This is an open access article published under a Creative Commons Non-Commercial No Derivative Works (CC-BY-NC-ND) Attribution License, which permits copying and redistribution of the article, and creation of adaptations, all for non-commercial purposes.

JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

Article

Cite This: J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

pubs.acs.org/jcim

Analyzing Learned Molecular Representations for Property Prediction

Kevin Yang,[*,†] Kyle Swanson,[*,†] Wengong Jin,[†] Connor Coley,[‡] Philipp Eiden,[¶] Hua Gao,[§] Angel Guzman-Perez,[§] Timothy Hopper,[§] Brian Kelley,[∥] Miriam Mathea,[¶] Andrew Palmer,[¶] Volker Settels,[¶] Tommi Jaakkola,[†] Klavs Jensen,[‡] and Regina Barzilay[†]

[†]Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, Massachusetts 02139, United States

[‡]Department of Chemical Engineering, MIT, Cambridge, Massachusetts 02139, United States

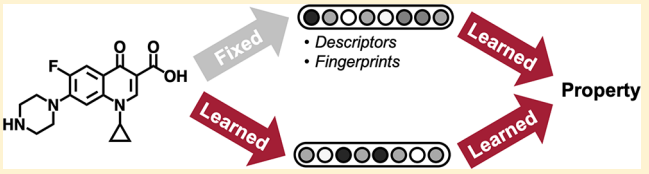
[¶]BASF SE, Ludwigshafen 67063, Germany

[§]Amgen Inc., Cambridge, Massachusetts 02141, United States

[∥]Novartis Institutes for BioMedical Research, Cambridge, Massachusetts 02139, United States

Supporting Information

ABSTRACT: Advancements in neural machinery have led to a wide range of algorithmic solutions for molecular property prediction. Two classes of models in particular have yielded promising results: neural networks applied to computed molecular fingerprints or expert-crafted descriptors and graph convolutional neural networks that construct a learned molecular representation by operating on the graph structure of the molecule. However, recent literature has yet to clearly determine which of these two methods is superior when generalizing to new chemical space. Furthermore, prior research has rarely examined these new models in industry research settings in comparison to existing employed models. In this paper, we benchmark models extensively on 19 public and 16 proprietary industrial data sets spanning a wide variety of chemical end points. In addition, we introduce a graph convolutional model that consistently matches or outperforms models using fixed molecular descriptors as well as previous graph neural architectures on both public and proprietary data sets. Our empirical findings indicate that while approaches based on these representations have yet to reach the level of experimental reproducibility, our proposed model nevertheless offers significant improvements over models currently used in industrial workflows.



INTRODUCTION

Molecular property prediction, one of the oldest cheminformatics tasks, has received new attention in light of recent advancements in deep neural networks. These architectures either operate over fixed molecular fingerprints common in traditional QSAR models, or they learn their own task-specific representations using graph convolutions.[1−11] Both approaches are reported to yield substantial performance gains, raising state-of-the-art accuracy in property prediction.

Despite these successes, many questions remain unanswered. The first question concerns the comparison between learned molecular representations and fingerprints or descriptors. Unfortunately, current published results on this topic do not provide a clear answer. Wu et al.[2] demonstrate that convolution-based models typically outperform fingerprint-based models, while experiments reported in Mayr et al.[12] report the opposite. Part of these discrepancies can be attributed to differences in evaluation setup, including the way data sets are constructed. This leads us to a broader question concerning current evaluation protocols and their capacity to measure the generalization power of a method when applied to a new chemical space, as is common in drug discovery. Unless special care is taken to replicate this distributional shift in evaluation, neural models may overfit the training data but still score highly on the test data. This is particularly true for convolutional models that can learn a poor molecular representation by memorizing the molecular scaffolds in the training data and thereby fail to generalize to new ones. Therefore, a meaningful evaluation of property prediction models needs to account explicitly for scaffold overlap between train and test data in light of generalization requirements.

In this paper, we aim to answer both of these questions by designing a comprehensive evaluation setup for assessing neural architectures. We also introduce an algorithm for property prediction that outperforms existing strong baselines across a range of data sets. The model has two distinctive features: (1) It operates over a hybrid representation that combines convolutions and descriptors. This design gives it flexibility in learning a task specific encoding, while providing a strong prior with fixed descriptors. (2) It learns to construct molecular encodings by using convolutions centered on bonds instead of atoms, thereby avoiding unnecessary loops during the message passing phase of the algorithm.

Received: March 18, 2019
Published: July 30, 2019

Downloaded via 50.201.114.62 on August 14, 2019 at 00:19:26 (UTC).
See https://pubs.acs.org/sharingguidelines for options on how to legitimately share published articles.

We extensively evaluate our model and other recently published neural architectures with over 850 experiments on 19 publicly available benchmarks from Wu et al.[2] and Mayr et al.[12] and on 16 proprietary data sets from Amgen, Novartis, and BASF (Badische Anilin und Soda Fabrik). Our goal is to assess whether the models' performance on the public data sets and their relative ranking are representative of their ranking on the proprietary data sets. We demonstrate that under a scaffold split of training and testing data, the relative ranking of the models is consistent across the two classes of data sets. We also show that a scaffold-based split of the training and testing data is a good approximation of the temporal split commonly used in industry in terms of the relevant metrics. By contrast, a purely random split is a poor approximation to a temporal split, confirming the findings of Sheridan.[13] To put the performance of current models in perspective, we report bounds on experimental error and show that there is still room for improving deep learning models to match the accuracy and reproducibility of screening results.

Building on the diversity of our benchmark data sets, we explore the impact of molecular representation with respect to the data set characteristics. We find that a hybrid representation yields higher performance and generalizes better than either convolution-based or fingerprint-based models. We also note that on small data sets (up to 1000 training molecules) fingerprint models can outperform learned representations, which are negatively impacted by data sparsity. Beyond molecular representation issues, we observe that hyperparameter selection plays a crucial role in model performance, consistent with prior work.[14] We show that Bayesian optimization yields a robust, automatic solution to this issue. The addition of ensembling further improves accuracy, again consistent with the literature.[15]

Our experiments show that our model achieves consistently strong out-of-the-box performance and even stronger optimized performance across a wide variety of public and proprietary data sets. Our model achieves comparable or better performance on 12 out of 19 public data sets and on all 16 proprietary data sets compared to all baseline models. Furthermore, no single baseline model is clearly superior across the remaining 7 public data sets, and the relative performance of the baseline models often varies from data set to data set, whereas our model is consistently strong across data sets. These results indicate that our model and learned molecular fingerprints, in general, are applicable and ready to be used as a powerful tool for chemists actively working on drug discovery.

BACKGROUND

Since the core element of our model is a graph encoder architecture, our work is closely related to previous work on graph encoders, such as those for social networks[6,16] or for chemistry applications.[1,7−9,17−24]

Common approaches to molecular property prediction today involve the application of well-known models like support vector machines[25] or random forests[26] to expert-engineered descriptors or molecular fingerprints, such as the Dragon descriptors[27] or Morgan (ECFP) fingerprints.[28] One direction of advancement is the use of domain expertise to improve the base feature representation of molecular descriptors[27,29−32] to drive better performance.[12] Additionally, many studies have leveraged explicit 3D atomic coordinates to improve performance further.[2,33−36]

The other main line of research is the optimization of the model architecture, whether the model is applied to descriptors or fingerprints[12,37] or is directly applied to SMILES[38] strings[12] or the underlying graph of the molecule.[1−11] Our model belongs to the last category of models, known as graph convolutional neural networks. In essence, such models learn their own expert feature representations directly from the data, and they have been shown to be very flexible and capable of capturing complex relationships given sufficient data.[2,4]

In a direction orthogonal to our own improvements, Ishiguro et al.[39] also make a strong improvement to graph neural networks. Liu et al.[40] also evaluate their model against private industry data sets, but we cannot compare against their method directly owing to data set differences.[40]

The property prediction models most similar to our own are encapsulated in the Message Passing Neural Network (MPNN) framework presented in Gilmer et al.[4] We build upon this basic framework by adopting a message-passing paradigm based on updating representations of directed bonds rather than atoms. Additionally, we further improve the model by combining computed molecule-level features with the molecular representation learned by the MPNN.

METHODS

We first summarize MPNNs in general using the terminology of Gilmer et al.,[4] and then we expand on the characteristics of Directed MPNN (D-MPNN)[19] used in this paper. (D-MPNN is originally called *structure2vec* in Dai et al.[19] In this paper, we refer to it as Directed MPNN to show it is a variant of the generic MPNN architecture.)

Message Passing Neural Networks. An MPNN is a model which operates on an undirected graph G with node (atom) features x_v and edge (bond) features e_{vw}. MPNNs operate in two phases: a *message passing phase*, which transmits information across the molecule to build a neural representation of the molecule, and a *readout phase*, which uses the final representation of the molecule to make predictions about the properties of interest.

More specifically, the message passing phase consists of T steps. On each step t, hidden states h_v^t and messages m_v^t associated with each vertex v are updated using message function M_t and vertex update 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. The readout phase then uses a readout function R to make a property prediction based on the final hidden states according to

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

The output $\hat{y}$ may be either a scalar or a vector, depending on whether the MPNN is designed to predict a single property or multiple properties (in a multitask setting).

During training, the network takes molecular graphs as input and outputs a prediction for each molecule. A loss function is computed based on the predicted outputs and the ground truth values, and the gradient of the loss is backpropagated through the readout phase and the message passing phase. The entire model is trained end-to-end.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

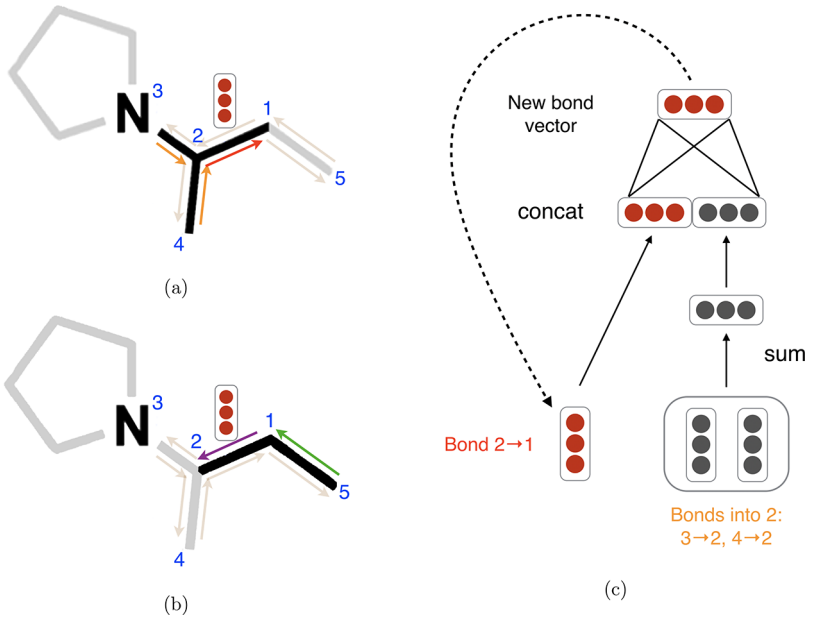


Figure 1. Illustration of bond-level message passing in our proposed D-MPNN. (a) Messages from the orange directed bonds are used to inform the update to the hidden state of the red directed bond. By contrast, in a traditional MPNN, messages are passed from atoms to atoms (for example, atoms 1, 3, and 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).

