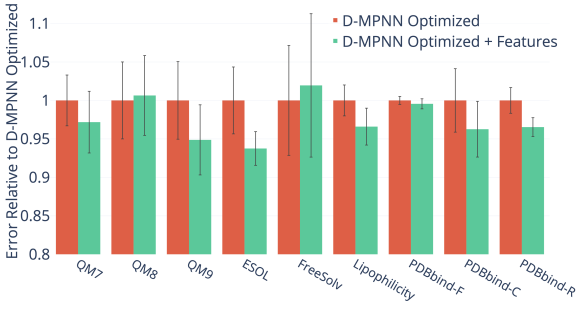
To improve model performance, I performed Bayesian Optimization to select the best model hyperparameters for each dataset. Figure 5-5 illustrates the benefit of performing this optimization, as model performance improves on virtually every dataset. Interestingly, some datasets are particularly sensitive to hyperparameters. While most datasets experience a moderate 2-5% improvement in performance following hyperparameter optimization, the quantum mechanics datasets (QM7, QM8, and QM9) and PCBA see dramatic improvements in performance, with the D-MPNN model performing 37% better on QM9 after optimization.

5.3.2 RDKit Features

After selecting the best hyperparameters for each dataset, I examined the impact of adding additional molecule-level features from RDKit to my model. Figure 5-6 shows the e□ect on model performance. The results appear to be highly dataset-dependent. Some datasets, such as QM9 and ESOL, show marked improvement with

the addition of features, while other datasets, such as PCBA and HIV, actually show worse performance with the features. I hypothesize that this is because the features are particularly relevant to certain tasks while possibly confusing and distracting the model on other tasks. This implies that my model's performance on a given dataset may be further optimized by selecting di□erent features more relevant to the task of interest.

Another interesting trend is the e□ect of adding features to the three PDBbind datasets. The features appear to help on all three datasets, but the benefit is much more pronounced on the extremely small PDBbind-C (core) dataset than it is on the larger PDBbind-R (refined) and PDBbind-F (full) datasets. This indicates that the features may help compensate for the lack of training data and thus may be particularly relevant in low-data regimes. In particular, I hypothesize that the features may help to regularize a representation derived from a small dataset: because the features are derived from more general chemical knowledge, they implicitly provide the model some understanding of a larger chemical domain. Thus, it is worthwhile to consider the addition of features both when they are particularly relevant to the task of interest and when the dataset is especially small.



(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-6: E□ect of adding molecule-level features generated with RDKit to my model.

5.3.3 Ensembling

To maximize performance, I trained an ensemble of models. For each dataset, I selected the best single model—i.e. the best hyperparameters along with the RDKit features if the features improved performance—and I trained five models instead of one. The results appear in Figure 5-7. On most datasets, ensembling only provides a small 1-5% benefit, but as with hyperparameter optimization, there are certain datasets, particularly the quantum mechanics datasets, which especially benefit from the effect of ensembling.

5.4 Effect of Data Size

Finally, I analyze the effect of data size on the performance of my model, using the ChEMBL dataset. ChEMBL is a large dataset of 456,331 molecules on 1,310 targets, but is extremely sparse: only half of the 1,310 targets have at least 300 labels. For this analysis, I used the original scaffold-based split of [32], containing 3 cross-validation folds. From Figure 5-8, it is apparent that the D-MPNN struggles on low-label targets in comparison to this baseline. As the D-MPNN model does not use any human-engineered fingerprints and must therefore learn its features completely from scratch based on the input data, it is unsurprising that the average ROC-AUC score of D-MPNN is worse than that of the feed-forward network running on human-engineered descriptors in [32]. When I filtered the ChEMBL dataset by pruning low-data targets

(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

Figure 5-7: Effect of using an ensemble of five models instead of a single model.

at di□erent thresholds, I found that the D-MPNN indeed outperforms the best model of [32] at larger data thresholds, even on [32]'s splits (see Figure 5-8).

Figure 5-8: E□ect of data size on the performance of the model from [32] and of the D-MPNN model (higher = better).

Chapter 6

Conclusion and Future Work

In this thesis, I performed an extensive evaluation of the D-MPNN, both in its original and optimized form. I found that the D-MPNN shows consistent, strong performance out-of-the-box, and it achieves state-of-the-art performance on a number of publicly available datasets with additional optimizations. These results indicate that the D-MPNN outperforms both models based on molecular descriptors or fingerprints and prior graph convolutional models. Therefore, the D-MPNN should be the go-to model whenever experimenting with molecular property prediction.

However, there is still significant room for improvement. The biggest limitation of the current message passing paradigm is that it operates only on the graph structure of the molecule without incorporating any 3-dimensional information. This is a severe drawback because many molecular properties, such as protein binding a□nity, are highly dependent on 3D information that is completely missing in the message passing approach.

Another avenue for improvement is pretraining. Many of the datasets used in this thesis are relatively small, meaning even the most powerful model only has limited information to learn from. In other fields of machine learning, such as natural language processing, it is common to train models on large, semi-related datasets prior to training on the small dataset of interest. Doing so allows models to build a strong understanding of the input space that can then be leveraged when learning the task of interest. An e□ective pretraining approach for molecular property prediction could

have a very significant e□ect due to the abundance of small datasets in chemistry.

Appendix A

Code and Website

My code is publicly available at https://github.com/swansonk14/chemprop , which also includes a web interface that supports non-programmatic training and predicting with the model. Code for computing the RDKit features is available at https://github.com/bp-kelley/descriptastorus . A public web demonstration of the model's prediction capability on public datasets is available at http://chemprop.csail.mit.edu . See Figure A-1 for a screenshot of the web demo. Additionally, I ran the baseline from [32] using the code at https://github.com/yangkevin2/lsc_experiments .

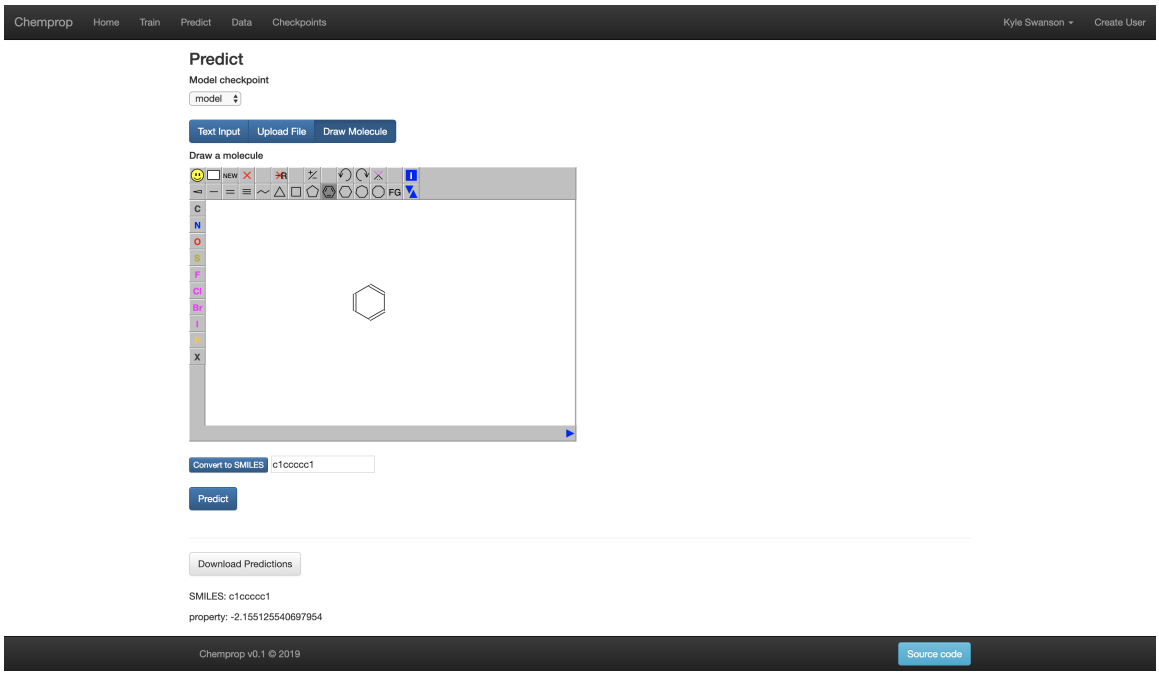


Figure A-1: Screenshot of the prediction page of the web interface.

