JOURNAL OF CHEMICAL INFORMATION AND MODELING

Article

pubs.acs.org/jcim

Is Multitask Deep Learning Practical for Pharma?

Bharath Ramsundar,[†] Bowen Liu,[‡] Zhenqin Wu,[‡] Andreas Verras,[¶] Matthew Tudor,[§] Robert P. Sheridan,[¶] and Vijay Pande*[,‡]

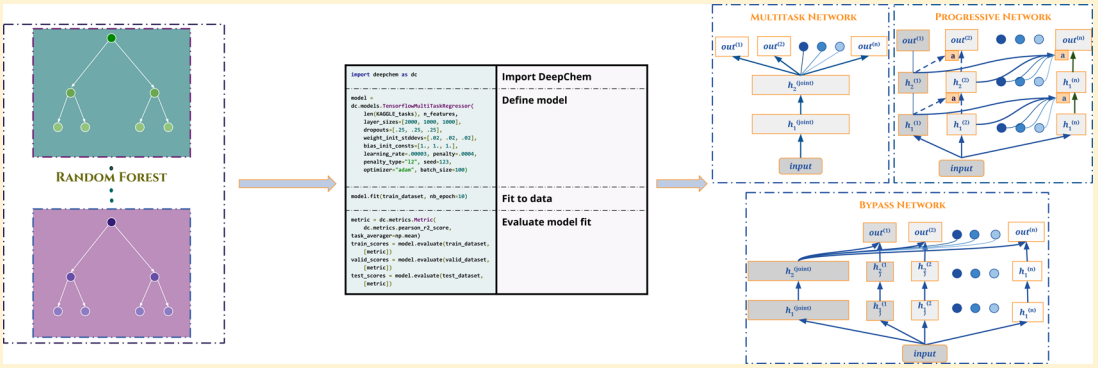
[†]Department of Computer Science, Stanford University, Stanford, California 94305, United States

[‡]Department of Chemistry, Stanford University, Stanford, California 94305, United States

[¶]Chemistry Capabilities and Screening, Merck & Co., Inc., 2000 Galloping Hill Road, Kenilworth, New Jersey 07033, United States

[§]Chemistry Capabilities and Screening, Merck & Co., Inc., 770 Sumneytown Pike, West Point, Pennsylvania 19846, United States

Supporting Information



ABSTRACT: Multitask deep learning has emerged as a powerful tool for computational drug discovery. However, despite a number of preliminary studies, multitask deep networks have yet to be widely deployed in the pharmaceutical and biotech industries. This lack of acceptance stems from both software difficulties and lack of understanding of the robustness of multitask deep networks. Our work aims to resolve both of these barriers to adoption. We introduce a high-quality open-source implementation of multitask deep networks as part of the DeepChem open-source platform. Our implementation enables simple python scripts to construct, fit, and evaluate sophisticated deep models. We use our implementation to analyze the performance of multitask deep networks and related deep models on four collections of pharmaceutical data (three of which have not previously been analyzed in the literature). We split these data sets into train/valid/test using time and neighbor splits to test multitask deep learning performance under challenging conditions. Our results demonstrate that multitask deep networks are surprisingly robust and can offer strong improvement over random forests. Our analysis and open-source implementation in DeepChem provide an argument that multitask deep networks are ready for widespread use in commercial drug discovery.

INTRODUCTION

Deep learning has had unprecedented impact on computer science and technology over the last 5 years. The advent of sophisticated deep models has enabled the construction of powerful visual, speech, and natural language understanding systems.[1] Over the past few years, deep learning has begun to influence drug discovery. In 2012, MSD (Merck & Co., Inc., Kenilworth, NJ U.S.A.) hosted a Kaggle contest to measure the ability of data science to improve predictive accuracies of quantitative structure−activity relationship (QSAR) methods. The winning entry used multitask deep networks (ensembled with other machine-learning techniques) to achieve a 15% relative improvement over the baseline.[2] In follow-up work, MSD demonstrated that strong improvements could be achieved on the Kaggle data sets using a single deep learning architecture, without constructing ensemble models.[3] In parallel, other research has demonstrated that large-scale multitask deep networks can yield strong improvements over singletask models on public data sets with hundreds of tasks, extending MSD's work with smaller data set collections.[4,5] Similarly, recent work from Vertex demonstrates that multitask deep networks can offer improvements over simpler methods for modeling ADMET (absorption, distribution, metabolism, excretion, and toxicity) data sets.[6]

However, despite significant preliminary research, deep multitask networks have not succeeded in achieving broad adoption in the pharmaceutical and biotechnology industries. Transforming a machine-learning-based research prototype into a production system can present many challenges. For example, the winning entry for Netflix's million dollar recommendation challenge was not actually deployed in Netflix's production system due to engineering costs.[7] Parts of the solution (singular value decompositions and restricted Boltzmann machines)