Directed MPNN. The main difference between the Directed MPNN (D-MPNN)[19] and the generic MPNN described above is the nature of the messages sent during the message passing phase. Rather than using messages associated with vertices (atoms), D-MPNN uses messages associated with directed edges (bonds). The motivation of this design is to prevent totters,[41] that is, to avoid messages being passed along any path of the form $v_1v_2\cdots v_n$ where $v_i = v_{i+2}$ for some i. Such excursions are likely to introduce noise into the graph representation. Using Figure 1 as an illustration, in D-MPNN, the message $1 \rightarrow 2$ will only be propagated to nodes 3 and 4 in the next iteration, whereas in the original MPNN it will be sent to node 1 as well, creating an unnecessary loop in the message passing trajectory. Compared to the atom based message passing approach, this message passing procedure is more similar to belief propagation in probabilistic graphical models.[42] We refer to Dai et al.[19] for further discussion about the connection between D-MPNN and belief propagation.

The D-MPNN works as follows. The D-MPNN operates on hidden states h^t_{vw} and messages m^t_{vw} instead of on node based hidden states h^t_v and messages m^t_v. Note that the direction of messages matters (i.e., h^t_{vw} and m^t_{vw} are distinct from h^t_{wv} and m^t_{wv}). The corresponding message passing update equations are thus

$$m^{t+1}_{vw} = \sum_{k\in\{N(v)\setminus w\}} M_t(x_v, x_k, h^t_{kv})$$

$$h^{t+1}_{vw} = U_t(h^t_{vw}, m^{t+1}_{vw})$$

Observe that message m^{t+1}_{vw} does not depend on its reverse message m^t_{wv} from the previous iteration. Prior to the first step of message passing, we initialize edge hidden states with

$$h^0_{vw} = \tau(W_i\ \mathrm{cat}(x_v, e_{vw}))$$

where $W_i \in \mathbb{R}^{h\times h_i}$ is a learned matrix, $\mathrm{cat}(x_v, e_{vw}) \in \mathbb{R}^{h_i}$ is the concatenation of the atom features x_v for atom v and the bond features e_{vw} for bond vw, and τ is the ReLU activation function.[43]

We choose to use relatively simple message passing functions M_t and edge update functions U_t. Specifically, we define $M_t(x_v, x_w, h^t_{vw}) = h^t_{vw}$, and we implement U_t with the same neural network on every step

$$U_t(h^t_{vw}, m^{t+1}_{vw}) = U(h^t_{vw}, m^{t+1}_{vw}) = \tau(h^0_{vw} + W_m m^{t+1}_{vw})$$

where $W_m \in \mathbb{R}^{h\times h}$ is a learned matrix with hidden size h. Note that the addition of h^0_{vw} on every step provides a skip connection to the original feature vector for that edge.

Finally, we return to an atom representation of the molecule by summing the incoming bond features according to

$$m_v = \sum_{k\in N(v)} h^T_{kv}$$

$$h_v = \tau(W_a\ \mathrm{cat}(x_v, m_v))$$

where $W_a \in \mathbb{R}^{h\times h}$ is a learned matrix.

Altogether, the D-MPNN message passing phase operates according to

$$h^0_{vw} = \tau(W_i\ \mathrm{cat}(x_v, e_{vw}))$$

followed by

$$m^{t+1}_{vw} = \sum_{k\in\{N(v)\setminus w\}} h^t_{kv}$$

$$h^{t+1}_{vw} = \tau(h^0_{vw} + W_m m^{t+1}_{vw})$$

for $t \in \{1, \ldots, T\}$, followed by

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

$$m_v = \sum_{w \in N(v)} h_{vw}^T$$

$$h_v = \tau(W_a \, \mathrm{cat}(x_v, m_v))$$

The readout phase of the D-MPNN is the same as the readout phase of a generic MPNN. In our implementation of the readout function R, we first sum the atom hidden states to obtain a feature vector for the molecule

$$h = \sum_{v \in G} h_v$$

Finally, we generate property predictions $\hat{y} = f(h)$ where $f(\cdot)$ is a feed-forward neural network.

Initial Featurization. Our model's initial atom and bond features are listed in Tables 1 and 2, respectively. The D-MPNN's initial node features x_v are simply the atom features for that node, while the D-MPNN's initial edge features e_{vw} are the bond features for bond vw. All features are computed using the open-source package RDKit.[44]

Table 1. Atom Features[a]

feature	description	size
atom type	type of atom (ex. C, N, O), by atomic number	100
# bonds	number of bonds the atom is involved in	6
formal charge	integer electronic charge assigned to atom	5
chirality	unspecified, tetrahedral CW/CCW, or other	4
# Hs	number of bonded hydrogen atoms	5
hybridization	sp, sp2, sp3, sp3d, or sp3d2	5
aromaticity	whether this atom is part of an aromatic system	1
atomic mass	mass of the atom, divided by 100	1

[a]All features are one-hot encodings except for atomic mass, which is a real number scaled to be on the same order of magnitude.

Table 2. Bond Features[a]

feature	description	size
bond type	single, double, triple, or aromatic	4
conjugated	whether the bond is conjugated	1
in ring	whether the bond is part of a ring	1
stereo	none, any, E/Z or cis/trans	6

[a]All features are one-hot encodings.

D-MPNN with Features. Next, we discuss further extensions and optimizations to improve performance. 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 data sets 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 the molecular graph, i.e., $T < diam(G)$, 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, we introduce 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 using the distributions available in the scikit-learn package,[45] a sample of which can be seen in Figure 2. One could do a similar normalization using publicly available databases such as ZINC[46] and PubChem.[47] 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 data sets.[48] 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 2. As such, since the sample data was primarily used for chemical screening against biological targets, the distribution used here may not accurately reflect the distribution of reagents used for chemical synthesis. For the full list of calculated features, please refer to the Supporting Information.

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

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

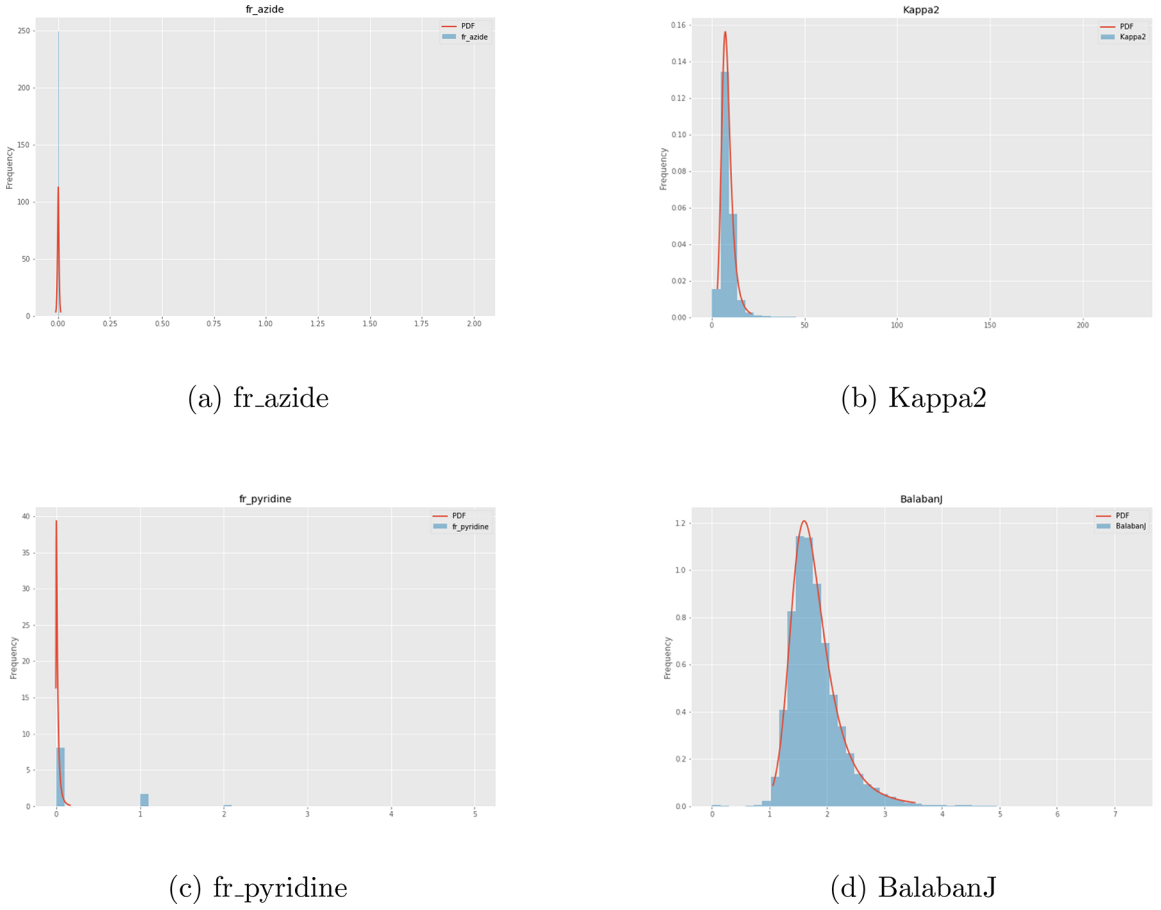
This is a very general method of incorporating external information and can be used with any MPNN and any computed features or descriptors.

Hyperparameter Optimization. The performance of MPNNs, like most neural networks, can depend greatly on the settings of the various model hyperparameters, such as the hidden size of the neural network layers. Thus, to maximize performance, we perform hyperparameter optimization via Bayesian Optimization[14] using the Hyperopt[49] Python package. We specifically optimize our model's depth (number of message-passing steps), hidden size (size of bond message vectors), number of feed-forward network layers, and dropout probability.

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.[15] We apply 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 we make to prior work use a single D-MPNN model for a fair comparison. However, we

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) fr_azide (b) Kappa2

(c) fr_pyridine (d) BalabanJ

Figure 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.

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

Implementation. We implement our model using the PyTorch[50] deep learning framework. All code for the D-MPNN and its variants is available in our GitHub repository.[51] Code for computing and using the RDKit feature CDFs is available in the Descriptastorus package.[52] Additionally, a Web demonstration of our model's predictive capability on public data sets is available online.[53]

EXPERIMENTS

Data. We test our model on 19 publicly available data sets from Wu et al.[2] and Mayr et al.[12] These data sets range in size from less than 200 molecules to over 450,000 molecules. They include a wide range of regression and classification targets spanning quantum mechanics, physical chemistry, biophysics, and physiology. Detailed descriptions are provided in Table 3.

Summary statistics for all the data sets are provided in Table 4, and further details on the data sets are available in Wu et al.,[2] with the exception of the ChEMBL data set which is described in Mayr et al.[12] Additional information on the class balance of the classification data sets is provided in the Supporting Information. Although most classification data sets are reasonably balanced, the MUV data set is particularly unbalanced, with only 0.2% of molecules classified as positive. This makes our model unstable, leading to the wide variation in performance on this data set in the subsequent sections.