Appendix B

Additional Dataset Statistics

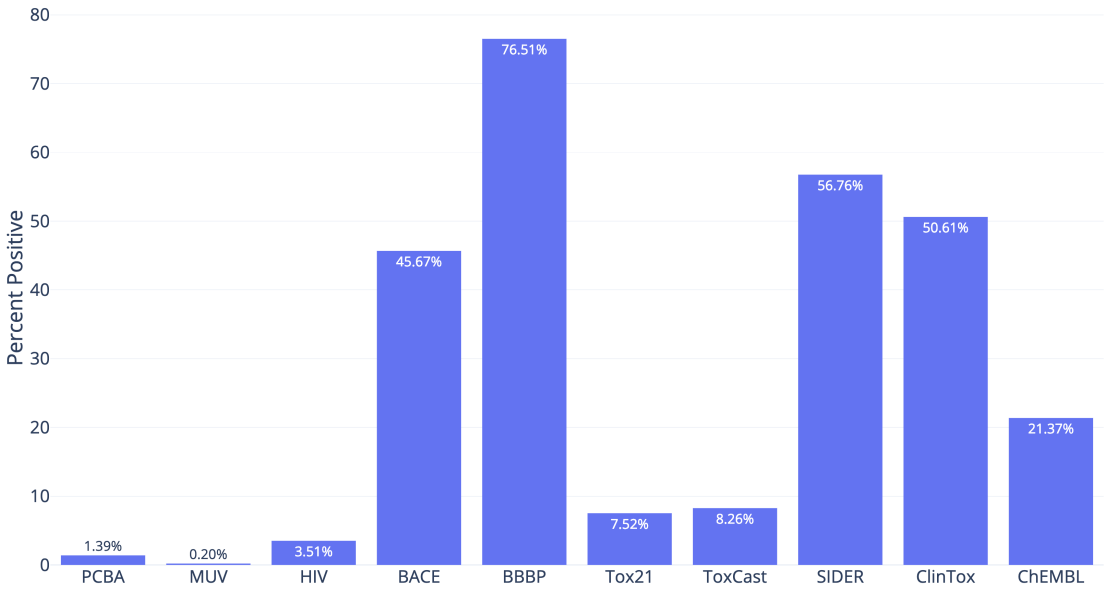


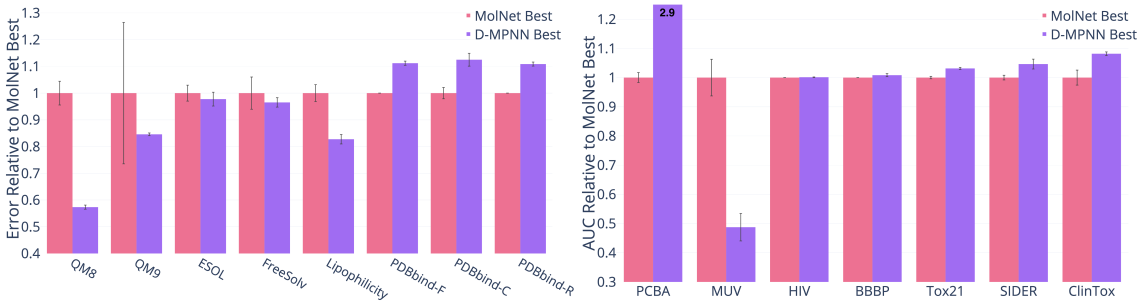
Figure B-1: Class balance on the publicly available classification datasets. The Y-axis is the average percent of positives in the tasks in a dataset, weighted by the number of molecules with known values for each task.

Appendix C

Comparison to Baselines

C.1 Comparison to MoleculeNet

Comparison between the best single D-MPNN model (i.e. optimized hyperparameters and optionally RDKit features but without ensembling) and the best model from MoleculeNet using the splits from MoleculeNet [46]. I was unable to reproduce the splits from MoleculeNet on QM7, BACE, and ToxCast, so I leave out those datasets. Note: the QM8, QM9, and PDBbind datasets include 3D coordinates that the D-MPNN does not use but some MoleculeNet models may use.



(a) Regression Datasets (lower = better). (b) Classification Datasets (higher = better).

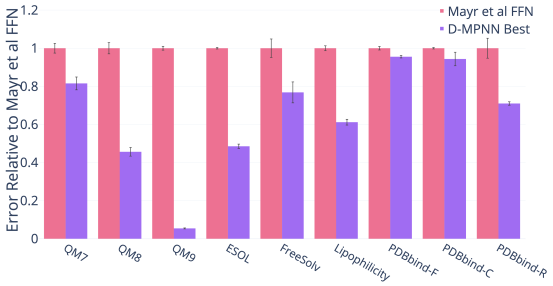
Figure C-1: Comparison to MoleculeNet models on [46]'s original splits. Note that D-MPNN significantly outperforms MoleculeNet on PCBA, so much so that the bar extends beyond the top of the chart.

Dataset	Metric	Split Type	MolNet Best	D-MPNN Best
QM8	MAE	Random	0.0143 ± 0.0011	0.0082 ± 0.0002 (-42.66%)
QM9	MAE	Random	2.400 ± 1.100	2.030 ± 0.021 (-15.42%)
ESOL	RMSE	Random	0.580 ± 0.030	0.567 ± 0.026 (-2.24%)
FreeSolv	RMSE	Random	1.150 ± 0.120	1.110 ± 0.035 (-3.48%)
Lipophilicity	RMSE	Random	0.655 ± 0.036	0.542 ± 0.020 (-17.25%)
PDBbind-F	RMSE	Time	1.250 ± 0.000	1.390 ± 0.017 (+11.20%)
PDBbind-C	RMSE	Time	1.920 ± 0.070	2.160 ± 0.080 (+12.50%)
PDBbind-R	RMSE	Time	1.380 ± 0.000	1.530 ± 0.018 (+10.87%)
PCBA	PRC-AUC	Random	0.136 ± 0.004	0.397 ± 0.001 (+191.91%)
MUV	PRC-AUC	Random	0.1840 ± 0.0200	0.090 ± 0.015 (-51.25%)
HIV	ROC-AUC	Sca□old	0.792 ± 0.000	0.793 ± 0.001 (+0.13%)
BBBP	ROC-AUC	Sca□old	0.729 ± 0.000	0.735 ± 0.006 (+0.82%)
Tox21	ROC-AUC	Random	0.829 ± 0.006	0.855 ± 0.005 (+3.14%)
SIDER	ROC-AUC	Random	0.648 ± 0.009	0.678 ± 0.019 (+4.63%)
ClinTox	ROC-AUC	Random	0.832 ± 0.037	0.900 ± 0.009 (+8.17%)

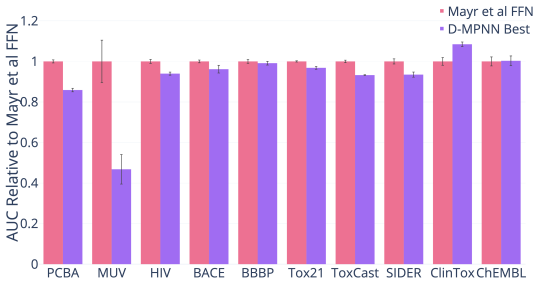
Table C.1: Comparison to MoleculeNet models on [46]'s original splits.

C.2 Comparison to Mayr et al

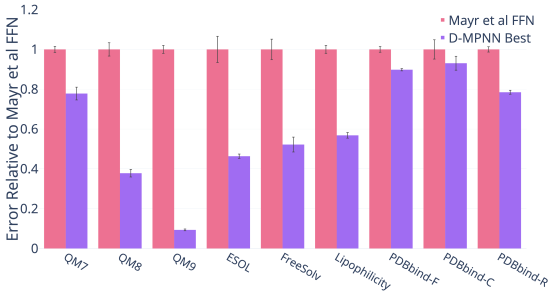
Comparison between the best single D-MPNN model (i.e. optimized hyperparameters and optionally RDKit features but without ensembling) and the feed-forward network (FFN) architecture of [32] using their best descriptor set. PCBA is omitted because the method of [32] is unstable on that dataset.