Received: March 13, 2017
Published: July 10, 2017

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

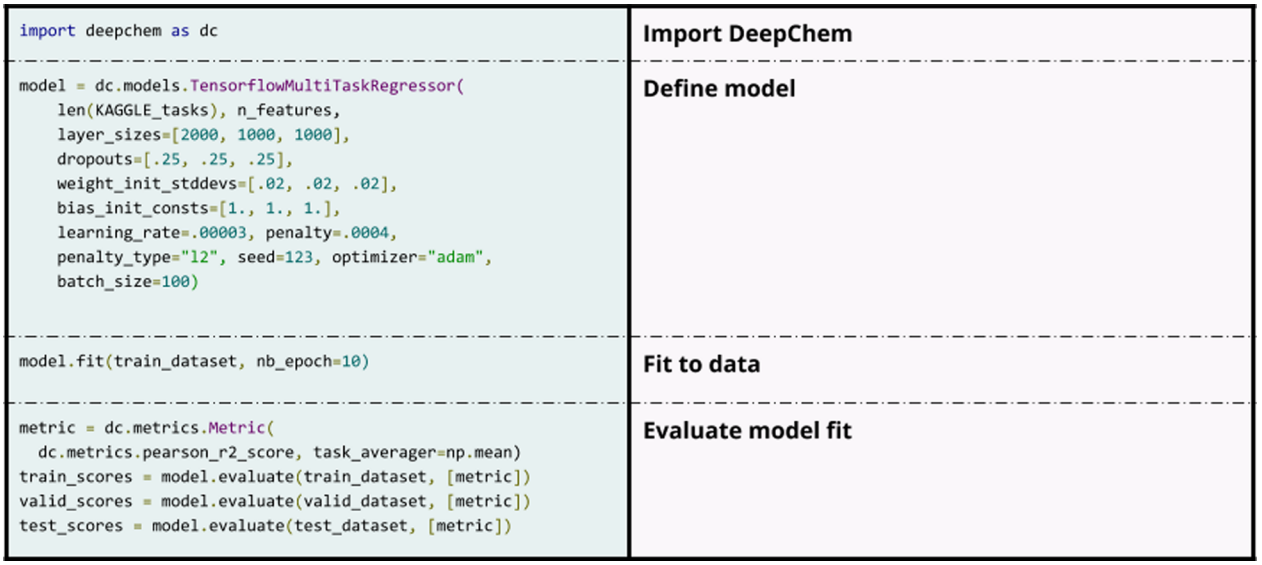


Figure 1. Example code for defining, training, and evaluating multitask deep networks for the Kaggle data set with DeepChem. Code example is nearly complete, except for loading code. Full code for building networks on Kaggle data sets is available online.

were deployed, but shifting customer requirements rendered the full solution too unwieldy for broad implementation.[8]

Similar challenges may well be slowing the broad adoption of multitask deep networks in the pharmaceutical industry. Drug discovery companies often do not have strong software development teams, and creating a high-quality multitask deep network implementation is a formidable software engineering challenge. Each piece of research cited above uses an independent multitask implementation. While a research team may be able to create custom deep learning implementations, such feats remain challenging for industrial teams. Until recently, a similar situation held more broadly within the deep learning community. While research teams could implement deep learning systems, industrial users often stayed away due to the engineering challenges. However, this situation changed dramatically with the advent of sophisticated open-source deep learning frameworks such as Tensorflow, Theano, and Keras.[9−11] These platforms offer convenient software primitives for building sophisticated deep architectures with low overhead. As a result, the adoption of deep learning techniques by the software industry has increased accordingly.

While deep learning systems offer powerful conveniences for users, the native packages are not well-suited to handle chemical data sets. Data loading procedures do not support common chemical formats such as SMILES strings[12] or provide for data splits based on chemical scaffolds.[13] The DeepChem library developed at Stanford provides a wrapper around Tensorflow that understands and facilitates the processing of chemical data sets.[14] DeepChem has been used for both application projects (modeling inhibitor design for BACE-1[15]) and algorithmic development (of novel one-shot deep learning techniques for drug discovery[16]).

In this work, we introduce a high-quality multitask deep network implementation to DeepChem. We use this implementation to construct multitask models on four broad collections of pharmaceutical data. We start by using DeepChem's implementation to reproduce results on the Kaggle collection explored by MSD previously. We then construct multitask deep models on the Factors, Kinase, and UV data set collections from MSD (previously not analyzed in the literature). We aim to investigate whether multitask deep learning is a robust tool, one that provides strong results across broad collections of pharmaceutical data.

We focus our analysis on gauging whether multitask networks provide consistent improvements over random forest (RF) baselines. In particular, we aim to answer the question of whether deep networks can replace RF models without causing significant failures on a subset of predictive models. Furthermore, due to the ease of implementing new deep learning architectures in DeepChem, we test the performances of two alternate deep learning architectures on these data sets. Google's Deepmind recently proposed progressive neural networks as a suitable architecture for solving sequences of challenging learning tasks[17] while leveraging transfer learning and avoiding learning failures. Progressive networks require an ordering of learning tasks (often unnatural in a drug discovery setting with a collection of assays); therefore, we also consider a variant bypass architecture halfway between multitask networks and progressive networks. Our analysis reveals that, while these alternative architectures offer reasonable results, the simple multitask deep architecture currently remains the most robust deep architecture for QSAR data sets. On average, multitask deep networks offer performance boosts over RF methods, but the improvements are not yet across the board.

To encourage adoption of multitask deep learning methods, we open source all modeling code and data sets for the Kaggle, Factors, Kinase, and UV data set collections as part of the DeepChem example suite. We hope that this example code and data will facilitate broader adoption of multitask deep learning techniques for commercial drug discovery.

■ MULTITASK DEEP LEARNING WITH DEEPCHEM

DeepChem offers a python API for constructing multitask deep learning models. The API is object-oriented and modeled on the interfaces for popular machine-learning packages such as Scikit-Learn[18] and Keras.[11] As part of this work, we contribute a high-quality multitask deep network implementation to DeepChem. Users can build, train, and evaluate multitask deep networks with simple python scripts.

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

Figure 1 provides an (almost complete) code sample for using the multitask deep learning API in DeepChem to build a model for the Kaggle data collection. Users can specify model hyperparameters in the constructor for the multitask deep network. Figure 1 demonstrates how a three hidden layer deep network with dropout[19] and L^2 regularization can be specified. Calling the `fit()` method fits the model using the ADAM optimization algorithm[20] for 10 epochs over the training data set. Note that code in Figure 1 has a `seed` flag for deep models. This flag allows multitask deep networks to be constructed in DeepChem with fixed random seed. Otherwise, model performance may vary on the order of a few percent due to random initialization choice, making debugging and model optimization challenging. Calling the `evaluate()` method allows users to evaluate the model using the squared Pearson correlation coefficient.

The full example for the Kaggle data set is only slightly longer and has been open-sourced as part of the DeepChem example suite.

DATA SETS

Data Description. Modeling was performed on four collections of assays from MSD (see Table 1 for details). The Kaggle collection is a collection of 15 enzymatic inhibition and ADME/Tox data sets[3] with about 100 000 unique compounds. Not every compound in the Kaggle collection is tested against every assay. The Factors collection measures about 1500 of MSD's in-house compounds for IC50 of inhibition on 12 serine proteases, some of which are blood-clotting factors. The Kinase colection contains about 2500 in-house compounds measured for IC50 of inhibition on 99 protein kinases, carried out by a contract research partner. The UV collection tests 10 000 compounds on 190 absorption wavelengths, between 210 and 400 nm.