Table 3. Descriptions of the Public Data Sets Used in This Paper

data set	category	description
QM7, QM8, QM9	quantum mechanics	computer-generated quantum mechanical properties
ESOL	physical chemistry	water solubility
FreeSolv	physical chemistry	hydration free energy in water
Lipophilicity	physical chemistry	octanol/water distribution coefficients
PDBbind	biophysics	protein binding affinity
PCBA	biophysics	assorted biological assays
MUV	biophysics	assorted biological assays
HIV	biophysics	inhibition of HIV replication
BACE	biophysics	inhibition of human β-secretase 1
BBBP	physiology	ability to penetrate the blood-brain barrier
Tox21	physiology	toxicity
ToxCast	physiology	toxicity
SIDER	physiology	side effects of drugs
ClinTox	physiology	toxicity
ChEMBL	physiology	biological assays

It is worth noting that for some data sets, the number of compounds in Table 4 does not precisely match the numbers from Wu et al.[2] This is because Wu et al.[2] included duplicate molecules in that count while we count the unique number of molecules. Additionally, we left out one or two molecules which

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

Table 4. Summary Statistics of the Public Data Sets Used in This Paper[a]

data set	no. of tasks	task type	no. of compounds	metric
QM7	1	regression	6,830	MAE
QM8	12	regression	21,786	MAE
QM9	12	regression	133,885	MAE
ESOL	1	regression	1,128	RMSE
FreeSolv	1	regression	642	RMSE
Lipophilicity	1	regression	4,200	RMSE
PDBbind-F	1	regression	9,880	RMSE
PDBbind-C	1	regression	168	RMSE
PDBbind-R	1	regression	3,040	RMSE
PCBA	128	classification	437,929	PRC-AUC
MUV	17	classification	93,087	PRC-AUC
HIV	1	classification	41,127	ROC-AUC
BACE	1	classification	1,513	ROC-AUC
BBBP	1	classification	2,039	ROC-AUC
Tox21	12	classification	7,831	ROC-AUC
ToxCast	617	classification	8,576	ROC-AUC
SIDER	27	classification	1,427	ROC-AUC
ClinTox	2	classification	1,478	ROC-AUC
ChEMBL	1310	classification	456,331	ROC-AUC

[a]Note: PDBbind-F, PDBbind-C, and PDBbind-R refer to the full, core, and refined PDBbind data sets from Wu et al.[2]

could not be processed by RDKit.[44] However, the impact of removing these molecules is negligible on overall model performance. Furthermore, we have fewer molecules in QM7 because we used SMILES strings generated by Wu et al.[2] from the original 3D coordinates in the data set, but the SMILES conversion process failed for ∼300 molecules. For this reason, we do not directly compare our model's performance on QM7 to the QM7 performance numbers reported by Wu et al.[2]

Experimental Procedure. *Cross-Validation and Hyperparameter Optimization.* Since many of the data sets are very small (2000 molecules or fewer), we use a cross-validation approach to decrease noise in the results both while optimizing the hyperparameters and while determining final performance numbers. For consistency, we maintain the same approach for all of our data sets. Specifically, for each data set, we use 20 iterations of Bayesian optimization on 10 randomly seeded 80:10:10 data splits to determine the best hyperparameters, selecting hyperparameters based on validation set performance. We then evaluate the model by retraining using the optimal hyperparameters and checking performance on the test set. Due to computational cost, we only use 3 splits for HIV, QM9, MUV, PCBA, and ChEMBL. When we run the best model from Mayr et al.[12] for comparative purposes, we optimize their model's hyperparameters with the same splits, using their original hyperparameter optimization script.

Split Type. We evaluate all models on random and scaffold-based splits as well as on the original splits from Wu et al.[2] and Mayr et al.[12] The one exception is the model of Mayr et al.,[12] which we only ran on scaffold-based splits, due to the large computational cost of optimizing their model. Results on scaffold-based splits are reported below, while results on random splits are presented in the Supporting Information.

Our scaffold split is similar to that of Wu et al.[2] Molecules are partitioned into bins based on their Murcko scaffold calculated by RDKit.[44] 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. As this latter process involves randomly placing scaffolds into bins, we are able to generate several different scaffold splits for evaluation.

None of our splits on classification data sets are stratified; we do not enforce class balance. Compared to random splits, the scaffold splits have more class imbalance on average but are not excessively imbalanced; we analyze this class balance quantitatively in the Additional Data Set Statistics section of the Supporting Information.

Compared to a random split, a scaffold split is a more challenging and realistic evaluation setting as shown in Figures 11 and 13. This allows us to use a scaffold split as a proxy for the chronological split present in real-world property prediction data, where one trains a model on past data to make predictions on future data, although chronological splits are still preferred when available. However, as chronological information is not available for most public data sets, we use a scaffold-based split for all evaluations except for our direct comparison with the MoleculeNet models from Wu et al.,[2] for which we use their original data splits.

Baselines. We compare our model to the following baselines:

- The best model for each data set from MoleculeNet by Wu et al.[2]
- The best model from Mayr et al.,[12] 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 our D-MPNN uses during its readout phase.
- FFN on count-based Morgan fingerprints.
- FFN on RDKit-calculated descriptors.

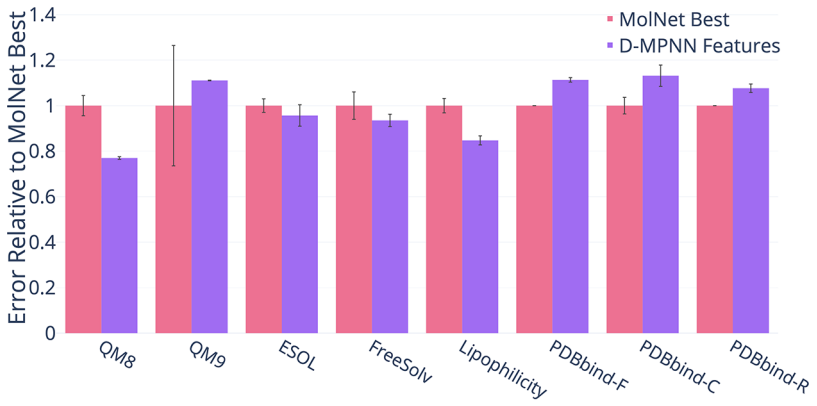
The models in MoleculeNet by Wu et al.[2] include MPNN,[4] Weave,[3] GraphConv, kernel ridge regression, gradient boosting,[54] random forest,[26] logistic regression,[55] directed acyclic graph models,[56] support vector machines,[57] Deep Tensor Neural Networks,[10] multitask networks,[58] bypass networks,[59] influence relevance voting,[60] and/or ANI-1,[61] depending on the data set. Full details can be found in Wu et al.[2] For the feed-forward network model from Mayr et al.,[12] we modified the authors' original code with their guidance in order to run their code on all of the data sets, not just on the ChEMBL data set they experimented with. We tuned learning rates and hidden dimensions in addition to the extensive hyperparameter search already present in their code.

RESULTS AND DISCUSSION

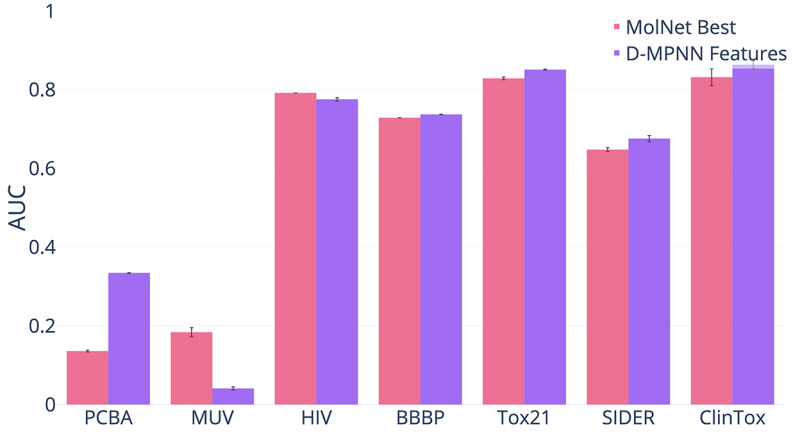
In the following sections, we analyze the performance of our model on both public and proprietary data sets. Specifically, we aim to answer the following questions:

1. How does our model perform on both public and proprietary data sets compared to public benchmarks, and how close are we to the upper bound on performance represented by experimental reproducibility?
2. How should we be splitting our data, and how does the method of splitting affect our evaluation of the model's generalization performance?
3. What are the key elements of our model, and how can we maximize its performance?

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 3. Comparison of our D-MPNN with features to the best models from Wu et al.[2]

In the following sections, all results using root-mean-square error (RMSE) or mean absolute error (MAE) are displayed as plots showing change relative to a baseline model rather than showing absolute performance numbers. This is because the scale of the errors can differ drastically between data sets. All results using R^2, area under the receiver operating characteristic curve (ROC-AUC), or area under the precision recall curve (PRC-AUC) are displayed as plots showing the actual values. For RMSE and MAE, lower is better, while for R^2, ROC-AUC, and PRC-AUC, higher is better. Table 4 indicates the metric used for each data set. Tables showing the exact performance numbers for all experiments can be found in the Supporting Information. 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.

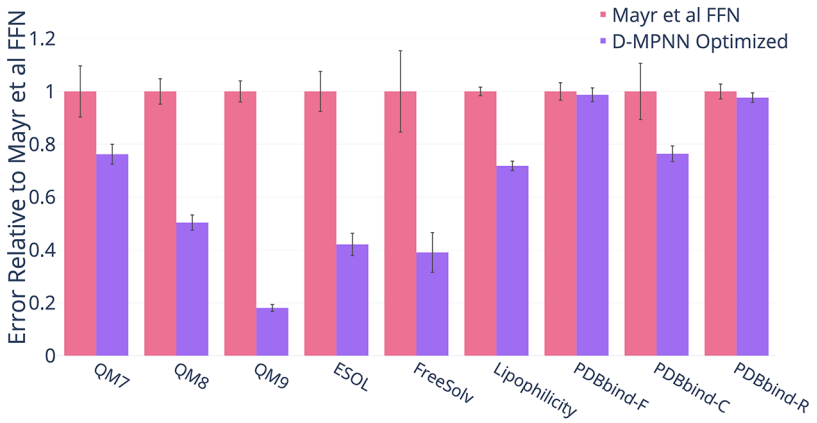
We evaluate statistical significance using two statistical tests: a one-sided Wilcoxon signed-rank test and a one-sided Welch's t test. While the Wilcoxon test is stronger, as it is a paired test comparing performance molecule-by-molecule, it requires knowing per-molecule predictions, which we do not have easy access to for the models from MoleculeNet[2] and Mayr et al.[12] Furthermore, comparisons between data split types inherently involve comparing performance on different test molecules, meaning a per-molecule test is not possible. Therefore, for these comparisons we use the weaker Welch's t test, and for all other comparisons we use the Wilcoxon test. When using the Wilcoxon test for regression data sets, we directly compare test errors molecule-by-molecule. For the classification data sets, we divide all the test molecules into 30 equal parts, compute AUC on each part, and then use the Wilcoxon test on these AUC values. This subdivision of the test molecules into 30 parts gives the Wilcoxon test more strength than evaluating directly on the original 3 or 10 test cross-validation folds while still keeping each part large enough to result in a meaningful AUC computation. We define statistical significance as p-value less than 0.05.

Additionally, note that in all figures and tables, "D-MPNN" refers to the base D-MPNN model, "D-MPNN Features" refers to the D-MPNN with RDKit features, "D-MPNN Optimized" refers to the D-MPNN with RDKit features and optimized hyperparameters, and "D-MPNN Ensemble" refers to an ensemble of five D-MPNNs with RDKit features and optimized hyperparameters.

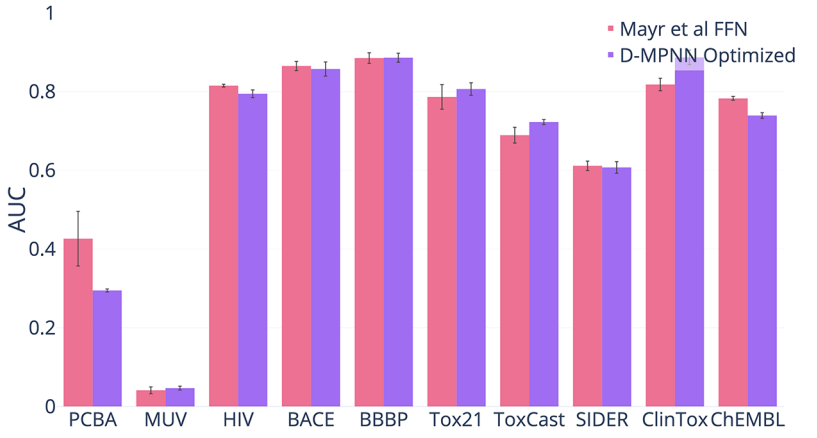
Comparison to Baselines. After optimizing our model, we compare our best single (nonensembled) model on each data set against models from prior work.