(a) Regression Datasets (Random Split, lower = better).



(b) Classification Datasets (Random Split, higher = better).



(c) Regression Datasets (Sca□old Split, lower = better).

(d) Classifiation Datasets (Sca□old Split, higher = better).

Figure C-2: Comparison to Mayr et al.

Dataset	Metric	Mayr et al FFN	D-MPNN Best
QM7	11.494	108.100 ± 8.750	60.106 ± 2.250 (-44.40%)
QM8	0.002092	0.029 ± 0.003	0.008 ± 0.000 (-71.38%)
QM9	0.374544	45.400 ± 1.460	1.992 ± 0.017 (-95.61%)
ESOL	0.061	1.700 ± 0.021	0.620 ± 0.041 (-63.53%)
FreeSolv	0.486	2.800 ± 0.430	1.027 ± 0.211 (-63.32%)
Lipophilicity	0.049	1.020 ± 0.041	0.557 ± 0.058 (-45.39%)
PDBbind-F	0.028	1.410 ± 0.041	1.287 ± 0.027 (-8.72%)
PDBbind-C	0.243601	2.170 ± 0.025	1.933 ± 0.143 (-10.92%)
PDBbind-R	0.056603	1.990 ± 0.330	1.325 ± 0.060 (-33.40%)
MUV	0.038943	0.169 ± 0.056	0.126 ± 0.056 (-25.68%)
HIV	0.020414	0.850 ± 0.027	0.827 ± 0.029 (-2.69%)
BACE	0.053	0.886 ± 0.019	0.889 ± 0.024 (+0.34%)
BBBP	0.027	0.927 ± 0.027	0.931 ± 0.016 (+0.43%)
Tox21	0.019	0.853 ± 0.008	0.862 ± 0.008 (+1.06%)
ToxCast	0.005	0.770 ± 0.015	0.769 ± 0.015 (-0.13%)
SIDER	0.027	0.676 ± 0.027	0.658 ± 0.018 (-2.66%)
ClinTox	0.029	0.827 ± 0.050	0.903 ± 0.032 (+9.19%)
ChEMBL	0.031382	0.755 ± 0.029	0.767 ± 0.035 (+1.58%)

Table C.2: Comparison to Mayr et al (Random Split).

Dataset	Metric	Mayr et al FFN		D-MPNN Best	
QM7	MAE	113.250	5.440	88.136	11.494 (-22.18%)
QM8	MAE	0.035	0.004	0.013	0.002 (-62.19%)
QM9	MAE	26.400	1.680	2.465	0.375 (-90.66%)
ESOL	RMSE	1.780	0.370	0.825	0.061 (-53.65%)
FreeSolv	RMSE	4.120	0.670	2.151	0.486 (-47.79%)
Lipophilicity	RMSE	1.098	0.068	0.624	0.049 (-43.17%)
PDBbind-F	RMSE	1.500	0.068	1.347	0.028 (-10.20%)
PDBbind-C	RMSE	2.200	0.337	2.047	0.244 (-6.94%)
PDBbind-R	RMSE	1.800	0.076	1.413	0.057 (-21.52%)
MUV	PRC-AUC	0.110	0.043	0.079	0.039 (-28.10%)
HIV	ROC-AUC	0.808	0.025	0.799	0.020 (-1.14%)
BACE	ROC-AUC	0.860	0.059	0.852	0.053 (-0.93%)
BBBP	ROC-AUC	0.891	0.030	0.919	0.027 (+3.14%)
Tox21	ROC-AUC	0.809	0.022	0.826	0.019 (+2.10%)
ToxCast	ROC-AUC	0.712	0.010	0.718	0.005 (+0.84%)
SIDER	ROC-AUC	0.636	0.025	0.632	0.027 (-0.63%)
ClinTox	ROC-AUC	0.811	0.070	0.897	0.029 (+10.60%)
ChEMBL	ROC-AUC	0.731	0.019	0.757	0.031 (+3.59%)

Table C.3: Comparison to Mayr et al (Sca old Split).

C.3 Comparison to Other Baselines

Comparison to several baselines using feed-forward neural networks on molecular ngerprints or descriptors.

(a) Regression Datasets (Random Split, lower = better).

(b) Classi cation Datasets (Random Split, higher = better).

(c) Regression Datasets (Sca old Split, lower = better).

(d) Classi cation Datasets (Sca old Split, higher = better).

Figure C-3: Comparison to Baselines.

Dataset	Metric	D-MPNN		D-MPNN Best Ensemble	
QM7	MAE	67.657	3.647	57.952	2.674 (-14.35%)
QM8	MAE	0.0111	0.0001	0.0077	0.0001 (-30.23%)
QM9	MAE	3.705	0.537	1.585	0.011 (-57.22%)
ESOL	RMSE	0.691	0.080	0.607	0.051 (-12.13%)
FreeSolv	RMSE	1.133	0.176	0.999	0.198 (-11.81%)
Lipophilicity	RMSE	0.598	0.064	0.548	0.050 (-8.38%)
PDBbind-F	RMSE	1.325	0.027	1.264	0.026 (-4.61%)
PDBbind-C	RMSE	2.088	0.194	1.910	0.137 (-8.55%)
PDBbind-R	RMSE	1.411	0.060	1.318	0.063 (-6.62%)
PCBA	PRC-AUC	0.329	0.007	0.411	0.008 (+24.96%)
MUV	PRC-AUC	0.0865	0.0456	0.1356	0.0660 (+56.71%)
HIV	ROC-AUC	0.811	0.037	0.843	0.030 (+3.96%)
BACE	ROC-AUC	0.863	0.028	0.893	0.019 (+3.45%)
BBBP	ROC-AUC	0.920	0.017	0.930	0.017 (+1.07%)
Tox21	ROC-AUC	0.847	0.012	0.868	0.008 (+2.46%)
ToxCast	ROC-AUC	0.735	0.014	0.777	0.014 (+5.73%)
SIDER	ROC-AUC	0.639	0.024	0.660	0.018 (+3.27%)
ClinTox	ROC-AUC	0.892	0.043	0.917	0.023 (+2.77%)

Table C.4: Comparison to Baselines, Part I (Random Split).

Dataset	RF on Morgan		FFN on Morgan	
QM7	124.376	4.278 (+83.83%)	137.932	3.420 (+103.87%)
QM8	0.0140	0.0003 (+26.37%)	0.0236	0.0003 (+112.89%)
QM9	14.104	0.125 (+280.65%)	12.258	0.058 (+230.82%)
ESOL	1.176	0.076 (+70.25%)	1.953	0.205 (+182.78%)
FreeSolv	2.146	0.538 (+89.40%)	2.989	0.533 (+163.76%)
Lipophilicity	0.842	0.026 (+40.84%)	0.925	0.035 (+54.74%)
PDBbind-F	1.309	0.027 (-1.23%)	1.422	0.021 (+7.28%)
PDBbind-C	2.124	0.181 (+1.69%)	2.776	0.617 (+32.92%)
PDBbind-R	1.377	0.062 (-2.40%)	1.590	0.138 (+12.63%)
PCBA	0.000	0.000 (-100.00%)	0.256	0.005 (-22.02%)
MUV	0.0000	0.0000 (-100.00%)	0.1207	0.0567 (+39.43%)
HIV	0.656	0.015 (-19.07%)	0.807	0.031 (-0.44%)
BACE	0.825	0.021 (-4.39%)	0.868	0.022 (+0.52%)
BBBP	0.803	0.036 (-12.71%)	0.902	0.027 (-1.94%)
Tox21	0.619	0.010 (-26.96%)	0.792	0.012 (-6.53%)
ToxCast	0.000	0.000 (-100.00%)	0.687	0.018 (-6.54%)
SIDER	0.567	0.008 (-11.29%)	0.649	0.026 (+1.54%)
ClinTox	0.551	0.039 (-38.23%)	0.639	0.085 (-28.33%)

Table C.5: Comparison to Baselines, Part II (Random Split).