Table 1. Data Set Collections and Approximate Training Data Counts

data set collection	number of compounds	number of tasks
Kaggle	100 000	15
Factors	1500	12
Kinase	2500	99
UV	10 000	190

Precomputed AP,DP (atom-pair, donor-pair) descriptors[21,22] (of the same form as that described in previous work[3]) were utilized for these data collections. These descriptors used an unspecified ordering in order to make it difficult to reverse-engineer compound identities from AP,DP descriptors. Data sets were featurized and anonymized at MSD. The outputs were stored as CSV files. Each data set collection was stored as two files. The descriptor files had separate columns holding descriptor values and compound IDs. The activities files had columns for compound IDs and for the various assays in the data set collection. The descriptors were permuted and the IDs arbitrarily assigned to numerical values to prevent identification. The descriptor and activity files were transferred from MSD to Stanford for analysis. At Stanford, the activity and descriptor files were combined into single CSV files (one for each data set collection, perhaps sharded for convenience). The joined CSV files were directly fed into DeepChem for analysis (using `dc.feat.UserDefinedFeaturizer`).

This method of anonymizing compound data for transfer to academic partners may prove a useful model for future collaborations between industry and academia. However, as we discuss later, the lack of access to chemical structures may limit the predictive power of anonymized models.

Data Splits. All data sets were split into training, validation, and test sets at MSD. The Kaggle and UV collections were split 75/25 into training and validation/test using time splits (training compounds were experimentally evaluated before those in validation and test; the remaining 25% was randomly split into validation/test). The Factors and Kinase collections were split 75/25 using neighbor splits.[23] That is, the test set contained the compounds with the fewest neighbors, where neighbors are defined as more similar than 0.7 using the AP descriptor and Dice similarity index. Note that both time and neighbor splits are known to be challenging for multitask deep networks; therefore, this choice of evaluation forms a particularly challenging test for multitask deep network performance relative to RFs.

The train, validation, and test collections were transmitted from MSD in separate files so that no choice of data splitting was made at Stanford. Following standard practice, the training set was used to train machine-learning models, the validation set for tuning model hyperparameters, and the test set for final evaluation of trained models.

MACHINE-LEARNING MODELS

We tested a number of machine-learning algorithms on MSD's internal data sets. The following sections briefly describe RFs,[24] multitask deep networks,[3] progressive networks,[17] and our hybrid bypass architecture. All deep learning models discussed below were implemented as part of this work and contributed to DeepChem. The multitask, progressive, and bypass architectures were contributed to DeepChem as part of this

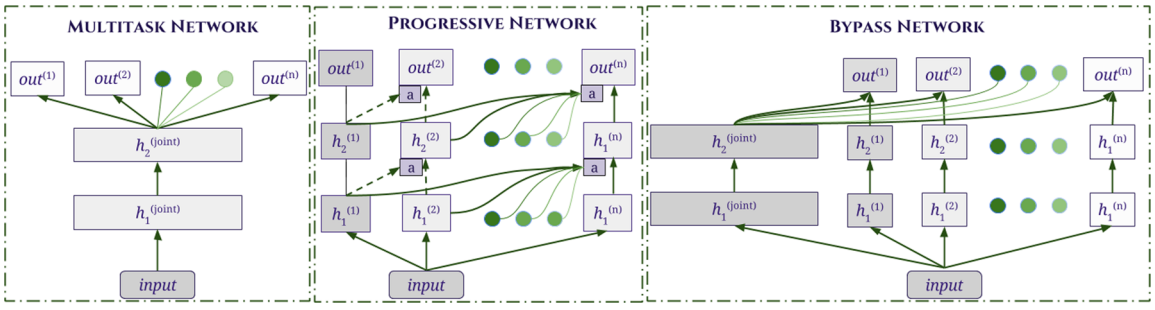


Figure 2. (Left) Standard multitask architecture. (Center) Progressive architecture. Dotted lines indicate frozen weights. (Right) Proposed bypass architecture. Note that terms such as "input" and "h" denote rows of neurons and not single neurons.

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

work. Figure 2 provides a graphical depiction of all deep architectures tested.

Random Forests. RFs[24] are an ensemble prediction method where individual decision trees are trained on subsampled subsets of the original data set. The results for individual trees are averaged to provide the output for the entire system. RFs are currently widely used in the pharmaceutical industry for QSAR tasks.[25]

We train RF models with 100 trees each and with $m/3$ descriptors used at each node, where m is the number of features in the AP,DP descriptors (using hyperparameters from previous work[3]). Tree nodes with five or fewer molecules are not split further. We use the RF implementation from Scikit-Learn[18] but through the DeepChem API for convenience.

Singletask Architecture. Singletask deep networks train a separate multilayer neural network for each learning task in the data set collection. The model for each task is trained separately using backpropagation on the data set for that task.

Multitask Architecture. Multitask deep networks train a joint representation of input data points shared for all learning tasks. This shared joint representation is fed into a separate linear model for each data set in a data set collection. The entire model is trained end-to-end on training data using backpropagation. The left panel in Figure 2 provides a graphical representation of the architecture. Detailed mathematical descriptions have been provided in previous works.[3,4]

Progressive Architecture. The problem of training models capable of solving many tasks is shared by many applications in machine-learning. The progressive neural network architecture[17] proposes a general solution to this problem by allotting each learning task an independent column of weights. However, the column for task i can refer to all of the weights for tasks 1, ..., $i - 1$ through nonlinear "adapter" connections. Progressive networks are trained one task at a time. When task i is training, only the weights in column i are updated. The weights from previous tasks are frozen and may be referred to by the current column but not updated. See the middle panel in Figure 2 for a graphical depiction of progressive networks, with full details in the original paper.[17]

The major drawback of progressive architectures is that the total number of learnable parameters for task i scales linearly with i due to connections to previously learned weights for tasks 1, ..., $i - 1$. Consequently, for a data set with N tasks, the total memory requirement scales as $O(N^2)$. For pharmaceutical data set collections with potentially hundreds of assays, this memory requirement quickly becomes burdensome. Furthermore, the progressive architecture imposes an ordering upon the available tasks. This ordering is often not meaningful for biochemical data sets. For example, a kinase data set collection may not have a natural notion of which kinase comes first and which comes second in the ordering.

Despite these issues, progressive networks have demonstrated strong results on challenging reinforcement learning and robotics tasks; therefore, we decided to implement and evaluate the effectiveness of progressive architectures for drug discovery data sets.