Comparison to MoleculeNet. We first compare our D-MPNN to the best model from MoleculeNet[2,62] on the same data sets and splits on which Wu et al.[2] evaluate their models. We were unable to reproduce their original data splits on BACE, ToxCast, and QM7, but we have evaluated our model against their original splits on all of the other data sets. The splits are a mix of random, scaffold, and time splits, as indicated in Figure 3.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 4. Comparison of our best single model (i.e., optimized hyperparameters and RDKit features) to the model from Mayr et al.

Overall on the 10 data sets where the MoleculeNet models only use 2D information, i.e., all data sets except the QM and PDBbind data sets, our D-MPNN is significantly better than the best MoleculeNet models on 5 data sets, is not significantly different on 3 data sets, and is significantly worse on 2 data sets. This indicates that D-MPNN tends to outperform even the best MoleculeNet models, with the added benefit that the D-MPNN model architecture is the same for every data set while the best MoleculeNet model architecture differs between data sets.

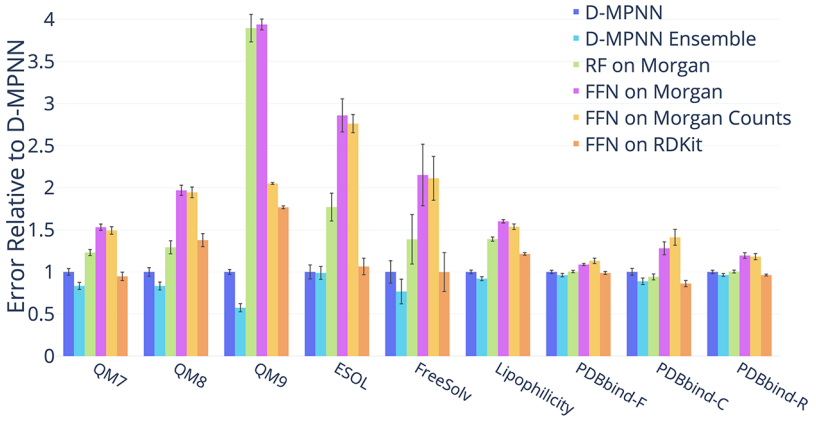
Furthermore, we note that there are two cases in which our D-MPNN may underperform. The first is the MUV data set, which is large but extremely imbalanced; only 0.2% of samples are labeled as positives. Wu et al.[2] also encountered great difficulty with this extreme class imbalance when experimenting with the MUV data set; all other data sets we experiment on contain at least 1% positives (see the Supporting Information for full class balance information). The second exception is when there is auxiliary 3D information available, as in the three variants of the PDBbind data set and in QM9. The current iteration of our D-MPNN does not use 3D coordinate information, and we leave this extension to future work. Thus, it is unsurprising that our D-MPNN model underperforms models using 3D information on a protein binding affinity prediction task such as PDBbind, where 3D structure is key. Nevertheless, our D-MPNN model outperforms the best graph-based method in MoleculeNet on PDBbind and QM9. Moreover, we note that on another data set that provides 3D coordinate information, QM8, our model outperforms the best model in MoleculeNet with or without 3D coordinates.

Comparison to Mayr et al.[12] In addition, we compare D-MPNN to the baseline from Mayr et al.[12] in Figure 4. We reproduced the features from their best model on each data set using their scripts or equivalent packages.[63] We then ran their code and hyperparameter optimization directly on the classification data sets, and we modified their code to run on regression data sets with the authors' guidance.[63] On most classification data sets, we obtain a similar performance to Mayr et al.[12] On regression data sets, the baseline from Mayr et al.[12] performs poorly in comparison, despite extensive tuning. We 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 our 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; this may be the source of some numerical instability.

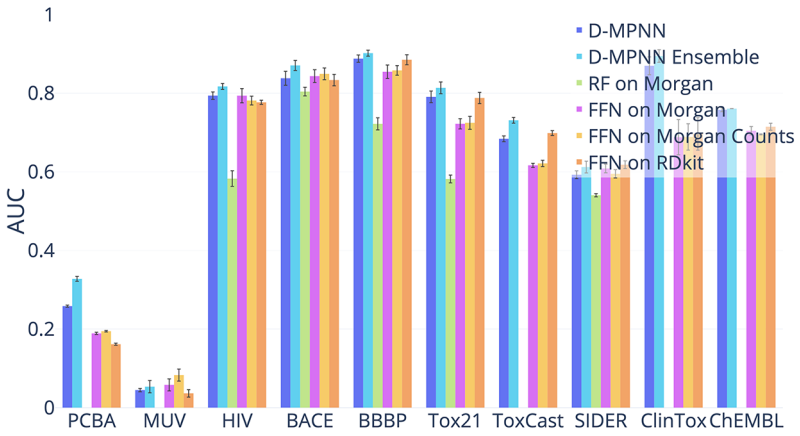
Overall, our D-MPNN is significantly better than the Mayr et al.[12] model on 8 data sets, is not significantly different on 10 data sets, and is significantly worse on 1 data set. This indicates that D-MPNN generally outperforms the Mayr et al.[12] model, especially on regression data sets.

Out-of-the-Box Comparison of D-MPNN to Other Baselines. For our final baseline comparison, we evaluate our model's

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 5. Comparison of our unoptimized D-MPNN against several baseline models. We omitted the random forest baseline on PCBA, MUV, ToxCast, and ChEMBL due to large computational cost. Random forest is omitted on ClinTox due to numerical instability. The D-MPNN significantly outperforms each baseline on at least 8 data sets.

Table 5. Details on Internal Amgen Data Sets[a]

category	data set	no. of tasks	task type	no. of compounds	metric
ADME	rPPB	1	regression	1,441	RMSE
physical chemistry	solubility	3	regression	18,007	RMSE
ADME	RLM	1	regression	64,862	RMSE
ADME	hPXR	2	regression	22,188	RMSE
ADME	hPXR (class)	2	classification	22,188	ROC-AUC

[a]Note: ADME stands for absorption, distribution, metabolism, and excretion.

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, we compare 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 descriptors.

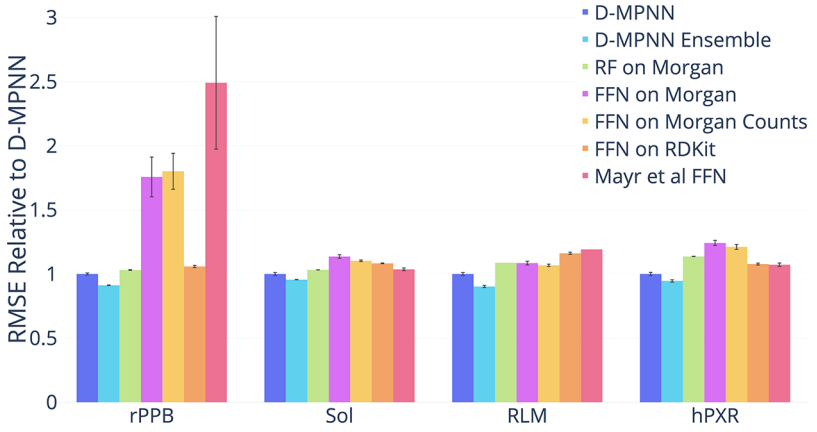
The parameters of the simple baseline models are also out-of-the-box defaults. We make this comparison in order to demonstrate the strong out-of-the-box performance of our model across a wide variety of data sets. Finally, we include the performance of the automatically optimized version of our model as a reference.

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

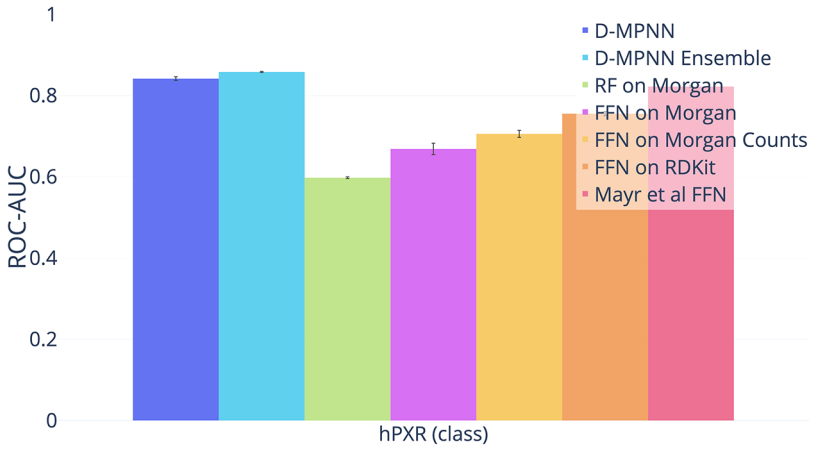
Proprietary Data Sets. We also ran our model on several private industry data sets, verifying that our model's strong performance on public data sets translates to real-world industrial data sets.

Amgen. We ran our model along with Mayr et al.[12] model and our simple baselines on four internal Amgen regression data sets. The data sets are as follows:

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 6. Comparison of our D-MPNN against baseline models on Amgen internal data sets on a chronological data split. D-MPNN outperforms all of the baselines. Note that the ensembles were ensembles of 3 models rather than 5 for the Amgen data sets only. Also note that RF on Morgan and Mayr et al. FFN were only run once on RLM.

Table 6. Details on Internal BASF Data Sets[a]

category	data set	tasks	task type	no. of compounds	metric
quantum mechanics	benzene	13	regression	30,733	R^2
quantum mechanics	cyclohexane	13	regression	30,733	R^2
quantum mechanics	dichloromethane	13	regression	30,733	R^2
quantum mechanics	DMSO	13	regression	30,733	R^2
quantum mechanics	ethanol	13	regression	30,733	R^2
quantum mechanics	ethyl acetate	13	regression	30,733	R^2
quantum mechanics	H_2O	13	regression	30,733	R^2
quantum mechanics	octanol	13	regression	30,733	R^2
quantum mechanics	tetrahydrofuran	13	regression	30,733	R^2
quantum mechanics	toluene	13	regression	30,733	R^2

[a]Note: R^2 is the square of Pearson's correlation coefficient.

1. Rat plasma protein binding free fraction (rPPB).
2. Solubility in 0.01 M hydrochloric acid solution, pH 7.4 phosphate buffer solution, and simulated intestinal fluid (Sol HCL, Sol PBS, and Sol SIF, respectively).
3. Rat liver microsomes intrinsic clearance (RLM).
4. Human pregnane X receptor % activation at 2 μM and 10 μM (hPXR).

In addition, we binarized the hPXR data set according to Amgen's recommendations in order to evaluate on a classification data set. Details of the data sets are shown in Table 5. Throughout the following, note that rPPB is in logit while Sol and RLM are in $\log_{10}$.

For each data set, we evaluate on a chronological split. Our model outperforms the baselines across the data sets, as shown in Figure 6. Thus our D-MPNN's strong performance on scaffold splits of public data sets can translate well to chronological splits of private industry data sets.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

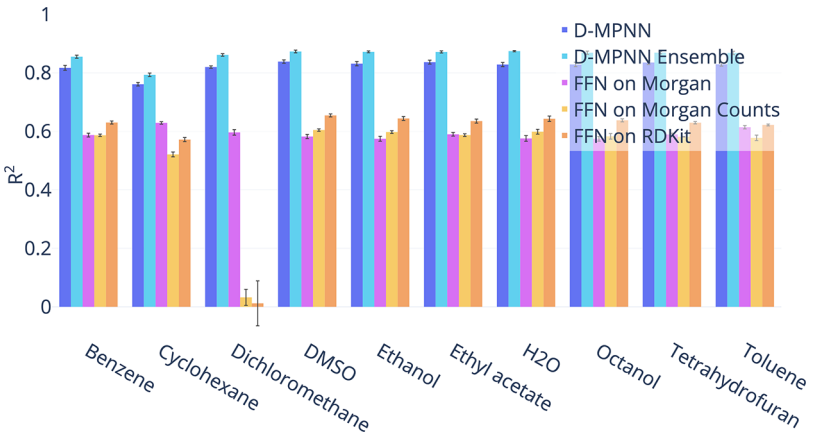


Figure 7. Comparison of our D-MPNN against baseline models on BASF internal regression data sets on a scaffold data split (higher = better). Our D-MPNN outperforms all baselines.