Dataset	FFN on Morgan Counts		FFN on RDKit	
QM7	123.449	3.806 (+82.46%)	76.300	2.828 (+12.77%)
QM8	0.0231	0.0002 (+108.16%)	0.0166	0.0002 (+49.67%)
QM9	6.002	0.053 (+61.97%)	6.170	0.060 (+66.53%)
ESOL	1.988	0.370 (+187.85%)	0.707	0.047 (+2.43%)
FreeSolv	3.054	0.345 (+169.48%)	1.363	0.296 (+20.32%)
Lipophilicity	0.900	0.044 (+50.57%)	0.746	0.046 (+24.83%)
PDBbind-F	1.423	0.030 (+7.37%)	1.325	0.024 (-0.05%)
PDBbind-C	2.837	0.483 (+35.86%)	1.883	0.124 (-9.81%)
PDBbind-R	1.589	0.140 (+12.57%)	1.357	0.048 (-3.87%)
PCBA	0.264	0.002 (-19.64%)	0.210	0.004 (-36.24%)
MUV	0.1204	0.0474 (+39.16%)	0.0759	0.0418 (-12.34%)
HIV	0.803	0.038 (-0.96%)	0.811	0.028 (+0.03%)
BACE	0.870	0.030 (+0.73%)	0.847	0.031 (-1.94%)
BBBP	0.901	0.035 (-2.03%)	0.911	0.023 (-1.02%)
Tox21	0.789	0.010 (-6.85%)	0.834	0.010 (-1.48%)
ToxCast	0.694	0.017 (-5.63%)	0.742	0.015 (+0.91%)
SIDER	0.637	0.027 (-0.39%)	0.664	0.024 (+3.96%)
ClinTox	0.662	0.070 (-25.85%)	0.755	0.075 (-15.33%)

Table C.6: Comparison to Baselines, Part III (Random Split).

Dataset	Metric	D-MPNN		D-MPNN Best Ensemble	
QM7	MAE	102.302	7.532	85.173	9.300 (-16.74%)
QM8	MAE	0.0148	0.0019	0.0125	0.0021 (-15.66%)
QM9	MAE	5.177	1.848	2.043	0.391 (-60.54%)
ESOL	RMSE	0.862	0.094	0.801	0.057 (-7.07%)
FreeSolv	RMSE	1.932	0.412	2.122	0.460 (+9.83%)
Lipophilicity	RMSE	0.656	0.047	0.611	0.045 (-6.92%)
PDBbind-F	RMSE	1.374	0.028	1.319	0.024 (-4.05%)
PDBbind-C	RMSE	2.154	0.290	1.984	0.254 (-7.87%)
PDBbind-R	RMSE	1.468	0.075	1.387	0.057 (-5.51%)
PCBA	PRC-AUC	0.271	0.011	0.317	0.003 (+17.09%)
MUV	PRC-AUC	0.0807	0.0494	0.0913	0.0421 (+13.15%)
HIV	ROC-AUC	0.782	0.016	0.809	0.024 (+3.50%)
BACE	ROC-AUC	0.825	0.039	0.860	0.048 (+4.19%)
BBBP	ROC-AUC	0.915	0.037	0.925	0.023 (+1.03%)
Tox21	ROC-AUC	0.808	0.025	0.832	0.018 (+2.97%)
ToxCast	ROC-AUC	0.690	0.011	0.725	0.009 (+5.03%)
SIDER	ROC-AUC	0.606	0.029	0.633	0.029 (+4.47%)
ClinTox	ROC-AUC	0.874	0.041	0.913	0.024 (+4.37%)
ChEMBL	ROC-AUC	0.709	0.010	0.774	0.024 (+9.12%)

Table C.7: Comparison to Baselines, Part I (Sca old Split).

Dataset	RF on Morgan		FFN on Morgan	
QM7	125.661	11.322 (+22.83%)	168.496	12.789 (+64.71%)
QM8	0.0190	0.0030 (+27.91%)	0.0283	0.0025 (+90.90%)
QM9	14.392	1.306 (+178.01%)	14.177	0.743 (+173.85%)
ESOL	1.741	0.350 (+102.06%)	2.757	0.360 (+219.95%)
FreeSolv	3.994	0.751 (+106.70%)	5.133	1.706 (+165.68%)
Lipophilicity	0.936	0.057 (+42.63%)	1.056	0.061 (+60.99%)
PDBbind-F	1.370	0.043 (-0.33%)	1.533	0.035 (+11.52%)
PDBbind-C	2.074	0.299 (-3.71%)	2.785	0.816 (+29.33%)
PDBbind-R	1.481	0.057 (+0.92%)	1.791	0.149 (+22.02%)
PCBA	0.000	0.000 (-100.00%)	0.171	0.001 (-36.96%)
MUV	0.0000	0.0000 (-100.00%)	0.0780	0.0293 (-3.40%)
HIV	0.586	0.029 (-25.12%)	0.770	0.024 (-1.49%)
BACE	0.801	0.065 (-3.00%)	0.843	0.057 (+2.12%)
BBBP	0.770	0.046 (-15.82%)	0.863	0.039 (-5.70%)
Tox21	0.565	0.013 (-30.08%)	0.714	0.021 (-11.62%)
ToxCast	0.000	0.000 (-100.00%)	0.611	0.013 (-11.51%)
SIDER	0.549	0.015 (-9.27%)	0.605	0.025 (-0.12%)
ClinTox	0.496	0.000 (-43.23%)	0.639	0.085 (-26.87%)
ChEMBL	0.000	0.000 (-100.00%)	0.677	0.009 (-4.61%)

Table C.8: Comparison to Baselines, Part II (Sca old Split).

Dataset	FFN on Morgan Counts		FFN on RDKit	
QM7	158.701	13.798 (+55.13%)	93.275	12.208 (-8.82%)
QM8	0.0280	0.0024 (+88.68%)	0.0201	0.0028 (+35.73%)
QM9	7.271	0.228 (+40.45%)	6.539	0.694 (+26.30%)
ESOL	3.338	1.404 (+287.33%)	0.910	0.106 (+5.63%)
FreeSolv	5.779	1.707 (+199.11%)	2.495	0.598 (+29.15%)
Lipophilicity	0.986	0.080 (+50.23%)	0.815	0.052 (+24.26%)
PDBbind-F	1.524	0.034 (+10.91%)	1.368	0.032 (-0.43%)
PDBbind-C	2.939	0.341 (+36.45%)	1.976	0.251 (-8.26%)
PDBbind-R	1.908	0.205 (+30.01%)	1.422	0.062 (-3.10%)
PCBA	0.185	0.004 (-31.62%)	0.168	0.004 (-38.12%)
MUV	0.0900	0.0262 (+11.50%)	0.0450	0.0285 (-44.28%)
HIV	0.763	0.021 (-2.41%)	0.777	0.026 (-0.71%)
BACE	0.853	0.058 (+3.32%)	0.829	0.054 (+0.48%)
BBBP	0.866	0.029 (-5.31%)	0.895	0.028 (-2.14%)
Tox21	0.719	0.021 (-11.03%)	0.804	0.019 (-0.48%)
ToxCast	0.624	0.011 (-9.63%)	0.702	0.012 (+1.60%)
SIDER	0.601	0.032 (-0.69%)	0.617	0.027 (+1.84%)
ClinTox	0.709	0.066 (-18.89%)	0.729	0.080 (-16.67%)
ChEMBL	0.687	0.010 (-3.08%)	0.727	0.004 (+2.56%)

Table C.9: Comparison to Baselines, Part III (Sca old Split).

Appendix D

Ablations

D.1 Message Type

Below is a description of the implementation and performance of atom-based and undirected bond-based messages. For the most direct comparison, I implemented these as options in the D-MPNN code; the changes are only a few lines of code in each case. Therefore, in each case, I simply detail the differences from the directed bond-based messages.

D.1.1 Atom Messages

Messages are initialized based on atom features rather than bond features, according to $h_v^0 = \tau(W_i x_v)$ rather than $h_{vw}^0 = \tau(W_i \text{ cat}(x_v, e_{vw}))$, with matrix dimensions adjusted accordingly. During message passing, each atom receives messages according to $m_v^{t+1} = \sum_{k \in N(v)} h_k^t$. Finally, m_v is the sum of all of the atom hidden states at the end of message passing.