Bypass Architecture. Progressive architectures allow for each task to have independent nonlinear transformations. The multitask architecture lacks this property, which may render multitask architectures weak on data set collections with very different tasks. At the same time, progressive architectures lack the shared learnable representations that render multitask architectures powerful. Consequently, we experiment with a bypass architecture that merges the per-task indepedent nonlinearities of progressive networks and the shared learnable layers of multitask networks. The bypass architecture has shared joint layers like the multitask architectures but also has an independent column of weights that "bypass" the shared representation for each task. However, unlike the progressive architecture, the independent weights for a given task may not interact with the weights for other tasks. This decision prevents the memory growth of progressive networks and also removes the artificial ordering implicit in progressive architectures. See the rightmost panel in Figure 2 for a depiction of bypass architectures.

Model Evaluation Metrics. Model performance is evaluated by using the squared Pearson correlation coefficient (R^2).[3] For multitask models, we consider the mean Pearson R^2 over all tasks present in the data set. Because the mean performance only gives a rough sense of relative performance of models, we consider other metrics that measure performance against a RF baseline. First, we compute the fraction of learning tasks where performance is improved relative to the RF baseline. Past work has noted that multitask models do not offer consistent improvements (many tasks improve performance, but some do worse as well).[4] Computing the fraction improved allows us to understand the failure modes of deep architectures relative to baseline methods. We also compute the largest per-task R^2 increase and decrease of deep models compared to RF baselines.

These metrics are meant to measure the "robustness" of deep learning architectures. Namely, a robust deep learning architecture should always outperform baseline methods. While this ideal is not achievable with current architectures, our goal is to evaluate current deep architectures for their degrees of robustness. We propose that robustness is a general principle for proposed architectures and should be considered in future algorithmic work as a guiding principle.

We also consider intertask correlations within the raw data sets. Previous work has indicated that multitask learning leverages correlations between tasks in a data set collection. To provide a rough measure of intertask correlation, we plot histograms of all pairwise Pearson R^2 scores between tasks in a data set collection. Histograms with large peaks at 0 indicate that many tasks are uncorrelated with one another; histograms with large peaks at positive values indicate stronger intertask correlations. Note that this visualization is only meaningful for dense data sets; every data point has to have a label for every task.

EXPERIMENTAL RESULTS

In this section, we report the performance of RFs, multitask networks, progressive networks, and bypass networks trained on the Kaggle, Factors, Kinase, and UV data set collections. Model hyperparameters were tuned on validation sets with a combination of manual hyperparameter tuning and random hyperparameter search.[26]

Kaggle. The Kaggle collection serves as a proof of concept that our contributed multitask DeepChem implementation is capable of recapitulating results from the literature.[3] Table 2 reports mean train, validation, and test R^2 values for all models on the Kaggle collection. Notably, the singletask and multitask deep architectures achieve performance quantitatively similar to that from previously reported work, differing only by a couple percentage points (but within previously reported performance ranges for deep networks on this collection). The multitask

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

Table 2. Model Performance of the Kaggle Collection[a]

	mean training R^2	mean validation R^2	mean test R^2
multitask	0.793 ± 0.005	0.467 ± 0.014	0.468 ± 0.012
progressive	0.887 ± 0.015	0.436 ± 0.022	0.432 ± 0.016
bypass	0.843 ± 0.004	0.454 ± 0.012	0.456 ± 0.012
singletask	0.942 ± 0.002	0.450 ± 0.009	0.448 ± 0.009
RF	0.941 ± 0.000	0.423 ± 0.008	0.428 ± 0.007

[a]All experiments were repeated three times with a different choice of random seed. Standard deviations are means across per-task standard deviations over trials.

deep networks offer the strongest validation and test performance of all models tested on the Kaggle collection.

All models overfit the training sets, with mean train R^2 significantly higher than mean validation and test R^2. Surprisingly, the multitask networks overfit the least, suggesting that the multitask effect acts partially as a regularizer controlling the degree of overfitting. By comparison, both singletask deep networks and RFs have considerably higher training R^2, suggesting a greater degree of overfit compared to multitask models. The progressive networks offer comparable performance to RFs but seem not to perform as well as other deep architectures. The bypass networks behave more similarly to multitask architectures, with strong validation and test performance but with greater overfitting on the training set, suggesting that the independent bypass layers may harm as well as help.

Table 3 evaluates the robustness of all deep learning models compared to RF baselines. Multitask networks perfomed quite

Table 3. Performance Relative to the RF Baseline on Kaggle Test Sets

	fraction improved (vs RF)	largest task R^2 drop	largest task R^2 gain
multitask	11/15	−0.055	0.133
progressive	9/15	−0.153	0.130
bypass	9/15	−0.050	0.157
singletask	9/15	−0.117	0.141

robustly with 11 of 15 tasks improved and a worst-case R^2 drop of $-0.055R^2$. The bypass, singletask, and progressive networks all succeeded in improving 9 of 15 tasks. The progressive and singletask networks suffer more severe per-task R^2 drops. Figure 3 plots the per-task improvement for each model on the Kaggle collection.

Factors. The Factors collection measures IC50 of inhibition for 12 serine proteases. The measurements for the Factors collection are dense; each molecule is measured in every assay. Figure 4 reports the histogram of correlations between various Factors tasks. While many Factors tasks are effectively uncorrelated, a large proportion of tasks have moderate to high intertask correlations.

Results for trained models on this collection are reported in Table 4 and visualized in Figure 5. The multitask networks achieve the best mean validation and test R^2, with the bypass models following closely behind. The Factors collection has only 1500 compounds, compared to the 100 thousand in the Kaggle collection; therefore, perhaps unsurprisingly, all models overfit much more heavily than for the Kaggle collection. The multitask effect does not seem to regularize performance on this collection as effectively.

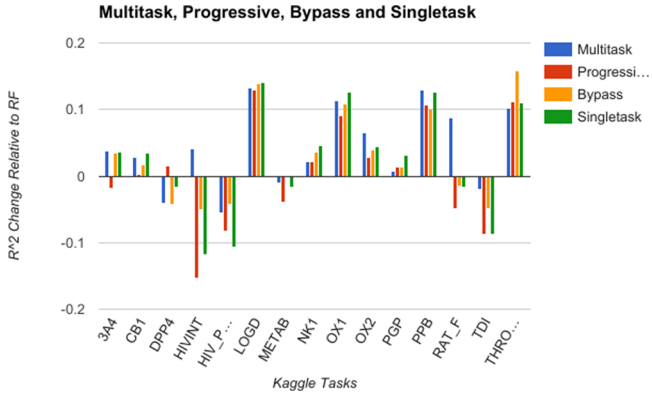


Figure 3. Change in performance relative to RFs on the Kaggle collection.