BASF. We ran our model on 10 highly related quantum mechanical data sets from BASF. Each data set contains 13 properties calculated on the same 30,733 molecules, varying the solvent in each data set. Data set details are in Table 6.

For these data sets, we used a scaffold-based split because a chronological split was unavailable. We found that the model of Mayr et al.[12] is numerically unstable on these data sets, and we therefore omit it from the comparison below. Once again we find that our model, originally designed to succeed on a wide range of public data sets, is robust enough to transfer to proprietary data sets as shown in Figure 7.

Novartis. Finally, we ran our model on one proprietary data set from Novartis as described in Table 7. As with other proprietary data sets, our D-MPNN outperforms the other baselines as shown in Figure 8.

Table 7. Details on the Internal Novartis Data Set

category	data set	tasks	task type	no. of compounds	metric
physical chemistry	logP	1	regression	20,294	RMSE

Experimental Error. As a final "oracle" baseline, we compare our model's performance to an experimental upper upper bound: the agreement between multiple runs of the same assay, which we refer to as the *experimental error*. Figure 9 shows the R^2 of our model on the private Amgen regression data sets together with the experimental error; in addition, this graph shows the performance of Amgen's internal model using expert-crafted descriptors. Both models remain far less accurate than the corresponding ground truth assays. Thus there remains significant space for further performance improvement in the future.

Analysis of Split Type. We now justify our use of scaffold splits for performance evaluation. The ultimate goal of building a property prediction model is to predict properties on new chemistry in order to aid the search for drugs from new classes of molecules. On proprietary company data sets, performance on new chemistry is evaluated using a chronological split of the data, i.e., everything before a certain date serves as the training set while everything after that date serves as the test set. This approximates model performance on molecules that chemists are likely to investigate in the future. Since chronological data is typically unavailable for public data sets, we investigate whether we can use our scaffold split as a reasonable proxy for a chronological split, following the work of Sheridan.[13]

Figure 10 provides motivation for a scaffold split approach. As illustrated in the figure, train and test sets according to a chronological split share fewer molecular scaffolds than train and test sets split randomly. Since our scaffold split enforces zero molecular scaffold overlap between the train and test sets, it

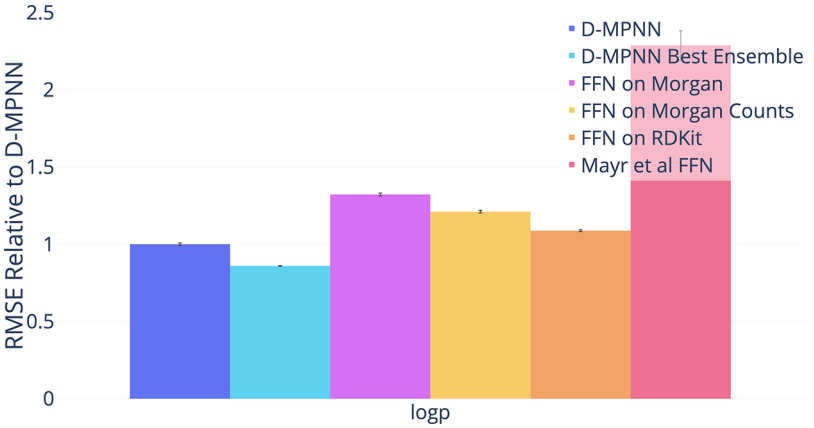


Figure 8. Comparison of our D-MPNN against baseline models on the Novartis internal regression data set on a chronological data split (lower = better). Our D-MPNN outperforms all baseline models.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

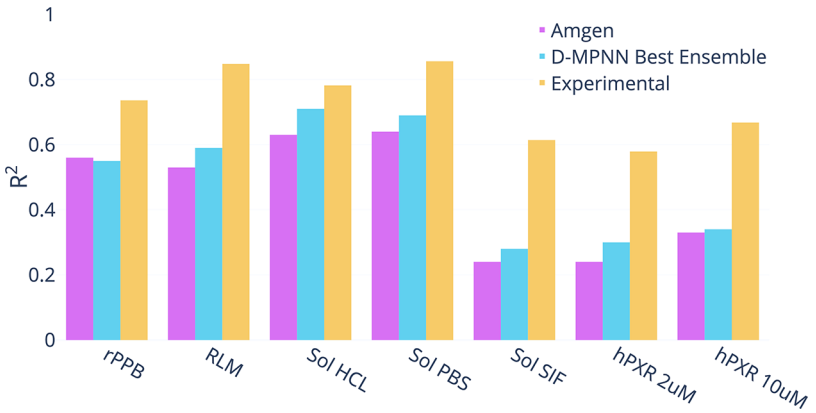


Figure 9. Comparison of Amgen's internal model and our D-MPNN (evaluated using a single run on a chronological split) to experimental error (higher = better). Note that the experimental error is not evaluated on the exact same time split as the two models since it can only be measured on molecules which were tested more than once, but even so the difference in performance is striking.

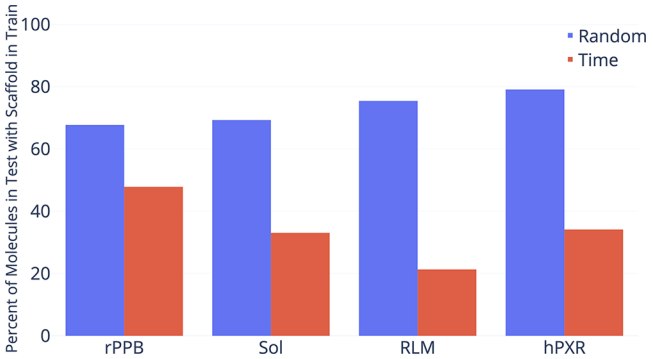


Figure 10. Overlap of molecular scaffolds between the train and test sets for a random or chronological split of four Amgen regression data sets. Overlap is defined as the percent of molecules in the test set which share a scaffold with a molecule in the train set.

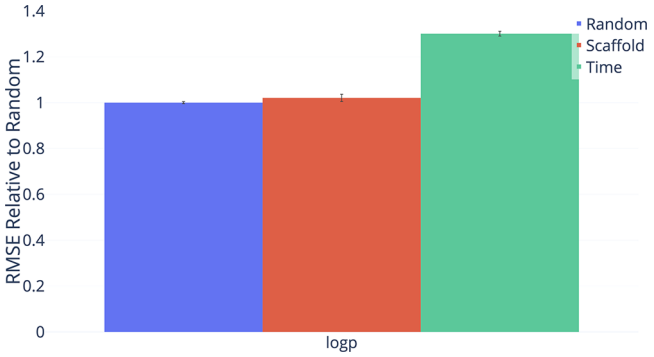


Figure 12. Performance of D-MPNN on the Novartis regression data set according to three methods of splitting the data (lower = better). The chronological split is significantly harder than the random split while the scaffold split is not.

should ideally provide a split that is at least as difficult as a chronological split.

As illustrated in Figures 11, 12, and 13, performance on our scaffold split is on average closer to performance on a chronological split on proprietary data sets from Amgen and Novartis and on the public PDBbind data sets. However, the results are noisy due to the nature of chronological splitting, where we only have a single data split, as opposed to random and scaffold splitting, which both have a random component and can generate different splits depending on the random seed. We can alleviate the problem with noise in chronological data sets by using a sliding time window to get different equally sized splits, at the cost of significantly decreasing the data set size. We report results on such sliding window splits in the Supporting Information, as the conclusions from these splits are qualitatively similar to those in the main paper.

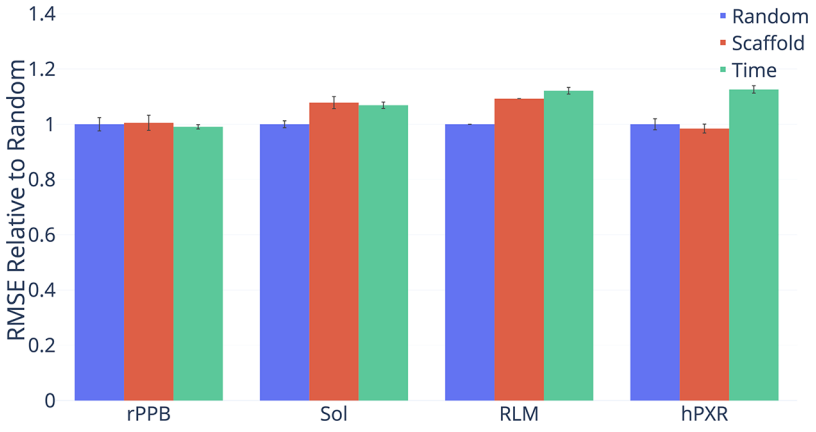


Figure 11. Performance of D-MPNN on four Amgen regression data sets according to three methods of splitting the data (lower = better). The chronological split is significantly harder than both random and scaffold on Sol and hPXR, while the scaffold split is significantly harder than the random split on Sol only.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

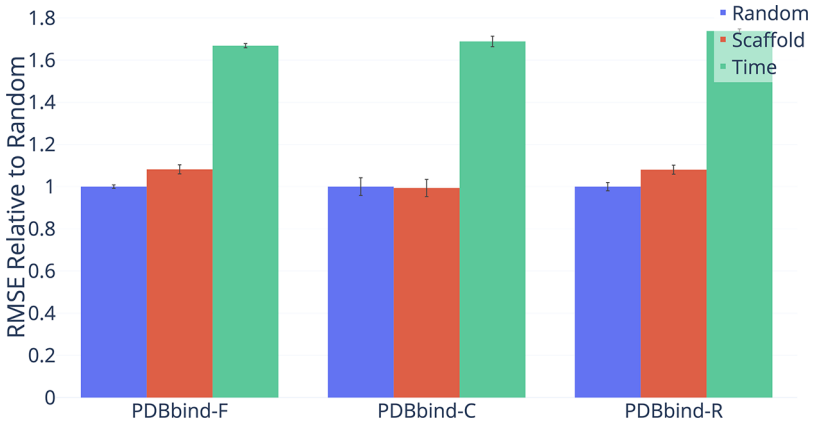
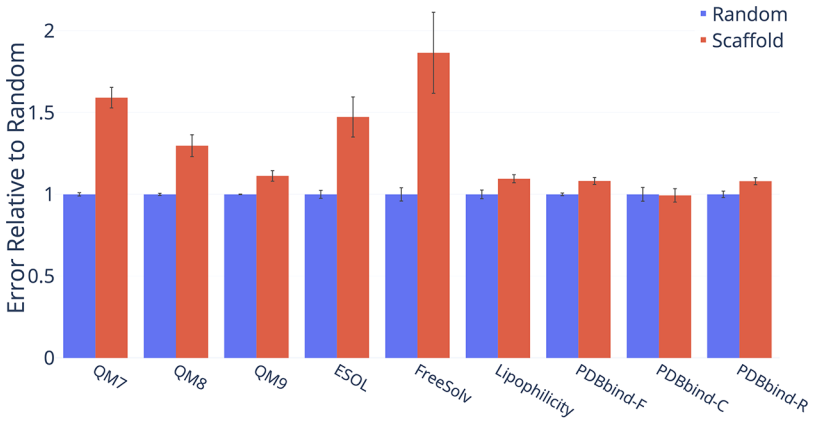
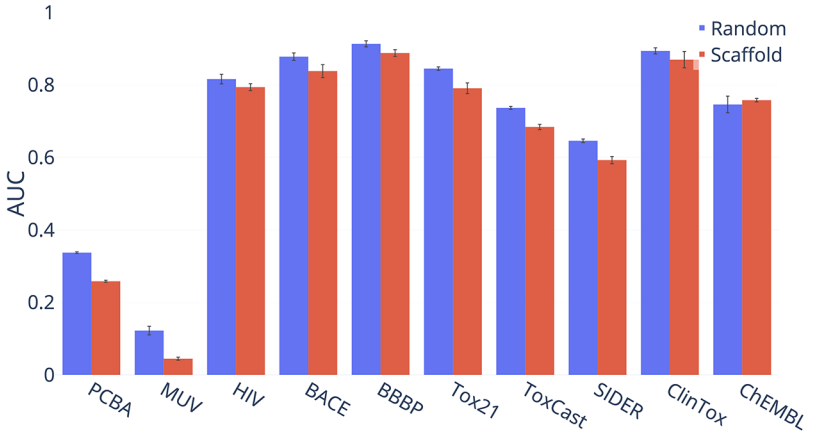


Figure 13. Performance of D-MPNN on the full (F), core (C), and refined (R) subsets of the PDBbind data set according to three methods of splitting the data (lower = better). The chronological and scaffold splits are significantly harder than the random split in all cases except for the PDBbind-C scaffold split.