D.1.2 Undirected Bond Messages

The only difference between undirected bonds and the D-MPNN is that before each message passing step, for each pair of bonded atoms v and w, I set h_{vw}^t and h_{wv}^t to each be equal to their average. Consequently, the hidden state for each directed bond

is always equal to the hidden state of its reverse bond, resulting in message passing on undirected bonds.

As performance with this variant is on average worse than the D-MPNN, the additional symmetry breaking introduced by directed bonds appears to be useful.

D.1.3 Comparison of Di erent Message Types

Comparison of performance using di erent message passing paradigms. The D-MPNN uses directed messages.

(a) Regression Datasets (Random Split, lower = better).

(b) Classi cation Datasets (Random Split, higher = better).

(c) Regression Datasets (Sca old Split, lower = better).

(d) Classi cation Datasets (Sca old Split, higher = better).

Figure D-1: Message Type.

Dataset	Metric	D-MPNN		Undirected		Atom	
QM7	MAE	67.657	3.647	69.919	3.876 (+3.34%)	73.860	2.849 (+9.17%)
QM8	MAE	0.0111	0.0001	0.0117	0.0002 (+5.59%)	0.0114	0.0001 (+2.29%)
QM9	MAE	3.705	0.537	3.501	0.233 (-5.52%)	4.257	0.533 (+14.88%)
ESOL	RMSE	0.691	0.080	0.738	0.079 (+6.88%)	0.756	0.089 (+9.41%)
FreeSolv	RMSE	1.133	0.176	1.229	0.196 (+8.42%)	1.176	0.169 (+3.75%)
Lipophilicity	RMSE	0.598	0.064	0.650	0.091 (+8.79%)	0.624	0.064 (+4.40%)
PDBbind-F	RMSE	1.325	0.027	1.351	0.022 (+1.91%)	1.340	0.022 (+1.15%)
PDBbind-C	RMSE	2.088	0.194	2.100	0.216 (+0.58%)	2.147	0.278 (+2.82%)
PDBbind-R	RMSE	1.411	0.060	1.444	0.048 (+2.32%)	1.438	0.044 (+1.87%)
PCBA	PRC-AUC	0.329	0.007	0.329	0.006 (+0.13%)	0.327	0.007 (-0.43%)
MUV	PRC-AUC	0.0865	0.0456	0.0790	0.0455 (-8.77%)	0.0852	0.0510 (-1.54%)
HIV	ROC-AUC	0.811	0.037	0.817	0.034 (+0.77%)	0.800	0.040 (-1.26%)
BACE	ROC-AUC	0.863	0.028	0.828	0.034 (-4.06%)	0.848	0.036 (-1.83%)
BBBP	ROC-AUC	0.920	0.017	0.920	0.020 (+0.01%)	0.920	0.020 (-0.01%)
Tox21	ROC-AUC	0.847	0.012	0.844	0.014 (-0.29%)	0.848	0.013 (+0.10%)
ToxCast	ROC-AUC	0.735	0.014	0.729	0.017 (-0.76%)	0.734	0.014 (-0.19%)
SIDER	ROC-AUC	0.639	0.024	0.627	0.023 (-1.90%)	0.637	0.022 (-0.36%)
ClinTox	ROC-AUC	0.892	0.043	0.881	0.049 (-1.24%)	0.897	0.043 (+0.52%)
ChEMBL	ROC-AUC	0.745	0.031	0.743	0.026 (-0.24%)	0.735	0.036 (-1.30%)

Table D.1: Message Type (Random Split).

Dataset	Metric	D-MPNN		Undirected		Atom	
QM7	MAE	102.302	7.532	108.315	13.999 (+5.88%)	109.495	11.577 (+7.03%)
QM8	MAE	0.0148	0.0019	0.0170	0.0071 (+14.71%)	0.0151	0.0019 (+1.61%)
QM9	MAE	5.177	1.848	5.264	2.014 (+1.69%)	6.100	1.542 (+17.82%)
ESOL	RMSE	0.862	0.094	0.913	0.086 (+5.89%)	0.970	0.104 (+12.52%)
FreeSolv	RMSE	1.932	0.412	1.959	0.376 (+1.40%)	1.921	0.486 (-0.59%)
Lipophilicity	RMSE	0.656	0.047	0.682	0.065 (+3.90%)	0.686	0.069 (+4.54%)
PDBbind-F	RMSE	1.374	0.028	1.386	0.023 (+0.89%)	1.380	0.027 (+0.43%)
PDBbind-C	RMSE	2.154	0.290	2.203	0.273 (+2.29%)	2.236	0.349 (+3.83%)
PDBbind-R	RMSE	1.468	0.075	1.488	0.056 (+1.39%)	1.493	0.062 (+1.71%)
PCBA	PRC-AUC	0.271	0.011	0.269	0.012 (-0.87%)	0.267	0.014 (-1.47%)
MUV	PRC-AUC	0.0807	0.049	0.1080	0.0587 (+33.86%)	0.1180	0.0405 (+46.24%)
HIV	ROC-AUC	0.782	0.016	0.769	0.025 (-1.67%)	0.766	0.029 (-2.02%)
BACE	ROC-AUC	0.825	0.039	0.783	0.061 (-5.12%)	0.815	0.044 (-1.30%)
BBBP	ROC-AUC	0.915	0.037	0.913	0.038 (-0.24%)	0.913	0.041 (-0.22%)
Tox21	ROC-AUC	0.808	0.025	0.805	0.023 (-0.34%)	0.808	0.024 (+0.02%)
ToxCast	ROC-AUC	0.690	0.011	0.683	0.012 (-1.05%)	0.691	0.013 (+0.12%)
SIDER	ROC-AUC	0.606	0.029	0.589	0.026 (-2.75%)	0.595	0.030 (-1.77%)
ClinTox	ROC-AUC	0.874	0.041	0.872	0.041 (-0.28%)	0.879	0.054 (+0.56%)
ChEMBL	ROC-AUC	0.709	0.010	0.710	0.016 (+0.07%)	0.710	0.016 (+0.12%)

Table D.2: Message Type (Sca old Split).

Appendix E

Model Optimizations

E.1 Hyperparameter Optimization

E ect of performing Bayesian hyperparameter optimization on the depth, hidden size, number of fully connected layers, and dropout of our model. Optimization was done on random splits and then the optimized model was applied to both random and sca old splits.

(a) Regression Datasets (Random Split, lower = better).

(b) Classi cation Datasets (Random Split, higher = better).

(c) Regression Datasets (Sca old Split, lower = better).

(d) Classi cation Datasets (Sca old Split, higher = better).

Figure E-1: Hyperparameter Optimization.

Dataset	Metric	D-MPNN		D-MPNN Optimized	
QM7	MAE	67.657	3.647	61.524	3.337 (-9.07%)
QM8	MAE	0.0111	0.0001	0.0083	0.0002 (-25.23%)
QM9	MAE	3.705	0.537	2.116	0.023 (-42.90%)
ESOL	RMSE	0.691	0.080	0.667	0.072 (-3.43%)
FreeSolv	RMSE	1.133	0.176	1.031	0.204 (-9.02%)
Lipophilicity	RMSE	0.598	0.064	0.577	0.042 (-3.46%)
PDBbind-F	RMSE	1.325	0.027	1.299	0.028 (-1.98%)
PDBbind-C	RMSE	2.088	0.194	2.107	0.191 (+0.91%)
PDBbind-R	RMSE	1.411	0.060	1.377	0.062 (-2.43%)
PCBA	PRC-AUC	0.329	0.007	0.390	0.006 (+18.66%)
MUV	PRC-AUC	0.0865	0.0456	0.126	0.056 (+45.13%)
HIV	ROC-AUC	0.811	0.037	0.827	0.029 (+2.03%)
BACE	ROC-AUC	0.863	0.028	0.886	0.030 (+2.62%)
BBBP	ROC-AUC	0.920	0.017	0.927	0.017 (+0.77%)
Tox21	ROC-AUC	0.847	0.012	0.854	0.011 (+0.84%)
ToxCast	ROC-AUC	0.735	0.014	0.759	0.014 (+3.28%)
SIDER	ROC-AUC	0.639	0.024	0.647	0.016 (+1.24%)
ClinTox	ROC-AUC	0.892	0.043	0.903	0.032 (+1.21%)
ChEMBL	ROC-AUC	0.745	0.031	0.767	0.035 (+2.92%)

Table E.1: Hyperparameter Optimization (Random Split).