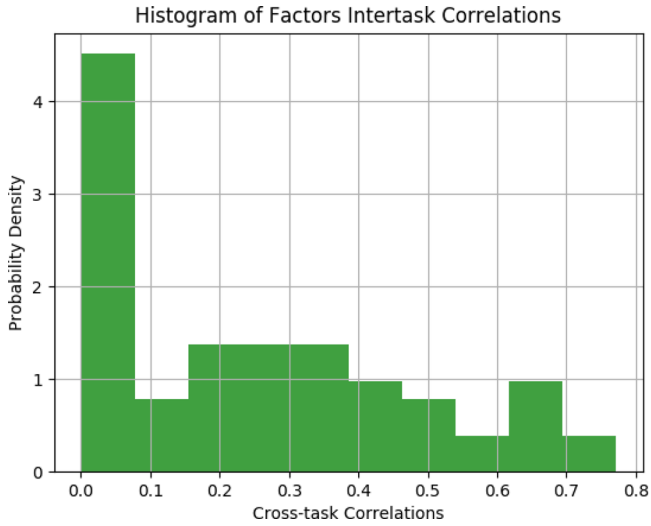


Figure 4. Histogram of correlations between Factors tasks. The Pearson R^2 is computed for each pair of tasks in the collection. The histogram displays all computed correlations.

Table 4. Model Performances of the Factors Collection[a]

	mean training R^2	mean validation R^2	mean test R^2
multitask	0.941 ± 0.003	0.517 ± 0.014	0.424 ± 0.006
progressive	0.935 ± 0.001	0.499 ± 0.001	0.409 ± 0.005
bypass	0.949 ± 0.001	0.508 ± 0.004	0.410 ± 0.001
singletask	0.954 ± 0.000	0.504 ± 0.005	0.393 ± 0.003
RF	0.949 ± 0.000	0.466 ± 0.005	0.412 ± 0.006

[a]All experiments were repeated three times with a different choice of random seed. Standard deviations are means across per-task standard deviations over trials.

Table 5 provides measurements of robustness for the deep models relative to the RF baseline, and Figure 5 provides a graphical representation of model improvement. Perhaps due to the large number of tasks with low correlations, the multitask networks succeed in improving only 5 of 12 tasks. However, tasks that are improved have large boosts in accuracy, while the worst-case drop of $-0.064R^2$ is relatively small. The progressive, singletask, and bypass models each improve 4 of 12 assays but have worst-case drops larger than that for the multitask models.

Kinase. The Kinase collection consists of 2500 compounds tested against 99 kinase assays. This collection like Factors (and

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

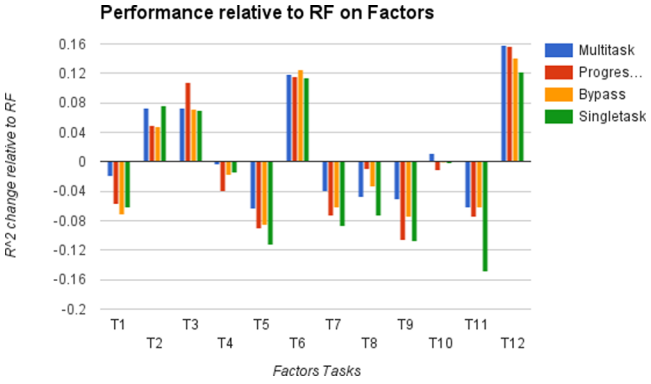


Figure 5. Change in performance relative to RFs on the Factors collection.

Table 5. Performance Relative to the RF Baseline on Factors Test Sets[a]

	fraction improved (vs RF)	largest task R^2 drop	largest task R^2 gain
multitask	5/12	−0.064	0.158
progressive	4/12	−0.108	0.157
bypass	4/12	−0.087	0.141
singletask	4/12	−0.150	0.122

[a]All experiments were repeated three times with a different choice of random seed. Standard deviations are means across per-task standard deviations over trials.

unlike Kaggle) is dense, with all molecules measured in all assays. Figure 6 reports the histogram of correlations between various Kinase tasks, displaying that most tasks have moderate to strong correlations with other tasks in the collection.

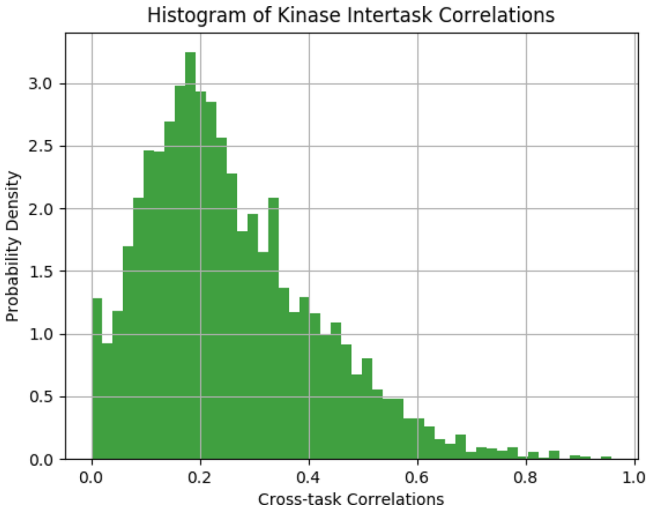


Figure 6. Histogram of correlations between Kinase tasks. The Pearson R^2 is computed for each pair of tasks in the collection. The histogram displays all computed correlations.

Table 6 reports the mean train, validation, and test R^2 of trained models on this collection, and Figure 7 displays graphical performance relative to the RF baseline. Multitask models demonstrate the strongest performance on both validation and test sets. Progressive networks again achieve performance similar to that of RFs, with the bypass and singletask models offering performance between RF and

Table 6. Model Performance on the Kinase Collection[a]

	mean training R^2	mean validation R^2	mean test R^2
multitask	0.897 ± 0.002	0.301 ± 0.012	0.238 ± 0.010
progressive	0.833 ± 0.012	0.284 ± 0.006	0.207 ± 0.001
bypass	0.941 ± 0.003	0.285 ± 0.009	0.224 ± 0.008
singletask	0.951 ± 0.004	0.284 ± 0.011	0.229 ± 0.010
RF	0.941 ± 0.001	0.263 ± 0.011	0.208 ± 0.001

[a]All experiments were repeated three times with a different choice of random seed. Standard deviations are means across per-task standard deviations over trials.