(a) Regression Data Sets (lower = better).



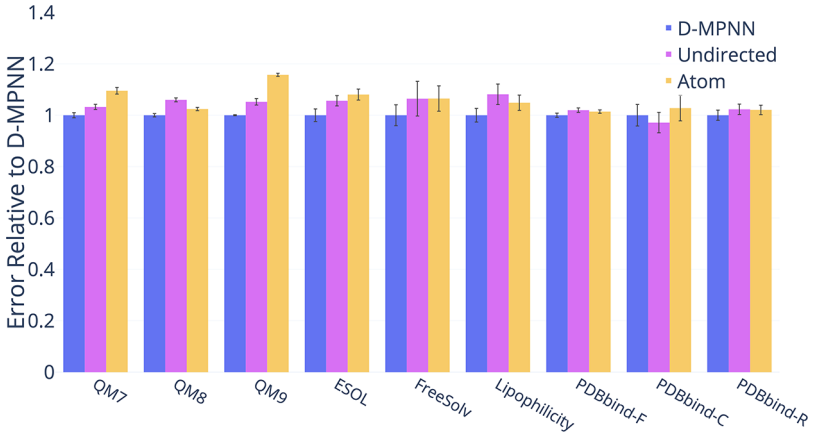
(b) Classification Data Sets (higher = better).

Figure 14. Performance of D-MPNN on random and scaffold splits for several public data sets. Only the results on PDBbind-C, HIV, ClinTox, and ChEMBL are not statistically significant.

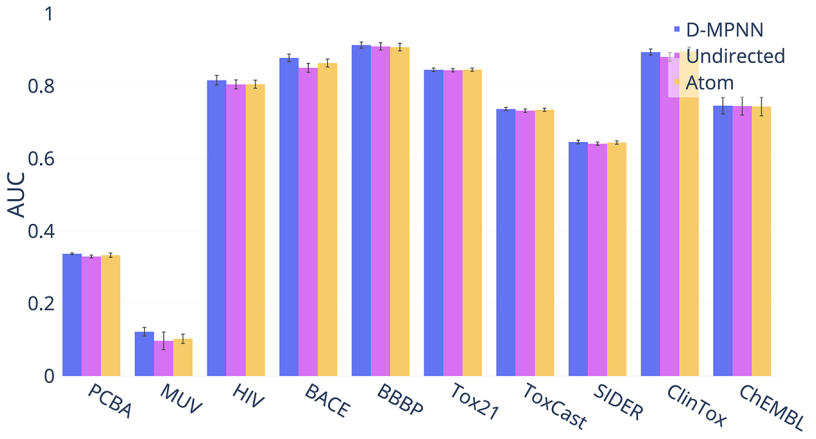
Figure 14 shows the difference between a random split and a scaffold split on the publicly available data sets, further demonstrating that a scaffold split generally results in a more difficult, and ideally more useful, measure of performance. Therefore, all of our results are reported on a scaffold split rather than a random split in order to better reflect the generalization ability of our model on new chemistry. Nevertheless, it should be emphasized that the chronological split is still the ideal split on which to evaluate when it is available. Thus we additionally report the results on chronological splits on all data sets where they are available.

Overall, our results confirm the findings of Sheridan[13] that scaffold and chronological splits are more difficult than random splits, and hence scaffold splits should be preferred over random splits during evaluation. Our findings differ somewhat from those in Sheridan[13] in that we find some evidence that

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 15. Comparison of performance of different message passing paradigms.

chronological splits may actually be harder than scaffold splits. However, owing to the small number of data sets where chronological splits are available, further investigation is necessary on this point, ideally on a larger range of data sets.

Ablations. Finally, we analyze and justify our modeling choices and optimizations.

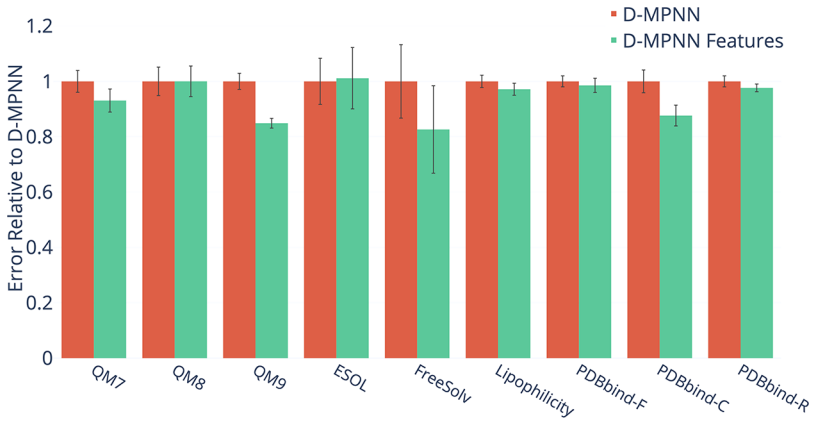
Message Type. The most important distinction between our D-MPNN and related work is the nature of the messages being passed across the molecule. Most prior work uses messages centered on atoms, whereas our D-MPNN uses messages centered on directed bonds. To isolate the effect of the message passing paradigm on property prediction performance, we implemented message passing on undirected bonds and on atoms as well, as detailed in the Supporting Information and in our code. Figure 15 illustrates the differences in performance between these three types of message passing. While on average the method using directed bonds outperforms the alternatives, the results are largely not statistically significant, so more investigation is warranted on this point.

RDKit Features. Next, we examined the impact of adding additional molecule-level features from RDKit to our model. Figure 16 shows the effect on model performance. The results appear to be highly data set-dependent. Some data sets, such as QM9 and ESOL, show marked improvement with the addition of features, while other data sets, such as PCBA and HIV, actually show worse performance with the features. We 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 our model's performance on a given data set may be further optimized by selecting different features more relevant to the task of interest.

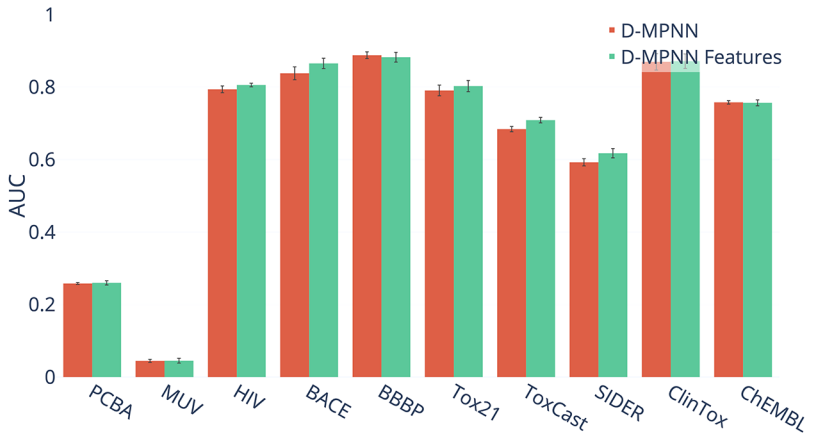
Another interesting trend is the effect of adding features to the three PDBbind data sets. The features appear to help on all three data sets, but the benefit is much more pronounced on the extremely small PDBbind-C (core) data set than it is on the larger PDBbind-R (refined) and PDBbind-F (full) data sets. 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, we hypothesize that the features may help to regularize a representation derived from a small data set: 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 data set is especially small.

Hyperparameter Optimization. To improve model performance, we performed Bayesian Optimization to select the best model hyperparameters for each data set. Figure 17 illustrates the benefit of performing this optimization, as model performance improves on virtually every data set. Interestingly, some data sets are particularly sensitive to hyperparameters. While most data sets experience a moderate 2−5% improvement in performance following hyperparameter optimization, the

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 16. Effect of adding molecule-level features generated with RDKit to our model.

quantum mechanics data sets (QM7, QM8, and QM9) and PCBA see dramatic improvements in performance, with our D-MPNN model performing 37% better on QM9 after optimization.

Ensembling. To maximize performance, we trained an ensemble of models. For each data set, we selected the best single model—i.e., the best hyperparameters along with the RDKit features if the features improved performance—and we trained five models instead of one. The results appear in Figure 19. On most data sets, ensembling only provides a small 1−5% benefit, but as with hyperparameter optimization, there are certain data sets, particularly the quantum mechanics data sets, which especially benefit from the effect of ensembling.

While each of the latter three optimizations (RDKit descriptors, hyperparameter optimization, and ensembling) on its own has limited benefits, altogether they significantly improve the model's performance on every data set except MUV.

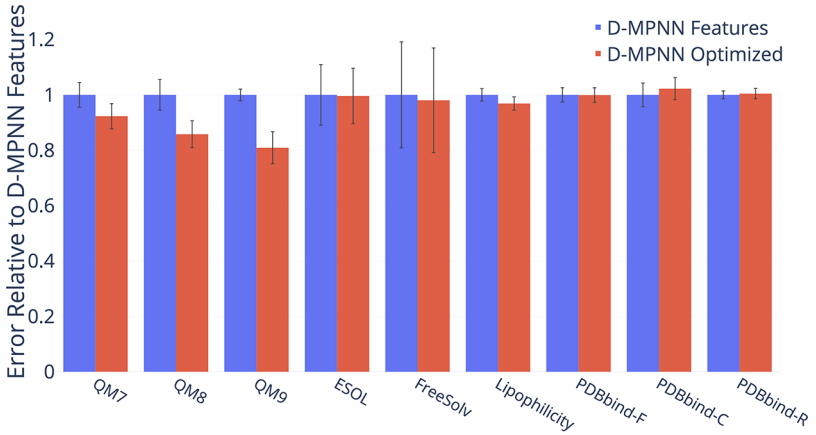
Effect of Data Size. Finally, we analyze the effect of data size on the performance of our model, using the ChEMBL data set. ChEMBL is a large data set of 456,331 molecules on 1,310 targets but is extremely sparse: only half of the 1,310 targets have at least 300 labels. For this analysis, we use the original scaffold-based split of Mayr et al.,[12] containing 3 cross-validation folds. From Figure 20, we hypothesize that our D-MPNN struggles on low-label targets in comparison to this baseline. As our D-MPNN model does not use any human-engineered fingerprints or descriptors and must therefore learn its features completely from scratch based on the input data, it would be unsurprising if the average ROC-AUC score of D-MPNN is worse than that of the feed-forward network running on human-engineered descriptors in Mayr et al.[12]

When we filter the ChEMBL data set by pruning low-data targets at different thresholds, we find that our D-MPNN indeed may outperform the best model of Mayr et al.[12] at larger data thresholds (Figure 20, though our results are not fully conclusive).