Dataset	Metric	D-MPNN		D-MPNN Optimized	
QM7	MAE	102.302	7.532	90.696	9.463 (-11.34%)
QM8	MAE	0.0148	0.0019	0.0132	0.0021 (-10.69%)
QM9	MAE	5.177	1.848	2.599	0.414 (-49.81%)
ESOL	RMSE	0.862	0.094	0.880	0.121 (+2.12%)
FreeSolv	RMSE	1.932	0.412	2.151	0.486 (+11.33%)
Lipophilicity	RMSE	0.656	0.047	0.646	0.041 (-1.55%)
PDBbind-F	RMSE	1.374	0.028	1.353	0.022 (-1.54%)
PDBbind-C	RMSE	2.154	0.290	2.127	0.277 (-1.24%)
PDBbind-R	RMSE	1.468	0.075	1.463	0.078 (-0.30%)
PCBA	PRC-AUC	0.271	0.011	0.295	0.005 (+9.00%)
MUV	PRC-AUC	0.0807	0.0494	0.0791	0.0389 (-1.99%)
HIV	ROC-AUC	0.782	0.016	0.789	0.024 (+0.85%)
BACE	ROC-AUC	0.825	0.039	0.841	0.053 (+1.89%)
BBBP	ROC-AUC	0.915	0.037	0.914	0.030 (-0.12%)
Tox21	ROC-AUC	0.808	0.025	0.818	0.023 (+1.22%)
ToxCast	ROC-AUC	0.690	0.011	0.705	0.011 (+2.10%)
SIDER	ROC-AUC	0.606	0.029	0.622	0.023 (+2.70%)
ClinTox	ROC-AUC	0.874	0.041	0.897	0.029 (+2.59%)
ChEMBL	ROC-AUC	0.709	0.010	0.757	0.031 (+6.76%)

Table E.2: Hyperparameter Optimization (Sca old Split).

Dataset	Depth	Dropout	# FFN Layers	Hidden Size	# Parameters
QM7	5	0.1	3	700	1,477,601
QM8	5	0	3	2000	9,256,212
QM9	6	0	3	2400	12,918,312
ESOL	6	0.2	2	900	2,143,501
FreeSolv	6	0	3	2100	10,131,001
Lipophilicity	5	0.2	3	1200	3,668,101
PDBbind-F	6	0	2	300	354,901
PDBbind-C	6	0	3	300	445,201
PDBbind-R	4	0.1	2	1800	7,526,401
PCBA	4	0.2	2	2400	12,953,228
MUV	3	0	2	300	359,717
HIV	6	0.3	3	2300	12,007,201
BACE	6	0.1	2	600	1,069,201
BBBP	5	0.15	1	1000	2,282,001
Tox21	6	0.15	2	700	1,390,612
ToxCast	6	0.1	2	600	1,254,617
SIDER	5	0.05	1	1400	4,351,227
ClinTox	6	0.15	3	2200	11,049,402
ChEMBL	6	0.05	2	700	1,781,310

Table E.3: Optimal Hyperparameter Settings.

E.2 RDKit Features

E ect of adding RDKit features to our optimized D-MPNN.

(a) Regression Datasets (Random Split, lower = better).

(b) Classi cation Datasets (Random Split, lower = better).

(c) Regression Datasets (Sca old Split, lower = better).

(d) Classi cation Datasets (Sca old Split, higher = better).

Figure E-2: RDKit Features.

Dataset	Metric	D-MPNN Optimized	D-MPNN Optimized + Features
QM7	MAE	61.524 3.337	60.106 2.250 (-2.30%)
QM8	MAE	0.0083 0.0002	0.0084 0.0002 (+1.46%)
QM9	MAE	2.116 0.023	1.992 0.017 (-5.85%)
ESOL	RMSE	0.667 0.072	0.620 0.041 (-7.05%)
FreeSolv	RMSE	1.031 0.204	1.027 0.211 (-0.39%)
Lipophilicity	RMSE	0.577 0.042	0.557 0.058 (-3.47%)
PDBbind-F	RMSE	1.299 0.028	1.287 0.027 (-0.92%)
PDBbind-C	RMSE	2.107 0.191	1.933 0.143 (-8.27%)
PDBbind-R	RMSE	1.377 0.062	1.325 0.060 (-3.74%)
PCBA	PRC-AUC	0.390 0.006	0.375 0.005 (-3.79%)
MUV	PRC-AUC	0.12560 0.056	0.1092 0.0483 (-13.03%)
HIV	ROC-AUC	0.827 0.029	0.803 0.053 (-2.97%)
BACE	ROC-AUC	0.886 0.030	0.889 0.024 (+0.34%)
BBBP	ROC-AUC	0.927 0.017	0.931 0.016 (+0.43%)
Tox21	ROC-AUC	0.854 0.011	0.862 0.008 (+0.94%)
ToxCast	ROC-AUC	0.759 0.014	0.769 0.015 (+1.32%)
SIDER	ROC-AUC	0.647 0.016	0.658 0.018 (+1.70%)
ClinTox	ROC-AUC	0.903 0.032	0.900 0.031 (-0.33%)
ChEMBL	ROC-AUC	0.767 0.035	0.761 0.041 (-0.75%)

Table E.4: RDKit Features (Random Split).

Dataset	Metric	D-MPNN Optimized	D-MPNN Optimized + Features
QM7	MAE	90.696 9.463	88.136 11.494 (-2.82%)
QM8	MAE	0.0132 0.0021	0.0133 0.0022 (+0.64%)
QM9	MAE	2.599 0.414	2.465 0.375 (-5.13%)
ESOL	RMSE	0.880 0.121	0.825 0.061 (-6.25%)
FreeSolv	RMSE	2.151 0.486	2.193 0.634 (+1.95%)
Lipophilicity	RMSE	0.646 0.041	0.624 0.049 (-3.41%)
PDBbind-F	RMSE	1.353 0.022	1.347 0.028 (-0.44%)
PDBbind-C	RMSE	2.127 0.277	2.047 0.244 (-3.74%)
PDBbind-R	RMSE	1.463 0.078	1.413 0.057 (-3.47%)
PCBA	PRC-AUC	0.295 0.005	0.290 0.007 (-1.70%)
MUV	PRC-AUC	0.0791 0.0389	0.0564 0.0346 (-28.65%)
HIV	ROC-AUC	0.789 0.024	0.799 0.020 (+1.28%)
BACE	ROC-AUC	0.841 0.053	0.852 0.053 (+1.31%)
BBBP	ROC-AUC	0.914 0.030	0.919 0.027 (+0.55%)
Tox21	ROC-AUC	0.818 0.023	0.826 0.019 (+0.98%)
ToxCast	ROC-AUC	0.705 0.011	0.718 0.005 (+1.84%)
SIDER	ROC-AUC	0.622 0.023	0.632 0.027 (+1.61%)
ClinTox	ROC-AUC	0.897 0.029	0.890 0.040 (-0.78%)
ChEMBL	ROC-AUC	0.757 0.031	0.753 0.016 (-0.57%)

Table E.5: RDKit Features (Sca old Split).

Dataset	Features? (Random)	Features? (Sca old)
QM7	Yes	Yes
QM8	No	No
QM9	Yes	Yes
ESOL	Yes	Yes
FreeSolv	Yes	No
Lipophilicity	Yes	Yes
PDBbind-F	Yes	Yes
PDBbind-C	Yes	Yes
PDBbind-R	Yes	Yes
PCBA	No	No
MUV	No	No
HIV	No	Yes
BACE	Yes	Yes
BBBP	Yes	Yes
Tox21	Yes	Yes
ToxCast	Yes	Yes
SIDER	Yes	Yes
ClinTox	No	No
ChEMBL	No	No

Table E.6: Whether RDKit features improve D-MPNN performance on random or sca old splits.

E.3 Ensembling

Bene t of ensembling ve models instead of a single model. All results are using our best model settings (i.e. optimized hyperparameters and RDKit features, if they improved performance in the single model setting).

(a) Regression Datasets (Random Split, lower = better).

(b) Classi cation Datasets (Random Split, higher = better).