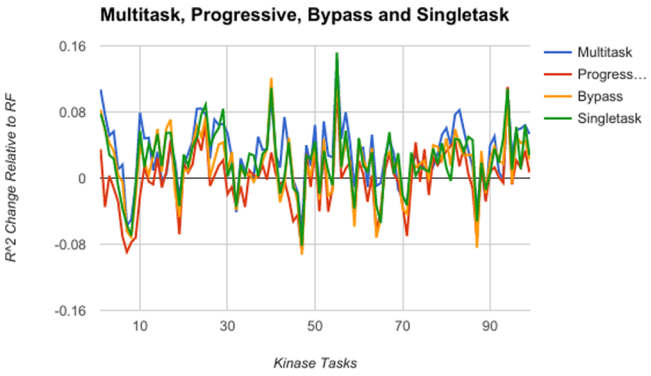


Figure 7. Change in performance relative to RFs on the Kinase collection. Separate lines for each model type are used instead of bars due to the large number of tasks.

multitask architectures. Given the greater number of compounds, the multitask networks once again overfit less severely than singletask deep networks and RFs. Notably, progressive networks appear to underfit this data set and achieve the lowest training R^2. This underfit is likely due to memory issues; to handle training of progressive networks with 99 tasks (recall that memory usage scales as $O(N^2)$, where N is the number of tasks), we limited the number of hidden nodes per task (per layer) to 50, which may have reduced the models' explanatory capacity.

Table 7 evaluates the robustness of the deep learning methods relative to the RF baseline. The multitask networks are

Table 7. Performance Relative to the RF Baseline on Kinase Test Sets

	fraction improved (vs RF)	largest task R^2 drop	largest task R^2 gain
multitask	79/99	−0.064	0.144
progressive	53/99	−0.089	0.111
bypass	74/99	−0.093	0.131
singletask	78/99	−0.082	0.152

quite robust on this collection, providing improvements for 79 of the 99 assays. This broad improvement may be due to the fact that kinase inhibitors are often promiscuous across many kinases,[27] suggesting that shared representations can be easily extracted. (Figure 6 supports this assertion.) The worst-case performance drop of $-0.064R^2$ for multitask models is moderate, suggesting that multitask methods are strong models for Kinase systems. The bypass and singletask models also succeed in offering broad improvement, respectively for 74/99 and 78/99 assays, but with slightly larger worst-case drops of $-0.089R^2$ and $-0.082R^2$, respectively. The progressive models

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

only improve 53/99 assays relative to RFs, possibly due to the model's inability to exploit intertask correlations.

UV. The UV data set collection consists of roughly 10,000 compounds evaluated over 190 absorption wavelengths (210–400 nm). The UV collection (like the Factors and Kinase collections) is dense, with all molecules measured at all wavelengths. Figure 8 reports the histogram of correlations

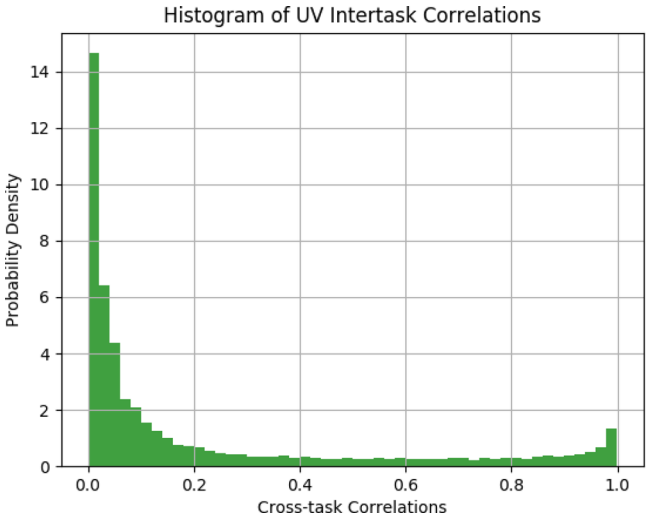


Figure 8. Histogram of correlations between UV tasks. Pearson R^2 is computed for each pair of tasks in the collection. The histogram displays all computed correlations.

between various UV tasks. Unlike other collections, most assays are weakly correlated. The small bump at high correlation is due to fact that absorption at nearby wavelengths is highly correlated (absorption at 231 and 232 nm likely have high correlation for example).

Table 8 reports mean train, valid, and test R^2 values for models on this data set collection. Multitask models achieve the

Table 8. Model Performance on the UV Collection[a]

	mean training R^2	mean validation R^2	mean test R^2
multitask	0.961 ± 0.001	0.420 ± 0.013	0.441 ± 0.014
bypass	0.903 ± 0.005	0.396 ± 0.014	0.421 ± 0.014
singletask	0.968 ± 0.002	0.399 ± 0.012	0.417 ± 0.013
RF	0.961 ± 0.000	0.404 ± 0.006	0.428 ± 0.006

[a]All experiments were repeated three times with a different choice of random seed. Standard deviations are means across per-task standard deviations over trials. Progressive networks were omitted because models took over a day to train.

highest mean validation and test R^2 but only by a slight margin. Note that Table 8 does not provide numbers for progressive networks. We could not successfully train progressive models on this collection within a day, likely due to the high graph complexity for a progressive architecture with nearly 200 tasks.

Table 9 evaluates the robustness of trained models relative to the RF baseline. Notably, multitask models do not succeed in outperforming the RF baseline on a majority of assays, with only 81 of the 190 data sets improved. As with Factors, the large number of uncorrelated tasks may have contributed to this outcome. The largest worst-case R^2 drop is moderate, with a $-0.111R^2$ drop observed on one task, but with large performance gains of 0.199 on the best-improved task. Figure

Table 9. Performance Relative to the RF Baseline on UV Test Sets

	fraction improved (vs RF)	largest task R^2 Drop	largest task R^2 gain
multitask	81/190	−0.111	0.199
bypass	60/190	−0.139	0.219
singletask	63/190	−0.122	0.177

9 shows an interesting pattern, with most assays underperforming the RF baseline but with a large fraction showing

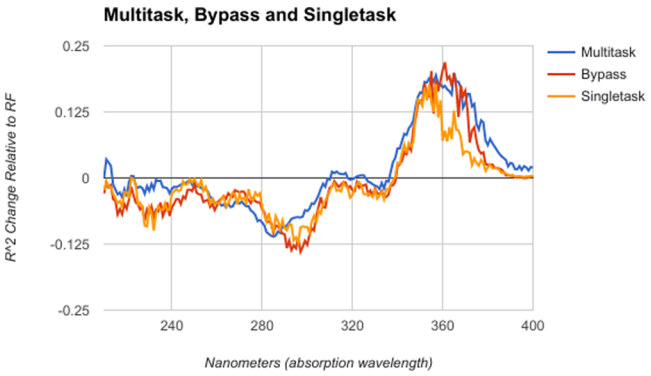


Figure 9. Change in performance relative to RFs on the UV collection. Separate lines for each model type are used instead of bars due to the large number of tasks.

significant improvements over the RF baseline. The improved assays are all adjacent, indicating that the multitask models may perhaps be learning some shared property of the physics at these wavelengths.

Singletask versus Multitask. To evaluate the relative improvement due to the multitask effect, as opposed to changes in performance due to deep learning, we evaluated the fraction of tasks where multitask models outperformed singletask models for all collections that we considered. Results are displayed in Table 10. Notably, a majority of assays in all

Table 10. Number of Tasks Improved by Multitasking

data set collection	multitask improved
Kaggle	8/15
Factors	11/12
Kinase	64/99
UV	155/190

collections show improvement of multitask over singletask, indicating that multitask deep learning performs better on our assay collections than singletask deep learning.

DISCUSSION AND CONCLUSION

Preliminary studies have demonstrated that multitask deep learning can offer performance improvements over simpler machine-learning techniques for drug discovery data sets.[3,4] However, multitask deep networks have yet to achieve a wide degree of adoption across pharmaceutical and biotechnology industries. Historically, transforming machine-learning prototypes into production systems can be challenging. As noted previously, Netflix decided not to use the winning entry in its million dollar recommendation challenge to serve actual customer recommendations.[7] We hypothesize that two key

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068–2076

barriers prevent the widespread adoption of multitask deep networks: First, implementing deep architectures can be a challenging software endeavor. Second, the failure modes of multitask deep networks relative to RF and singletask baselines remain poorly understood.

This work has aimed to overcome both barriers to entry for deep architectures. To overcome software challenges, we have introduced a high-quality open-source implementation of multitask deep networks as part of the DeepChem library for open-source drug discovery.[14] Our implementation makes it easy to build, fit, and evaluate high-quality deep learning models with simple python scripts (see Figure 1 for a real code example). We also contribute open-source implementations of two new deep learning architectures to DeepChem. Progressive networks[17] were first proposed for robust multitask deep learning in robotic and reinforcement learning applications. We also propose a bypass architecture, which aims to serve as a hybrid of multitask and progressive architectures.

We used our open-source DeepChem implementation to test the behavior of our deep architectures across four broad collections of data sets. We demonstrated that the DeepChem implementation can duplicate qualitatively and quantitatively the performance of deep models on the Kaggle collection analyzed previously in the literature.[3] We then evaluated the performance of multitask, singletask, progressive, bypass, and RFs across the Factors, Kinase, and UV data set collections. In particular, because our validation and test splits use time and neighbor splits (known to be challenging for multitask methods), our evaluation constitutes a "worst-case" test of multitask performance. Even under these challenging conditions, multitask models offer the strongest mean validation and test set R^2 across all evaluated collections.

To further evaluate model performance, we introduced a set of robustness measurements, which measure the fraction of assays improved by deep architectures relative to RF baselines. We also considered the worst-case and best-case per-task R^2 improvement over all tasks. Multitask architectures are more robust than progressive, singletask, and bypass architectures on these data sets, with a majority of assays improved over RF on the Kaggle and Kinase data sets. On the Factors and UV data sets, fewer assays are improved, likely due to lower intertask correlations, but some tasks see quite significant improvements and only moderate performance drops elsewhere. Worst-case performance drops are lowest for multitask architectures compared to other deep architectures. Broadly, our robustness analysis suggests that multitask deep models can offer improvements over RFs for a wide variety of models and especially in collections with high intertask correlations. Our work suggests that multitask architectures can be fruitfully applied to many data sets that arise in drug discovery.

In general, we note that while multitask deep learning provides improvements over RF methods on average many assays suffer worse performance compared with the RF baseline. While the full investigation of these failures is left to future work, we note that the use of obfuscated fingerprints that obscure molecular structures may be limiting the potential of deep learning methods on these data sets. Subsequent work by some of the authors of the present paper has shown that graph convolutional models[28] implemented in DeepChem can achieve significant boosts in performance over multitask deep networks on the MoleculeNet benchmark suite.[29] In the future, it may be interesting to test these graph convolutional methods in collaboration projects on pharmaceutical data. The DeepChem library will hopefully serve to enable such research.

We note that in this paper we did not attempt to place error bars on the predictions made for individual molecules. The presence of good error bars allows practitioners to reason about the domain of applicability for learned models. However, gauging the confidence of predictions from deep learning models is known to be challenging. There has been significant recent interest in the methods of Bayesian deep learning,[30] which may provide a structured solution to the uncertainty modeling problem. Extending Bayesian deep learning methods to pharmaceutical applications is beyond the scope of the present work, however, and we suggest it as a good topic for future work.

Our open-source multitask deep network implementation in DeepChem can help facilitate the broad adoption of deep networks in commercial drug discovery. In particular, we open source our deep learning models for all collections in this paper (along with the associated data sets) to serve as a community resource. We anticipate that our open-source implementation in tandem with our statistical robustness analysis across a broad collection of data sets will facilitate the widespread adoption of multitask deep learning in the pharmaceutical industry.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acs.jcim.7b00146.

Details about how machine learning models were trained and architectural details about models for Kaggle, Factors, Kinase, and UV architectures (PDF)

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: pande@stanford.edu.

ORCID

Bharath Ramsundar: 0000-0001-8450-4262

Bowen Liu: 0000-0003-0609-3087

Robert P. Sheridan: 0000-0002-6549-1635

Notes

The authors declare the following competing financial interest(s): A.V., M.T., and R.P.S. are employees of Merck & Co., Inc. of Kenilworth, NJ. V.S.P. is a consultant and SAB member of Schrodinger, LLC and Globavir, sits on the Board of Directors of Omada Health, and is a General Partner at Andreessen Horowitz.

■ ACKNOWLEDGMENTS

We would like to thank the Stanford Computing Resources for providing us with access to the Sherlock and Xstream GPU nodes. Thanks to Junshui Ma for help in debugging performance on the Kaggle collection. Thanks to Kevin McCloskey and Patrick Riley from Google for open-sourcing the code that formed the preliminary DeepChem multitask implementation. Thanks to Aarthi Ramsundar for help with diagram construction. The Pande Group is broadly supported by grants from the NIH (R01 GM062868 and U19 AI109662) as well as gift funds and contributions from Folding@home donors. We acknowledge the generous support of Dr. Anders G. Frøseth for our work on machine learning. B.R. was supported by the Fannie and John Hertz Foundation.

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076

REFERENCES

(1) LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436−444.

(2) Deep Learning How I Did It: Merck 1st Place Interview. http://blog.kaggle.com/2012/11/01/deep-learning-how-i-did-it-merck-1st-place-interview/ (accessed Dec 10, 2016).

(3) 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.

(4) Ramsundar, B.; Kearnes, S.; Riley, P.; Webster, D.; Konerding, D.; Pande, V. Massively Multitask Networks for Drug Discovery. *arXiv:1502.02072* 2015.

(5) Unterthiner, T.; Mayr, A.; Ünter Klambauer, G.; Steijaert, M.; Wenger, J.; Ceulemans, H.; Hochreiter, S. Deep Learning as an Opportunity in Virtual Screening. *Deep Learning and Representation Learning Workshop (NIPS 2014)*; 2014.

(6) Kearnes, S.; Goldman, B.; Pande, V. Modeling Industrial ADMET Data with Multitask Networks. *arXiv:1606.08793* 2016.

(7) Netflix Never Used Its $1 Million Algorithm Due to Engineering Costs. http://arstechnica.com/gadgets/2012/04/netflix-never-used-its-1-million-algorithm-due-to-engineering-costs/ (accessed Dec 10, 2016).

(8) Netflix Recommendations Beyond the 5 stars (Part 1). http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html (accessed Dec 10, 2016).

(9) Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; Kudlur, M.; Levenberg, J.; Monga, R.; Moore, S.; Murray, D.; Steiner, B.; Tucker, P.; Vasudevan, V.; Warden, P.; Wicke, M.; Yu, Y.; Zheng, X. TensorFlow: A System for Large-Scale Machine Learning. *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Savannah, GA, 2016; pp 265−283.

(10) Bastien, F.; Lamblin, P.; Pascanu, R.; Bergstra, J.; Goodfellow, I.; Bergeron, A.; Bouchard, N.; Warde-Farley, D.; Bengio, Y. Theano: New Features and Speed Improvements. *arXiv:1211.5590* 2012.

(11) Chollet, F. *Keras: Deep Learning Library for Theano and Tensorflow*; 2015.

(12) Weininger, D. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Model.* **1988**, *28*, 31−36.

(13) Bemis, G. W.; Murcko, M. A. The Properties of Known Drugs. 1. Molecular Frameworks. *J. Med. Chem.* **1996**, *39*, 2887−2893.

(14) DeepChem: Deep-learning models for Drug Discovery and Quantum Chemistry. https://github.com/deepchem/deepchem (accessed Dec 10, 2016).

(15) Subramanian, G.; Ramsundar, B.; Pande, V.; Denny, R. A. Computational Modeling of β-secretase 1 (BACE-1) Inhibitors using Ligand Based Approaches. *J. Chem. Inf. Model.* **2016**, *56*, 1936−1949.

(16) Altae-Tran, H.; Ramsundar, B.; Pappu, A., S.; Pande, V. Low Data Drug Discovery with One-Shot Learning. *ACS Cent. Sci.* **2017**, *3*, 283−293.

(17) Rusu, A. A.; Rabinowitz, N. C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; Hadsell, R. Progressive Neural Networks. *arXiv:1606.04671* 2016.

(18) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, É. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR)* **2011**, *12*, 2825−2830.

(19) Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: a Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)* **2014**, *15*, 1929−1958.

(20) Kingma, D.; Ba, J. A Method for Stochastic Optimization. *arXiv:1412.6980* 2014.

(21) Carhart, R. E.; Smith, D. H.; Venkataraghavan, R. Atom Pairs as Molecular Features in Structure-Activity Studies: Definition and Applications. *J. Chem. Inf. Model.* **1985**, *25*, 64−73.

(22) Kearsley, S. K.; Sallamack, S.; Fluder, E. M.; Andose, J. D.; Mosley, R. T.; Sheridan, R. P. Chemical Similarity using Physiochemical Property Descriptors. *J. Chem. Inf. Model.* **1996**, *36*, 118−127.

(23) 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.

(24) Breiman, L. Random Forests. *Machine learning* **2001**, *45*, 5−32.

(25) Svetnik, V.; Liaw, A.; Tong, C.; Culberson, J. C.; Sheridan, R. P.; Feuston, B. P. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *J. Chem. Inf. Model.* **2003**, *43*, 1947−1958.

(26) Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research (JMLR)* **2012**, *13*, 281−305.

(27) Anastassiadis, T.; Deacon, S. W.; Devarajan, K.; Ma, H.; Peterson, J. R. Comprehensive Assay of Kinase Catalytic Activity Reveals Features of Kinase Inhibitor Selectivity. *Nat. Biotechnol.* **2011**, *29*, 1039−1045.

(28) 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. *Neural Inf. Proc. Sys. (NIPS)* **2015**, 2224−2232.

(29) Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; Pande, V. MoleculeNet: A Benchmark for Molecular Machine Learning. *arXiv:1703.00564* 2017.

(30) Kendall, A.; Gal, Y. What Uncertainties do We Need in Bayesian Deep Learning for Computer Vision? *arXiv:1703.04977* 2017.

DOI: 10.1021/acs.jcim.7b00146
J. Chem. Inf. Model. 2017, 57, 2068−2076