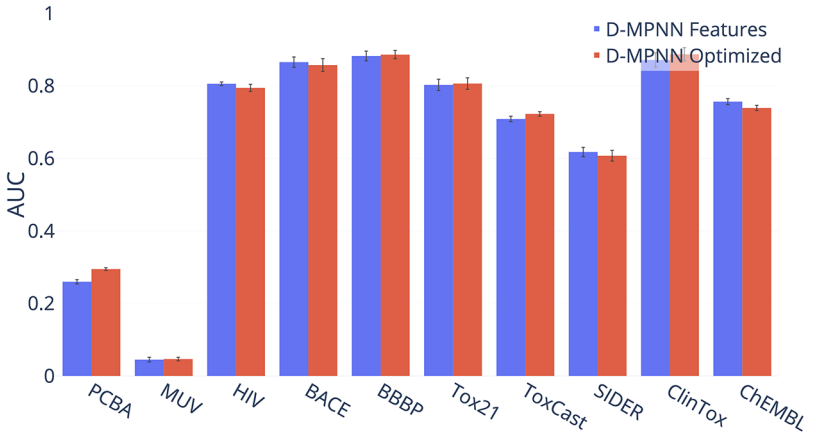
CONCLUSION AND FUTURE WORK

In this paper, we performed an extensive comparison of molecular property prediction models based on either fixed descriptors or learned molecular representations by performing over 850 experiments on 19 public and 16 proprietary data sets. Table 8 shows a summary of how our D-MPNN compares to each of the baseline models. Our model consistently matches or outperforms each baseline individually, and across all baselines, our model achieves comparable or better performance on 12 of the 19 public data sets: QM7, QM8, QM9, ESOL, FreeSolv, Lipophilicity, BBBP, PDBbind-F, PCBA, BACE, Tox21, and ClinTox. On the remaining 7 data sets, no single baseline model is consistently superior. Furthermore, our model's strong results transfer to proprietary data sets, where our model outperforms the random forest, feed-forward neural network, and Mayr et al.[12] models on every one of the 16 data sets. The strong

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 17. Effect of performing Bayesian hyperparameter optimization on the depth, hidden size, number of fully connected layers, and dropout of the D-MPNN.

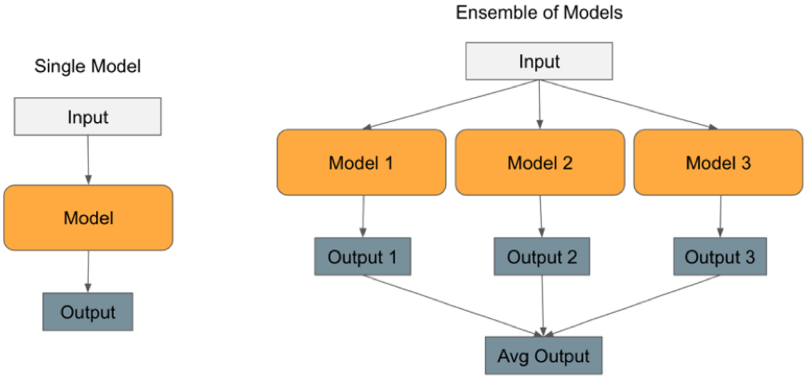
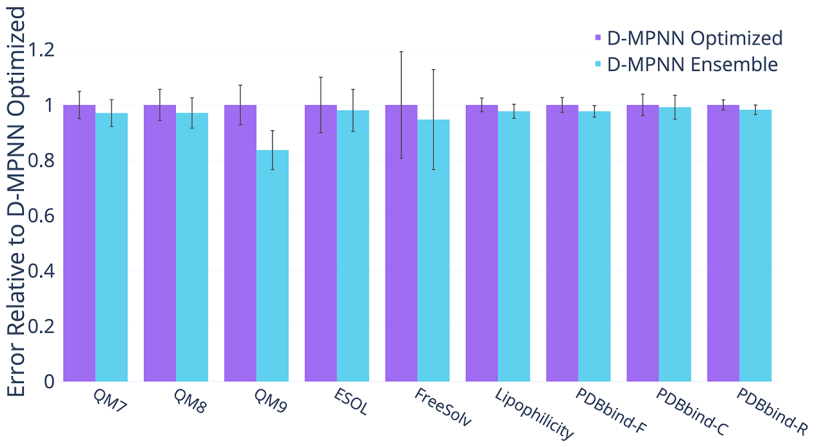


Figure 18. An illustration of ensembling models. On the left is a single model, which takes input and makes a prediction. On the right is an ensemble of 3 models. Each model takes the same input and makes a prediction independently, and then the predictions are averaged to generate the ensemble's prediction.

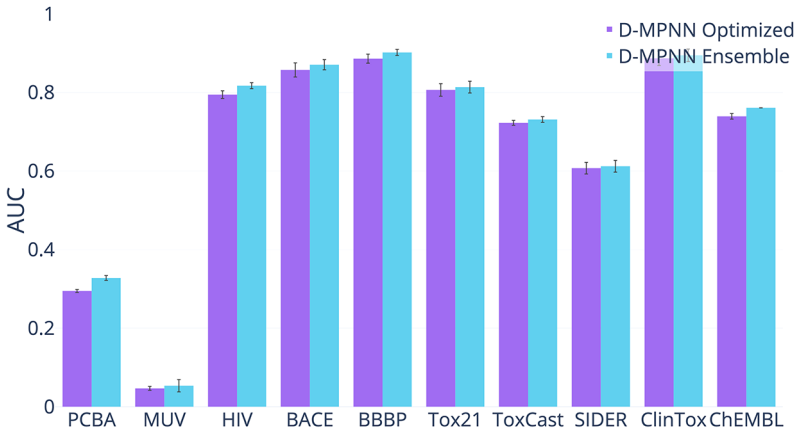
performance of our model over these baselines, many of which use computed fingerprints or descriptors, demonstrate that learned molecular representations are indeed ready for "prime time" use in industrial property prediction settings.

Nevertheless, several avenues for future research remain. When analyzing the performance of our D-MPNN, we found that it typically underperforms when either 1) the other models incorporate 3D information, as in MoleculeNet's best QM and PDBbind models, 2) the data set is especially small, as in the PDBbind-C data set, or 3) the classes are particularly imbalanced, as in the MUV data set. One avenue of improvement is the incorporation of additional 3D information into our model, which currently includes only a very restricted and naive representation of such features. Another potential improvement is a principled pretraining approach, which some authors have already begun to explore.[64,65] Such an approach

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX



(a) Regression Data Sets (lower = better).



(b) Classification Data Sets (higher = better).

Figure 19. Effect of using an ensemble of five models instead of a single model.

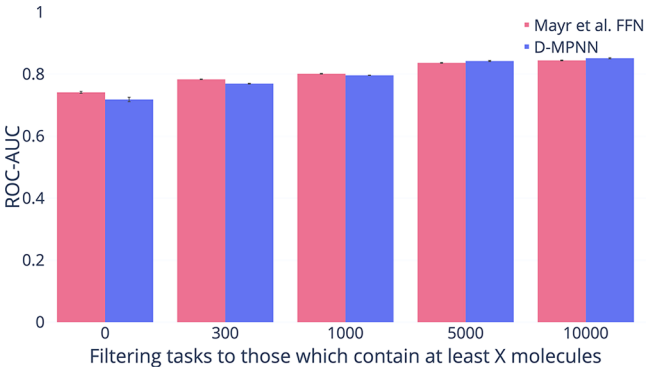


Figure 20. Effect of data size on the performance of the model from Mayr et al.[12] and of our D-MPNN model (higher = better). All comparisons besides the first are statistically significant.

could enable models to transfer learning from large chemical data sets to much smaller data sets, thereby improving performance in limited data settings. Another direction for future research is to determine how to adapt models and training algorithms to classification data sets with extreme class imbalance. Finally, in addition to these potential improvements, our analysis of how estimation of model generalizability is affected by split type opens the door to future work in uncertainty quantification and domain of applicability assessment.

Table 8. Number of Public Data Sets Where D-MPNN Is Statistically Significantly Better than, Equivalent to, or Worse than Each Baseline Model

baseline	D-MPNN is *better*	D-MPNN is the *same*	D-MPNN is *worse*	no. of data sets
MoleculeNet[2]	5	3	2	10
Mayr et al.[12]	8	10	1	19
RF on Morgan	14	0	1	15
FFN on Morgan	14	5	0	19
FFN on Morgan Counts	15	4	0	19
FFN on RDKit	8	5	4	19

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acs.jcim.9b00237.

Links to our code and to demonstration of our Web-based user interface, further comparisons of model performance on both scaffold-based and random splits of data, tables with all raw performance numbers (including p-values) which appear in charts in this paper, analysis of class balance of classification data sets, and list of RDKit calculated features used by our model (PDF)

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

AUTHOR INFORMATION

Corresponding Authors

*E-mail: yangk@mit.edu.
*E-mail: swansonk@mit.edu.

ORCID

Kyle Swanson: 0000-0002-7385-7844
Connor Coley: 0000-0002-8271-8723
Klavs Jensen: 0000-0001-7192-580X

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

This work was supported by the DARPA Make-It program under contract ARO W911NF-16-2-0023, and by the Machine Learning for Pharmaceutical Discovery and Synthesis (MLPDS) consortium. The authors from Amgen Inc., BASF, and Novartis are funded by their respective organizations. We would like to thank the other members of the computer science and chemical engineering groups in the Machine Learning for Pharmaceutical Discovery and Synthesis consortium for their helpful feedback throughout the research process. We would also like to thank the other industry members of the consortium for useful discussions regarding how to use our model in a real-world setting. We thank Nadine Schneider and Niko Fechner at Novartis for helping to analyze our model on internal Novartis data and for feedback on the manuscript. We thank Ryan White, Stephanie Geuns-Meyer, and Florian Boulnois at Amgen Inc. for their help in enabling us to run experiments on Amgen data sets. In addition, we thank Lior Hirschfeld for his work on our Web-based user interface.[53] Finally, we thank Zhenqin Wu for his aid in recreating the original data splits of Wu et al.,[2] and we thank Andreas Mayr for helpful suggestions regarding adapting the model from Mayr et al.[12] to new classification and regression data sets.

REFERENCES

(1) Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. Convolutional Networks on Graphs for Learning Molecular Fingerprints. *Advances in Neural Information Processing Systems* **2015**, 2224−2232.

(2) Wu, Z.; Ramsundar, B.; Feinberg, E.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; Pande, V. MoleculeNet: A Benchmark for Molecular Machine Learning. *Chem. Sci.* **2018**, *9*, 513−530.

(3) Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. Molecular Graph Convolutions: Moving Beyond Fingerprints. *J. Comput.-Aided Mol. Des.* **2016**, *30*, 595−608.

(4) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. Neural Message Passing for Quantum Chemistry. *Proceedings of the 34th International Conference on Machine Learning* **2017**, *70*, 1263−1272.

(5) Li, Y.; Tarlow, D.; Brockschmidt, M.; Zemel, R. Gated Graph Sequence Neural Networks. 2015, *arXiv preprint arXiv:1511.05493*. https://arxiv.org/abs/1511.05493 (accessed Aug 6, 2019).

(6) Kipf, T. N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. 2016, *arXiv preprint arXiv:1609.02907*. https://arxiv.org/abs/1609.02907 (accessed Aug 6, 2019).

(7) Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *Advances in Neural Information Processing Systems* **2016**, 3844−3852.

(8) Bruna, J.; Zaremba, W.; Szlam, A.; LeCun, Y. Spectral Networks and Locally Connected Networks on Graphs. 2013, *arXiv preprint arXiv:1312.6203*. https://arxiv.org/abs/1312.6203 (accessed Aug 6, 2019).

(9) Coley, C. W.; Barzilay, R.; Green, W. H.; Jaakkola, T. S.; Jensen, K. F. Convolutional Embedding of Attributed Molecular Graphs for Physical Property Prediction. *J. Chem. Inf. Model.* **2017**, *57*, 1757−1772.