(c) Regression Datasets (Sca old Split, lower = better).

(d) Classi cation Datasets (Sca old Split, higher = better).

Figure E-3: Ensembling.

Dataset	Metric	D-MPNN Best		D-MPNN Best Ensemble	
QM7	MAE	60.106	2.250	57.952	2.674 (-3.58%)
QM8	MAE	0.0083	0.0002	0.0077	0.0001 (-6.69%)
QM9	MAE	1.992	0.017	1.585	0.011 (-20.43%)
ESOL	RMSE	0.620	0.041	0.607	0.051 (-2.12%)
FreeSolv	RMSE	1.027	0.211	0.999	0.198 (-2.69%)
Lipophilicity	RMSE	0.557	0.058	0.548	0.050 (-1.69%)
PDBbind-F	RMSE	1.287	0.027	1.264	0.026 (-1.78%)
PDBbind-C	RMSE	1.933	0.143	1.910	0.137 (-1.21%)
PDBbind-R	RMSE	1.325	0.060	1.318	0.063 (-0.57%)
PCBA	PRC-AUC	0.390	0.006	0.411	0.008 (+5.31%)
MUV	PRC-AUC	0.1256	0.0565	0.1356	0.0660 (+7.98%)
HIV	ROC-AUC	0.827	0.029	0.843	0.030 (+1.89%)
BACE	ROC-AUC	0.889	0.024	0.893	0.019 (+0.47%)
BBBP	ROC-AUC	0.931	0.016	0.930	0.017 (-0.14%)
Tox21	ROC-AUC	0.862	0.008	0.868	0.008 (+0.66%)
ToxCast	ROC-AUC	0.769	0.015	0.777	0.014 (+1.04%)
SIDER	ROC-AUC	0.658	0.018	0.660	0.018 (+0.30%)
ClinTox	ROC-AUC	0.903	0.032	0.917	0.023 (+1.54%)
ChEMBL	ROC-AUC	0.767	0.035	0.778	0.026 (+1.48%)

Table E.7: Ensembling (Random Split).

Dataset	Metric	D-MPNN	Best	D-MPNN Best Ensemble	
QM7	MAE	88.136	11.494	85.173	9.300 (-3.36%)
QM8	MAE	0.0132	0.0021	0.0125	0.0021 (-5.56%)
QM9	MAE	2.465	0.375	2.043	0.391 (-17.14%)
ESOL	RMSE	0.825	0.061	0.801	0.057 (-2.94%)
FreeSolv	RMSE	2.151	0.486	2.122	0.460 (-1.34%)
Lipophilicity	RMSE	0.624	0.049	0.611	0.045 (-2.12%)
PDBbind-F	RMSE	1.347	0.028	1.319	0.024 (-2.11%)
PDBbind-C	RMSE	2.047	0.244	1.984	0.254 (-3.09%)
PDBbind-R	RMSE	1.413	0.057	1.387	0.057 (-1.82%)
PCBA	PRC-AUC	0.295	0.007	0.317	0.003 (+7.43%)
MUV	PRC-AUC	0.0791	0.0346	0.0913	0.0421 (+15.44%)
HIV	ROC-AUC	0.799	0.024	0.809	0.024 (+1.33%)
BACE	ROC-AUC	0.852	0.053	0.860	0.048 (+0.94%)
BBBP	ROC-AUC	0.919	0.030	0.925	0.023 (+0.60%)
Tox21	ROC-AUC	0.826	0.023	0.832	0.018 (+0.75%)
ToxCast	ROC-AUC	0.718	0.011	0.725	0.009 (+1.01%)
SIDER	ROC-AUC	0.632	0.023	0.633	0.029 (+0.11%)
ClinTox	ROC-AUC	0.897	0.040	0.913	0.024 (+1.74%)
ChEMBL	ROC-AUC	0.757	0.016	0.774	0.024 (+2.21%)

Table E.8: Ensembling (Sca old Split).

Appendix F

E ect of Data Size

Figure F-1: E ect of Data Size on ChEMBL (higher = better).

Min # of Compounds	[32] FFN	D-MPNN Best
0	0.741	0.718 (-3.10%)
1,000	0.801	0.796 (-0.62%)
5,000	0.836	0.842 (+0.72%)
10,000	0.844	0.851 (+0.83%)

Table F.1: E ect of Data Size on ChEMBL. All numbers are ROC-AUC.

Appendix G

RDKit-Calculated Features

I used the following list of 200 RDKit functions to calculate the molecule-level features used by the model.

BalabanJ	BertzCT	Chi0
Chi0n	Chi0v	Chi1
Chi1n	Chi1v	Chi2n
Chi2v	Chi3n	Chi3v
Chi4n	Chi4v	EState_VSA1
EState_VSA10	EState_VSA11	EState_VSA2
EState_VSA3	EState_VSA4	EState_VSA5
EState_VSA6	EState_VSA7	EState_VSA8
EState_VSA9	ExactMolWt	FpDensityMorgan1
FpDensityMorgan2	FpDensityMorgan3	FractionCSP3
HallKierAlpha	HeavyAtomCount	HeavyAtomMolWt
Ipc	Kappa1	Kappa2
Kappa3	LabuteASA	MaxAbsEStateIndex
MaxAbsPartialCharge	MaxEStateIndex	MaxPartialCharge
MinAbsEStateIndex	MinAbsPartialCharge	MinEStateIndex
MinPartialCharge	MolLogP	MolMR
MolWt	NHOHCount	NOCount

NumAliphaticCarbocycles	NumAliphaticHeterocycles	NumAliphaticRings
NumAromaticCarbocycles	NumAromaticHeterocycles	NumAromaticRings
NumHAcceptors	NumHDonors	NumHeteroatoms
NumRadicalElectrons	NumRotatableBonds	NumSaturatedCarbocycles
NumSaturatedHeterocycles	NumSaturatedRings	NumValenceElectrons
PEOE_VSA1	PEOE_VSA10	PEOE_VSA11
PEOE_VSA12	PEOE_VSA13	PEOE_VSA14
PEOE_VSA2	PEOE_VSA3	PEOE_VSA4
PEOE_VSA5	PEOE_VSA6	PEOE_VSA7
PEOE_VSA8	PEOE_VSA9	RingCount
SMR_VSA1	SMR_VSA10	SMR_VSA2
SMR_VSA3	SMR_VSA4	SMR_VSA5
SMR_VSA6	SMR_VSA7	SMR_VSA8
SMR_VSA9	SlogP_VSA1	SlogP_VSA10
SlogP_VSA11	SlogP_VSA12	SlogP_VSA2
SlogP_VSA3	SlogP_VSA4	SlogP_VSA5
SlogP_VSA6	SlogP_VSA7	SlogP_VSA8
SlogP_VSA9	TPSA	VSA_EState1
VSA_EState10	VSA_EState2	VSA_EState3
VSA_EState4	VSA_EState5	VSA_EState6
VSA_EState7	VSA_EState8	VSA_EState9
fr_Al_COO	fr_Al_OH	fr_Al_OH_noTert
fr_ArN	fr_Ar_COO	fr_Ar_N
fr_Ar_NH	fr_Ar_OH	fr_COO
fr_COO2	fr_C_O	fr_C_O_noCOO
fr_C_S	fr_HOCCN	fr_Imine
fr_NH0	fr_NH1	fr_NH2
fr_N_O	fr_Ndealkylation1	fr_Ndealkylation2
fr_Nhpyrrole	fr_SH	fr_aldehyde
fr_alkyl_carbamate	fr_alkyl_halide	fr_allylic_oxid

fr_amide	fr_amidine	fr_aniline
fr_aryl_methyl	fr_azide	fr_azo
fr_barbitur	fr_benzene	fr_benzodiazepine
fr_bicyclic	fr_diazo	fr_dihydropyridine
fr_epoxide	fr_ester	fr_ether
fr_furan	fr_guanido	fr_halogen
fr_hdrzine	fr_hdrzone	fr_imidazole
fr_imide	fr_isocyan	fr_isothiocyan
fr_ketone	fr_ketone_Topliss	fr_lactam
fr_lactone	fr_methoxy	fr_morpholine
fr_nitrile	fr_nitro	fr_nitro_arom
fr_nitro_arom_nonortho	fr_nitroso	fr_oxazole
fr_oxime	fr_para_hydroxylation	fr_phenol
fr_phenol_noOrthoHbond	fr_phos_acid	fr_phos_ester
fr_piperdine	fr_piperzine	fr_priamide
fr_prisulfonamd	fr_pyridine	fr_quatN
fr_sul de	fr_sulfonamd	fr_sulfone
fr_term_acetylene	fr_tetrazole	fr_thiazole
fr_thiocyan	fr_thiophene	fr_unbrch_alkane
fr_urea	qed	

Bibliography

[1] http://chemprop.csail.mit.edu .

[2] https://github.com/swansonk14/chemprop .

[3] https://github.com/bp-kelley/descriptastorus .

[4] https://github.com/yangkevin2/lsc_experiments .

[5] Peter Battaglia, Razvan Pascanu, Matthew Lai, and Danilo Jimenez Rezende. Interaction networks for learning about objects, relations and physics. Advances in Neural Information Processing Systems, pages 4502 4510, 2016.

[6] Leo Breiman. Random forests. Machine learning 45(1):5 32, 2001.

[7] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203, 2013.

[8] Dong-Sheng Cao, Qing-Song Xu, Qian-Nan Hu, and Yi-Zeng Liang. Chemopy: Freely available python package for computational biology and chemoinformatics. Bioinformatics, 29(8):1092 1094, 2013.

[9] Connor W. Coley, Regina Barzilay, William H. Green, Tommi S. Jaakkola, and Klavs F. Jensen. Convolutional embedding of attributed molecular graphs for physical property prediction. Journal of Chemical Information and Modeling, 57(8):1757 1772.

[10] Corinna Cortes and Vladimir Vapnik. Support vector machine. Machine Learning, 20(3):273 297, 1995.

[11] Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, pages 273 297, 1995.

[12] J. S. Cramer. Logit models from economics and other fields. 2003.

[13] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in Neural Information Processing Systems, pages 3844 3852, 2016.

[14] Thomas G Dietterich. Ensemble methods in machine learning. International Workshop on Multiple Classi er Systems, pages 1 15, 2000.

[15] Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. Reoptimization of mdl keys for use in drug discovery. Journal of Chemical Information and Computer Sciences, 42(6):1273 1280, 2002.

[16] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular ngerprints. Advances in Neural Information Processing Systems, pages 2224 2232, 2015.

[17] Felix A Faber, Luke Hutchison, Bing Huang, Justin Gilmer, Samuel S Schoenholz, George E Dahl, Oriol Vinyals, Steven Kearnes, Patrick F Riley, and O Anatole von Lilienfeld. Machine learning prediction errors better than dft accuracy. arXiv preprint arXiv:1702.05532, 2017.

[18] Evan N Feinberg, Debnil Sur, Zhenqin Wu, Brooke E Husic, Huanghao Mai, Yang Li, Saisai Sun, Jianyi Yang, Bharath Ramsundar, and Vijay S Pande. Potentialnet for molecular property prediction. ACS Central Science, 4(11):1520 1530, 2018.

[19] Jerome H Friedman. Greedy function approximation: A gradient boosting machine. Annals of Statistics, pages 1189 1232, 2001.

[20] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. Proceedings of the 34th International Conference on Machine Learning, 2017.

[21] John J Irwin and Brian K Shoichet. Zinc a free database of commercially available compounds for virtual screening. Journal of Chemical Information and Modeling, 45(1):177 182, 2005.

[22] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: Moving beyond ngerprints. Journal of Computer-Aided Molecular Design, 30(8):595 608, 2016.

[23] Sunghwan Kim, Paul A Thiessen, Evan E Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, and Benjamin A Shoemaker. Pubchem substance and compound databases. Nucleic Acids Research, 44(D1):D1202 D1213, 2015.

[24] Thomas N Kipf and Max Welling. Semi-supervised classi cation with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

[25] Risi Kondor, Hy Truong Son, Horace Pan, Brandon Anderson, and Shubhendu Trivedi. Covariant compositional networks for learning graphs. arXiv preprint arXiv:1801.02144, 2018.

[26] Greg Landrum. Rdkit: Open-source cheminformatics. 2006. https://rdkit.org/docs/index.html .

[27] Alpha A. Lee, Qingyi Yang, Asser Bassyouni, Christopher R. Butler, Xinjun Hou, Stephen Jenkinson, and David A. Price. Ligand biological activity predicted by cleaning positive and negative chemical correlations. Proceedings of the National Academy of Sciences, 2019.

[28] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493, 2015.

[29] Alessandro Lusci, Gianluca Pollastri, and Pierre Baldi. Deep architectures and deep learning in chemoinformatics: The prediction of aqueous solubility for drug-like molecules. Journal of Chemical Information and Modeling, 53(7):1563 1575, 2013.

[30] Junshui Ma, Robert P Sheridan, Andy Liaw, George E Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure activity relationships. Journal of Chemical Information and Modeling, 55(2):263 274, 2015.

[31] Andrea Mauri, Viviana Consonni, Manuela Pavan, and Roberto Todeschini. Dragon software: An easy approach to molecular descriptor calculations. Match, 56(2):237 248, 2006.

[32] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, Marvin Steijaert, Jörg K. Wegner, Hugo Ceulemans, Djork-Arné Clevert, and Sepp Hochreiter. Large-scale comparison of machine learning methods for drug target prediction on chembl. Chemical Science, 9:5441 5451, 2018.

[33] Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, and Tatsuya Takagi. Mordred: A molecular descriptor calculator. Journal of Cheminformatics, 10(1):4, 2018.

[34] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic di erentiation in pytorch. 31st Conference on Neural Information Processing Systems, 2017.

[35] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, and Vincent Dubourg. Scikit-learn: Machine learning in python. Journal of Machine Learning Research, 12(Oct):2825 2830, 2011.

[36] Bharath Ramsundar, Bowen Liu, Zhenqin Wu, Andreas Verras, Matthew Tudor, Robert P Sheridan, and Vijay Pande. Is multitask deep learning practical for pharma? Journal of Chemical Information and Modeling, 57(8):2068 2076, 2017.

[37] David Rogers and Mathew Hahn. Extended-connectivity ngerprints. Journal of chemical information and modeling, 50(5):742 754, 2010.

[38] Maureen A Sartor, George D Leikauf, and Mario Medvedovic. Lrpath: A logistic regression approach for identifying enriched biological groups in gene expression data. Bioinformatics, 25(2):211 217, 2008.

[39] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous- lter convolutional neural network for modeling quantum interactions. Advances in Neural Information Processing Systems, pages 991 1001, 2017.

[40] Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. Nature Communications, 8:13890, 2017.

[41] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. Proceedings of the IEEE, 104(1):148 175, Jan 2016.

[42] Justin S Smith, Olexandr Isayev, and Adrian E Roitberg. Ani-1: An extensible neural network potential with dft accuracy at force eld computational cost. Chemical Science, 8(4):3192 3203, 2017.

[43] S Joshua Swamidass, Chloé-Agathe Azencott, Ting-Wan Lin, Hugo Gramajo, Shiou-Chuan Tsai, and Pierre Baldi. In uence relevance voting: An accurate and interpretable virtual high throughput screening method. Journal of Chemical Information and Modeling, 49(4):756 766, 2009.

[44] S Joshua Swamidass, Jonathan Chen, Jocelyne Bruand, Peter Phung, Liva Ralaivola, and Pierre Baldi. Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. Bioinformatics, 21(suppl_1):i359 i368, 2005.

[45] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. Journal of Chemical Information and Computer Sciences, 28(1):31 36, 1988.

[46] Zhenqin Wu, Bharath Ramsundar, EvanÂ N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. Moleculenet: A benchmark for molecular machine learning. Chemical Science, 9(2):513â€“530, 2018.

[47] Kevin Yang. Are learned molecular representations ready for prime time? MIT Master's Thesis, 2019.

[48] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Tim Hopper, Brian Kelley, Miriam Mathea, Andrew Palmer, Volker Settels, Tommi Jaakkola, Klavs F. Jensen, and Regina Barzilay. Are learned molecular representations ready for prime time? submission to Journal for Chemical Information and Modeling, 2019.