(10) Schütt, K. T.; Arbabzadah, F.; Chmiela, S.; Müller, K. R.; Tkatchenko, A. Quantum-Chemical Insights from Deep Tensor Neural Networks. *Nat. Commun.* **2017**, *8*, 13890.

(11) Battaglia, P.; Pascanu, R.; Lai, M.; Rezende, D. J. Interaction Networks for Learning about Objects, Relations and Physics. *Advances in Neural Information Processing Systems* **2016**, 4502−4510.

(12) Mayr, A.; Klambauer, G.; Unterthiner, T.; Steijaert, M.; Wegner, J. K.; Ceulemans, H.; Clevert, D.-A.; Hochreiter, S. Large-Scale Comparison of Machine Learning Methods for Drug Target Prediction on ChEMBL. *Chem. Sci.* **2018**, *9*, 5441−5451.

(13) Sheridan, R. P. Time-Split Cross-Validation as a Method for Estimating the Goodness of Prospective Prediction. *J. Chem. Inf. Model.* **2013**, *53*, 783−790.

(14) Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; de Freitas, N. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* **2016**, *104*, 148−175.

(15) Dietterich, T. G. Ensemble Methods in Machine Learning. *International Workshop on Multiple Classifier Systems* **2000**, *1857*, 1−15.

(16) Hamilton, W.; Ying, Z.; Leskovec, J. Inductive Representation Learning on Large Graphs. *Advances in Neural Information Processing Systems* **2017**, 1024−1034.

(17) Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; Monfardini, G. The Graph Neural Network Model. *IEEE Transactions on Neural Networks* **2009**, *20*, 61−80.

(18) Henaff, M.; Bruna, J.; LeCun, Y. Deep Convolutional Networks on Graph-Structured Data. 2015, *arXiv preprint arXiv:1506.05163*. https://arxiv.org/abs/1506.05163 (accessed Aug 6, 2019).

(19) Dai, H.; Dai, B.; Song, L. Discriminative Embeddings of Latent Variable Models for Structured Data. *International Conference on Machine Learning* **2016**, 2702−2711.

(20) Lei, T.; Jin, W.; Barzilay, R.; Jaakkola, T. Deriving Neural Architectures from Sequence and Graph Kernels. 2017, *arXiv preprint arXiv:1705.09037*. https://arxiv.org/abs/1705.09037 (accessed Aug 6, 2019).

(21) Kusner, M. J.; Paige, B.; Hernández-Lobato, J. M. Grammar Variational Autoencoder. 2017, *arXiv preprint arXiv:1703.01925*. https://arxiv.org/abs/1703.01925 (accessed Aug 6, 2019).

(22) Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci.* **2018**, *4*, 268−76.

(23) Jin, W.; Barzilay, R.; Jaakkola, T. Junction Tree Variational Autoencoder for Molecular Graph Generation. 2018, *arXiv preprint arXiv:1802.04364*. https://arxiv.org/abs/1802.04364 (accessed Aug 6, 2019).

(24) Jin, W.; Yang, K.; Barzilay, R.; Jaakkola, T. Learning Multimodal Graph-to-Graph Translation for Molecular Optimization. 2018, *arXiv preprint arXiv:1812.01070*. https://arxiv.org/abs/1812.01070 (accessed Aug 6, 2019).

(25) Cortes, C.; Vapnik, V. Support Vector Machine. *Machine Learning* **1995**, *20*, 273−297.

(26) Breiman, L. Random Forests. *Machine Learning* **2001**, *45*, 5−32.

(27) Mauri, A.; Consonni, V.; Pavan, M.; Todeschini, R. Dragon Software: An Easy Approach to Molecular Descriptor Calculations. *Match* **2006**, *56*, 237−248.

(28) Rogers, D.; Hahn, M. Extended-Connectivity Fingerprints. *J. Chem. Inf. Model.* **2010**, *50*, 742−754.

(29) Swamidass, S. J.; Chen, J.; Bruand, J.; Phung, P.; Ralaivola, L.; Baldi, P. Kernels for Small Molecules and the Prediction of Mutagenicity, Toxicity and Anti-Cancer Activity. *Bioinformatics* **2005**, *21*, i359−i368.

(30) Cao, D.-S.; Xu, Q.-S.; Hu, Q.-N.; Liang, Y.-Z. ChemoPy: Freely Available Python Package for Computational Biology and Chemoinformatics. *Bioinformatics* **2013**, *29*, 1092−1094.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

(31) Durant, J. L.; Leland, B. A.; Henry, D. R.; Nourse, J. G. Reoptimization of MDL Keys for Use in Drug Discovery. *J. Chem. Inf. Model.* **2002**, *42*, 1273−1280.
(32) Moriwaki, H.; Tian, Y.-S.; Kawashita, N.; Takagi, T. Mordred: A Molecular Descriptor Calculator. *J. Cheminf.* **2018**, *10*, 4.
(33) Schütt, K.; Kindermans, P.-J.; Felix, H. E. S.; Chmiela, S.; Tkatchenko, A.; Müller, K.-R. SchNet: A Continuous-Filter Convolutional Neural Network for Modeling Quantum Interactions. *Advances in Neural Information Processing Systems* **2017**, 991−1001.
(34) Kondor, R.; Son, H. T.; Pan, H.; Anderson, B.; Trivedi, S. Covariant Compositional Networks for Learning Graphs. 2018, *arXiv preprint arXiv:1801.02144*. https://arxiv.org/abs/1801.02144 (accessed Aug 6, 2019).
(35) Faber, F. A.; Hutchison, L.; Huang, B.; Gilmer, J.; Schoenholz, S. S.; Dahl, G. E.; Vinyals, O.; Kearnes, S.; Riley, P. F.; von Lilienfeld, O. A. Machine Learning Prediction Errors Better than DFT Accuracy. 2017, *arXiv preprint arXiv:1702.05532*. https://arxiv.org/abs/1702.05532 (accessed Aug 6, 2019).
(36) Feinberg, E. N.; Sur, D.; Wu, Z.; Husic, B. E.; Mai, H.; Li, Y.; Sun, S.; Yang, J.; Ramsundar, B.; Pande, V. S. PotentialNet for Molecular Property Prediction. *ACS Cent. Sci.* **2018**, *4*, 1520−1530.
(37) Lee, A. A.; Yang, Q.; Bassyouni, A.; Butler, C. R.; Hou, X.; Jenkinson, S.; Price, D. A. Ligand Biological Activity Predicted by Cleaning Positive and Negative Chemical Correlations. *Proc. Natl. Acad. Sci. U. S. A.* **2019**, *116*, 3373.
(38) Weininger, D. SMILES, A Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Model.* **1988**, *28*, 31−36.
(39) Ishiguro, K.; Maeda, S.-i.; Koyama, M. Graph Warp Module: An Auxiliary Module for Boosting the Power of Graph Neural Networks. 2019, *arXiv preprint arXiv:1902.01020*. https://arxiv.org/abs/1902.01020 (accessed Aug 6, 2019).
(40) Liu, K.; Sun, X.; Jia, L.; Ma, J.; Xing, H.; Wu, J.; Gao, H.; Sun, Y.; Boulnois, F.; Fan, J. Chemi-net: A Graph Convolutional Network for Accurate Drug Property Prediction. 2018, *arXiv preprint arXiv:1803.06236*. https://arxiv.org/abs/1803.06236 (accessed Aug 6, 2019).
(41) Mahé, P.; Ueda, N.; Akutsu, T.; Perret, J.-L.; Vert, J.-P. Extensions of Marginalized Graph Kernels. *Proceedings of the Twenty-First International Conference on Machine Learning*; 2004; p 70.
(42) Koller, D.; Friedman, N.; Bach, F. *Probabilistic Graphical Models: Principles and Techniques*; MIT Press: 2009.
(43) Nair, V.; Hinton, G. E. Rectified Linear Units Improve Restricted Boltzmann Machines. *Proceedings of the 27th International Conference on Machine Learning*; 2010; pp 807−814.
(44) Landrum, G. *RDKit: Open-Source Cheminformatics*; 2006. https://rdkit.org/docs/index.html (accessed 2019-05-24).
(45) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825−2830.
(46) Irwin, J. J.; Shoichet, B. K. ZINC−A Free Database of Commercially Available Compounds for Virtual Screening. *J. Chem. Inf. Model.* **2005**, *45*, 177−182.
(47) Kim, S.; Thiessen, P. A.; Bolton, E. E.; Chen, J.; Fu, G.; Gindulyte, A.; Han, L.; He, J.; He, S.; Shoemaker, B. A. PubChem Substance and Compound Databases. *Nucleic Acids Res.* **2016**, *44*, D1202−D1213.
(48) Sartor, M. A.; Leikauf, G. D.; Medvedovic, M. LRpath: A Logistic Regression Approach for Identifying Enriched Biological Groups in Gene Expression Data. *Bioinformatics* **2009**, *25*, 211−217.
(49) *Distributed Asynchronous Hyperparameter Optimization in Python*. https://github.com/hyperopt/hyperopt (accessed 2019-05-24).
(50) Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in PyTorch. *31st Conference on Neural Information Processing Systems*; 2017.
(51) *Message Passing Neural Networks for Molecule Property Prediction*. https://github.com/swansonk14/chemprop (accessed 2019-05-24).
(52) *Descriptor computation(chemistry) and (optional) storage for machine learning*. https://github.com/bp-kelley/descriptastorus (accessed 2019-05-24).
(53) *Chemprop Machine Learning for Molecular Property Prediction*. http://chemprop.csail.mit.edu (accessed 2019-05-24).
(54) Friedman, J. H. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* **2001**, *29*, 1189−1232.
(55) Cramer, J. S. *Logit Models from Economics and Other Fields*; 2003; DOI: 10.1017/CBO9780511615412.
(56) 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*, 1563−1575.
(57) Cortes, C.; Vapnik, V. Support-Vector Networks. *Machine Learning* **1995**, *20*, 273−297.
(58) Ma, J.; Sheridan, R. P.; Liaw, A.; Dahl, G. E.; Svetnik, V. Deep Neural Nets as a Method for Quantitative Structure−Activity Relationships. *J. Chem. Inf. Model.* **2015**, *55*, 263−274.
(59) Ramsundar, B.; Liu, B.; Wu, Z.; Verras, A.; Tudor, M.; Sheridan, R. P.; Pande, V. Is Multitask Deep Learning Practical for Pharma? *J. Chem. Inf. Model.* **2017**, *57*, 2068−2076.
(60) Swamidass, S. J.; Azencott, C.-A.; Lin, T.-W.; Gramajo, H.; Tsai, S.-C.; Baldi, P. Influence Relevance Voting: An Accurate and Interpretable Virtual High Throughput Screening Method. *J. Chem. Inf. Model.* **2009**, *49*, 756−766.
(61) Smith, J. S.; Isayev, O.; Roitberg, A. E. ANI-1: An Extensible Neural Network Potential with DFT Accuracy at Force Field Computational Cost. *Chem. Sci.* **2017**, *8*, 3192−3203.
(62) Ramsundar, B.; Eastman, P.; Leswing, K.; Walters, P.; Pande, V. *Deep Learning for th0e Life Sciences*; O'Reilly Media: 2019.
(63) *Scripts for running lsc model on other datasets*. https://github.com/yangkevin2/lsc_experiments (accessed 2019-05-24).
(64) Navarin, N.; Tran, D. V.; Sperduti, A. Pre-training Graph Neural Networks with Kernels. 2018, *arXiv preprint arXiv:1811.06930*. https://arxiv.org/abs/1811.06930 (accessed Aug 6, 2019).
(65) Goh, G. B.; Siegel, C.; Vishnu, A.; Hodas, N. Using Rule-Based Labels for Weak Supervised Learning: A ChemNet for Transferable Chemical Property Prediction. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*; 2018; pp 302−310, DOI: 10.1145/3219819.3219838.

DOI: 10.1021/acs.jcim.9b00237
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX