JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

pubs.acs.org/jcim

Application Note

SolTranNet—A Machine Learning Tool for Fast Aqueous Solubility Prediction

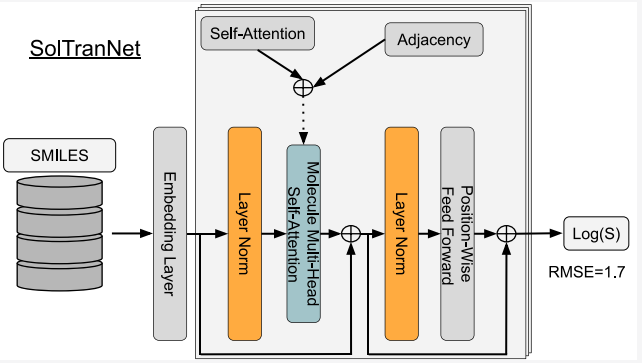
Paul G. Francoeur* and David R. Koes

Cite This: J. Chem. Inf. Model. 2021, 61, 2530−2536

Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: While accurate prediction of aqueous solubility remains a challenge in drug discovery, machine learning (ML) approaches have become increasingly popular for this task. For instance, in the Second Challenge to Predict Aqueous Solubility (SC2), all groups utilized machine learning methods in their submissions. We present SolTranNet, a molecule attention transformer to predict aqueous solubility from a molecule's SMILES representation. Atypically, we demonstrate that larger models perform worse at this task, with SolTranNet's final architecture having 3,393 parameters while outperforming linear ML approaches. SolTranNet has a 3-fold scaffold split cross-validation root-mean-square error (RMSE) of 1.459 on AqSolDB and an RMSE of 1.711 on a withheld test set. We also demonstrate that, when used as a classifier to filter out insoluble compounds, SolTranNet achieves a sensitivity of 94.8% on the SC2 data set and is competitive with the other methods submitted to the competition. SolTranNet is distributed via PIP, and its source code is available at https://github.com/gnina/SolTranNet.



INTRODUCTION

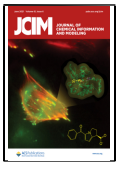
Aqueous solubility is an important physicochemical property for drug discovery, in part due to humans being 60% water[1] and water being the primary solvent in most assays. Besides absorption into the body, a drug's behavior in water is linked to its distribution and elimination in the body. Consequently, lack of aqueous solubility can potentially result in failure throughout the drug discovery pipeline.[2−4] Ideally, one would directly measure the solubility of a given compound. However, such an approach is slow and expensive and requires the compound to be available for experiments. With the increasing size of molecular screening libraries, up to 350 million compounds,[5] experimentally measuring the solubility of each compound becomes infeasible. Thus, there is a need for fast and accurate solubility prediction as an accessory to large-scale virtual screening.

Predicting aqueous solubility for a given molecule is typically performed with one of three methods: molecular simulation, quantum calculations, or an empirically fit function. Molecular simulation approaches utilize statistical mechanics and either directly calculate the solubility from the chemical potentials of the solute and water[6] or directly simulate the solute in explicit water molecules. For direct simulation of the solute, there are several methods available, as reviewed by Skyner et al.,[7] which all use a large amount of computing power and, in the case of direct simulation, also require a long time to reach equilibrium.

Quantum mechanics (QM)-based approaches operate at a higher level of theory than simulation and are divided into two broad categories based on if the solvent is included in the calculation. Full QM methods include the water molecules in their calculations and are based on density functional theory.[7] This approach is the most rigorous yet suffers from underestimating the equilibrium density of the solute[7] and requires the most computing power. The other approach, continuum solvent methods, treats the water as a bulk dielectric. This saves on computing power but does not sample the water's degrees of freedom and assumes that the solute's charge is entirely contained in the cavity it creates in the solvent.

Both of the previous methods require a large amount of computing power in order to perform their calculations. In order to avoid this, there has been extensive work in developing empirical models for solubility prediction.[7,8] These methods utilize some set of molecules with known solubility as training data and then fit a function based on features of said molecules to predict solubility. This approach is much faster but fails to generalize to molecules outside of the scope of the training set. As in many other applications,

Received: March 19, 2021
Published: May 26, 2021



https://doi.org/10.1021/acs.jcim.1c00331
J. Chem. Inf. Model. 2021, 61, 2530−2536

nonparametric machine learning (ML) approaches have been supplanting traditional function fitting for this empirical approach. Indeed, all of the approaches in the Second Challenge to Predict Aqueous Solubility (SC2) utilized ML algorithms.[9] Boobier et al.[10] showed typical ML algorithms performed equally as well as human experts for predicting the solubility of drug-like compounds. Lovric et al.[11] performed a study of several ML methods and showed that simpler ML approaches generalized better to unseen data. Lastly, Cui et al.[12] showed deeper models can succeed at this task, with their 20-layer ResNet architecture.

Maziarka et al.[13] developed the molecule attention transformer (MAT) architecture, which is modeled after the current state of the art transformer architecture for natural language processing tasks. MAT functions by applying self-attention to a molecular graph representation of the molecule, where each node is characterized by a feature vector as described in Table S3. This feature vector is combined with an adjacency matrix describing the connectivity of the molecule and a distance matrix that describes how far apart each atom is from each other atom in a generated 3D conformer of the molecule. The authors utilized MAT for a variety of molecular property prediction tasks and performed well at solubility prediction without optimizing for this task.

While ML approaches have become more common for predicting aqueous solubility, there is still a lack of readily available tools. Extensive code bases, such as DeepChem[14] and Chemprop,[15] contain training data and support for predicting aqueous solubility, but it remains the burden of the user to train new models if they wish to use the software. Similarly, while there were 37 participants in SC2, the provided information is not enough to recreate the models that were used to generate their predictions. For example, we know the general architecture (e.g., artificial neural networks) and training data that a particular submitter used but do not know the hyperparameters of the architecture nor how long the model was trained. As such, there is an unmet need for an ML-based solubility predictor that is easy to deploy and use.

Here, we present SolTranNet, an ML model based on the MAT architecture, for predicting aqueous solubility. We trained SolTranNet utilizing the SMILES, reported solubilities in AqSolDB,[16] as it was the largest publicly available curated set, and optimized SolTranNet for speed and quality of prediction. SolTranNet has 0.6764 R^2 and 1.459 root-mean-square error (RMSE) on clustered cross-validation scaffold-based splits of AqSolDB and 0.577 R^2 and 1.711 RMSE on our withheld test set when trained on all of AqSolDB. We also compare to other recently published ML models.[9−12] SolTranNet is available via PIP installation for PYTHON3, and the source code is available at https://github.com/gnina/SolTranNet. The data sets and scripts used for this paper are available at https://github.com/francoep/SolTranNet_paper.

METHODS

Here, we describe the installation process of SolTranNet, as well as its command line and Python API. We also describe the data set and hyperparameter sweep utilized to fit and select the final architecture of SolTranNet. Additionally, we describe the four external data sets we used to validate SolTranNet's performance against current methods.

Installation and Usage. We aim for SolTranNet to be easy to incorporate into a virtual screening pipeline. As such, we have made the entire package PIP installable for PYTHON3, which will allow for both a command line utility and usage in a PYTHON3 environment. SolTranNet's dependencies are RDKit[17] (2017.09.1+), NumPy[18] (1.19.3), PyTorch[19] (1.7.0+), and pathlib (1.0+). SolTranNet also supports CUDA enabled GPU acceleration via automatic detection through PyTorch. Installation is done through PIP

```
python3 -m pip install soltrannet
```

which installs a standalone command line utility

```
soltrannet mysmiles.txt predictions.txt
```

as well as a Python module

```
>>> import soltrannet as stn
>>> predictions = list(stn.predict(["c1ccccc1","Cn1cnc2n(C)c(=O)n(C)c(=O)c12"]))
```

for embedding in user defined scripts.

Data Sets. AqSolDB[16] is the data set we utilized for training SolTranNet, as it was the largest publicly available set. AqSolDB spans a wide range of solubility values (Figure S9) and is collated from differing data sets, most of which controlled for temperature. Notably, during this collation process, AqSolDB only screened for identical molecules and did not verify if the solubility measurements in its data sets were measured in buffered conditions or water or at what pH the measurement was taken. This is especially noteworthy as these differing conditions can change the measurement by orders of magnitude. However, an expansive solubility data set controlled for all these factors is not currently available. Additionally, from our previous work,[20] we have observed that neural network models tend to perform better with larger data sets, even if the data contains more noise. Thus, we elected to utilize AqSolDB for our training set over the smaller ESOL[21] which was used in the original MAT publication,[13] a comparative analysis of which can be found in Figures S7 and S8 and Table S6. Notably, while there are many other features available in AqsolDB, we only utilized the included SMILES strings and the reported solubilities (log S, S in mol/L). We then utilized RDkit[17] to calculate the Murcko scaffolds of the molecules, in order to cluster the molecules and generate a 3-fold clustered cross-validation (CCV) split of the data. We additionally created a withheld test set out of the molecules present in the SC2 test sets[9] and FreeSolv,[22] such that no molecule in the withheld test set has an RDkit fingerprint similarity of greater than or equal to 0.9 to any molecule in AqSolDB. The histograms of solubility values for each fold of the scaffold split, the full AqSolDB, and our withheld set are shown in Figure S3. These sets were utilized to optimize SolTranNet's final architecture, as described in the Model Optimization subsection.

While the data sets described above allow for an evaluation of SolTranNet, we also seek to compare to other published models. Thus, we also evaluate SolTranNet using previously published training and test sets. Boobier et al.[10] provided a training set and a testing set (75 and 25 molecules, respectively) in 2017 that showed a multilayer perceptron performing equally as well as human experts. Cui et al.[12] provided a training set of 9943 molecules and a testing set of 62 molecules that they utilized to evaluate a 20-layer ResNet-based ML model. Lovric et al.[11] provided a set of 829 molecules which they randomly split into a training, validation, and testing set consisting of 64%, 16%, and 20% of the molecules, respectively, and evaluated the performance of random forest, light gradient boosting method, and LASSO and partial least-squares regression models. Lastly, Llinas et al.[9]

https://doi.org/10.1021/acs.jcim.1c00331
J. Chem. Inf. Model. 2021, 61, 2530−2536

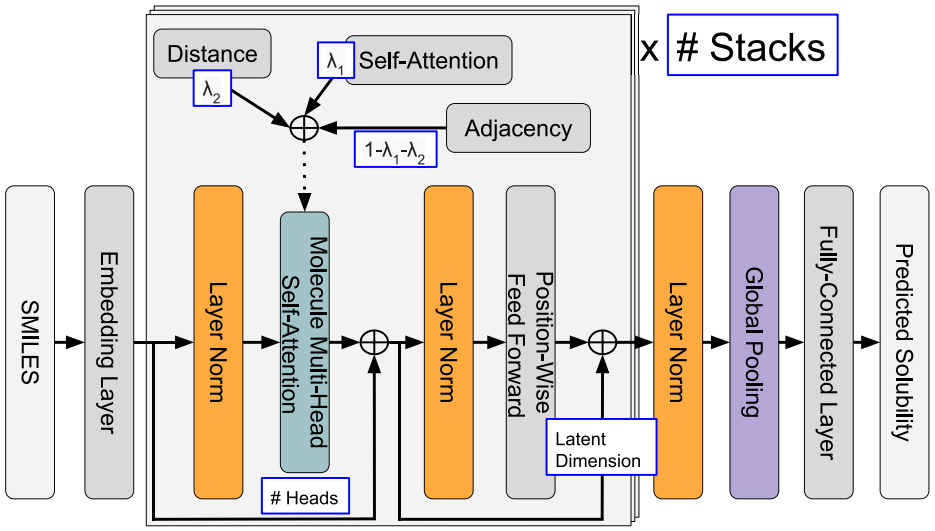


Figure 1. General architecture of SolTranNet. Each item in a blue box is a tuned hyperparameter, whose values were tested as shown in Figure S1b.

hosted the SC2, wherein two test sets were provided and participants could utilize any training set they desired, provided that no molecule was present in the test sets. As such, we filtered AqSolDB such that there was no overlap with any molecule present in the SC2 test set (determined by identical RDkit fingerprints) to use as a new training set.

Model Optimization. The general architecture of the MAT model is provided in Figure 1. In order to optimize the hyperparameters for SolTranNet, we utilized a two-stage optimization procedure (Table S1). All hyperparameter optimizations were performed utilizing the Weights and Biases platform.[23] The first stage was a Bayesian with a hyperband stopping criteria search over hyperparameters related to the model optimizer (Figure S1a). The objective of this search was to minimize the RMSE of the test set for the first fold of the CCV scaffold split of AqSolDB. This resulted in the selection of the Huber loss function, the stochastic gradient descent (SGD) optimizer with a momentum of 0.06, no weight decay, and a learning rate of 0.04. We then performed a grid search over the hyperparameters describing the SolTranNet architecture for 100 epochs over each fold of the CCV scaffold split of AqSolDB (Figure S1b). We additionally evaluated the first 10 models of this search with both 2D and 3D distance matrices, after which only the 2D versions of the models were evaluated for the remainder of the sweep. During the grid search, we evaluated one model for each combination of hyperparameters. In order to provide a point of comparison, we evaluated four linear ML models on each fold of the scaffold split of AqSolDB and the full AqSolDB. These linear models are LASSO, Elastic Net, partial least-squares, and ridge regression. Each was implemented through scikit-learn[24] and trained on bit size 2040 RDKit fingerprints for a maximum of 100,000 iterations.

In order to select the best performing model, we calculated the R^2 and the root-mean-square error (RMSE) of the predicted solubility. Since the solubility values in AqSolDB span several orders of magnitude, the R^2 correlation metric is easier to perform well on. Thus, we selected our best performing model by its RMSE performance on our withheld test set (Table S1). Additionally, as we intend for this tool to be deployed for use in very large data sets, we took into account the speed of SolTranNet in evaluating the results of our hyperparameter search.

Final Model Training. The final architecture of SolTranNet utilized the following hyperparameters: 0.1 dropout, 0 lambda distance, 0.5 lambda attention, 2 attention heads, a hidden dimension of size 8, and 8 stacks in the transformer encoding layer. A 2D molecular representation was selected because it provided statistically equivalent results to 3D representations (Figure S4) without the computational burden of generating a 3D conformation or computing a distance matrix. The molecular embedding layer was unchanged from the initial MAT implementation[13] where each atom is represented as a node with a feature vector as shown in Figure S3. We also note that our molecular embedding layer calculates the features for the molecular graph from the RDKit's molecule object of each SMILES, which ensures consistency of results across different SMILES representations. In order to select the final deployed model, we trained five models with different random seeds on all of AqSolDB utilizing the Huber loss function, the SGD optimizer with a momentum of 0.06, no weight decay, and a learning rate of 0.04. We dynamically stopped training after performance on the training set stopped improving for eight epochs. After which, we selected the model architecture with the best average RMSE performance on our withheld test set and selected the best performing model from within that architecture class (Figure S2). The dropout in the final model and our early stopping criteria both help to prevent overtraining of the deployed SolTranNet.

RESULTS

We first show a sampling of results of the hyperparameter sweep for the SolTranNet architecture. We then show the speed benefit of using 2D conformers to process the SMILES string input without a loss of prediction performance. We also analyze the effect of rare atom types on the efficacy of SolTranNet. Lastly, we compare SolTranNet to a variety of other published ML-based solubility models.

Selecting the Final Architecture. To determine the final architecture of SolTranNet, we performed the two-stage hyperparameter search as described in Methods. Table S1 shows the RMSE performance of several models from the

https://doi.org/10.1021/acs.jcim.1c00331
J. Chem. Inf. Model. 2021, 61, 2530−2536

search on the 3-fold scaffold split of AqSolDB. We selected the top RMSE performers of each class of models (by number of parameters) to show. Note that only a single model was evaluated for each combination of hyperparameters. Contrary to expectation, we find that models with fewer parameters tended to perform better at solubility prediction for AqSolDB. This was driven by a better performance on Fold1. It is the most challenging split because the distribution of solubilities in the test set is distinct from the training set (Figure S3). We focus on the RMSE metric, since the large range of true solubilities makes it easier to achieve a higher Pearson R^2 (Table S2).

We quantified SolTranNet's generalization by training five different models on all of AqSolDB and testing them on our withheld set (Table S4). An ensemble of five models outperforms the mean of said models but fails to beat the best performing seed (Figure S2). As we desire SolTranNet to be as fast as possible, we elected to deploy a single model with the best performing seed.

Conformer Generation. SolTranNet first creates the molecular graph representation for each input molecule's SMILES. Part of this representation is the calculation of the distance matrix, which stores the distance between each pair of atoms. In MAT's implementation, this matrix is calculated from a 3D conformer of the molecule generated by RDkit, which can be a costly process. A much computationally cheaper process would be to utilize a 2D conformer of the molecule generated by RDkit, which is the behavior of the original MAT code if a given 3D conformer could not be computed. We show that using a 2D or 3D conformer does not have a statistically significant difference on RMSE or R^2 for the first 10 hyperparameter combinations of SolTranNet during the architecture sweep ($p = 0.144$) (Figure S4). Thus, we used distances determined via RDKit's Compute2DCoords function with default parameters as the 2D conformer for the remainder of the sweep. Since the final architecture does not use distance information ($\lambda_2 = 0$), distance matrix calculation is skipped, accelerating predictions.

Salts and Rare Atom Types. In AqSolDB, salts are explicitly represented and consist of about 11% of the data. Additionally 9.87% of the AqSolDB molecules contain atoms that are typed as "Other" (i.e., not B, N, C, O, F, P, S, Cl, Br, or I) by SolTranNet. These two groups overlap by 75.2% and 83.9%, respectively; that is, unusual atom types are typically due to salts, typically in the counterion. Thus, we investigate the effect of fragmenting salts (by keeping the largest component, thus removing the counterion) and removing successively larger molecules with "Other" typed atoms from the training data (Figures S5 and S6) in order to gain an understanding of the effect rare atom types have on model performance. For all comparisons between identical test sets, there is no statistically significant difference between training on the full salt or the fragmented salt for the RMSE evaluations. We note that when removing molecules containing any "Other" typed atom, training on fragmented salts performed better for the R^2 evaluation when testing with fragmented salts without losing performance on testing with normal salts (Figure S6d). We train on the full salt since it allows less preprocessing for the user. Additionally, removing more "Other" typed molecules from the training and testing data results in performance gains, as expected. Since these "Other" typed atoms are potentially encountered during drug discovery, we kept them in SolTranNet's training set, but have these molecules raise a warning.

Run-Time Performance. To benchmark SolTranNet, we determined the mean time for a solubility prediction from SMILES strings for 1,000 molecules (repeats of the 132 SC2 molecules). Table 1 shows the mean and standard deviation of 10 runs performed with a Nvidia Titan-XP graphics card and using 1 core of an Intel(R) Core(TM) i7-4930K CPU. For the original MAT implementation, utilizing 3D conformer generation takes 34 times longer than 2D conformer generation on GPU (5 times longer on CPU). SolTranNet, with fewer parameters and skipping conformer generation, runs 2.3 times faster on GPU (12.0 times faster on CPU) than MAT utilizing the 2D conformer generation. Additionally, by implementing multiprocessing and running on a single GPU and 12 CPU cores, we are able predict 1 million molecules in 8.5 min.

Table 1. Mean Time for the Model to Predict Solubility for the Original MAT Implementation and SolTranNet (STN)[a]

	mean time (std) [s/mol]	
model	GPU	CPU
MAT 3D	0.1247 (0.002075)	0.1501 (0.001071)
MAT 2D	0.003638 (0.00039)	0.02834 (0.001317)
STN 1 CPU	0.001583 (0.000004)	0.002352 (0.000033)
STN 12 CPU	0.001058 (0.000019)	0.001484 (0.000027)

[a]The predictions were for 1,000 molecules consisting of repeats of the SC2 data set. We report the mean and standard deviation (in parentheses) of 10 runs for each model in seconds per molecule. All predictions were performed on a machine with a Nvidia Titan-XP graphics card and an Intel(R) Core(TM) i7-4930K CPU, with a batch size of 32.

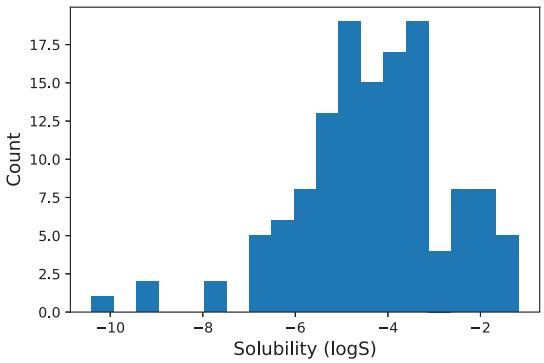
SolTranNet Performance on Other Data Sets. In order to compare SolTranNet to other methods, we first verified SolTranNet's performance on ESOL[21] as in the MAT publication[13] and then evaluated SolTranNet on four recently published solubility training and testing sets[9−12] (Tables 2 and S5). To provide fair comparisons, we evaluate models trained with the provided training and testing data and the deployed version of SolTranNet (trained on AqSolDB). For each comparison, we initially trained five models with the same dynamic stopping criteria as our training. However, for the data sets provided by ESOL,[21] Lovric et al.,[11] and Boobier et al.,[10] this performed poorly and stopped training early due to their smaller size (1128, 530, and 75 molecules, respectively) as compared to our AqSolDB training sets (6655 CCV and 9982 full). As such, we trained new models for 1000, 1000, and 500 epochs, respectively, for those sets. Reassuringly, our implementation of the original MAT architecture and SolTranNet exhibits no statistically significant difference in performance when training with the provided ESOL random splits ($p = 0.073$ and $p = 0.26$, respectively). Notably, the deployed version of SolTranNet has a much worse performance on the ESOL splits (RMSE = 2.99) but maintains a high correlation ($R^2 = 0.890$). See Supplemental Table S6 for further analysis. For the data sets provided by Cui et al.,[12] Lovric et al.,[11] and Boobier et al.,[10] SolTranNet's best performing model trained on the provided data sets achieves similar results to the method reported in the respective paper. However, for the SC2 data,[9] SolTranNet has a much worse

https://doi.org/10.1021/acs.jcim.1c00331
J. Chem. Inf. Model. 2021, 61, 2530−2536

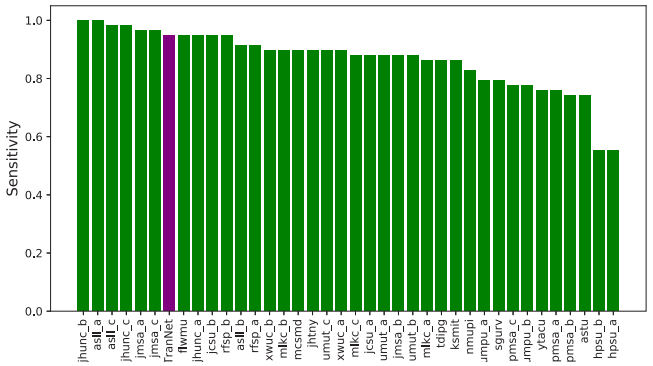
Table 2. RMSE Performance of SolTranNet (STN) on Other Published Data Sets[a]

data set	reported	training	best	deployed	overlap
ESOL MAT	0.278 (0.20)	0.306 (0.027)			1128/1128
ESOL STN	0.278 (0.20)	0.289 (0.011)		2.99 (0.108)	1119/1128
Cui2020	0.681	0.860 (0.215)	0.624	0.813	0/62
Boobier2017	0.985	1.274 (0.178)	1.010	0.845	23/25
Lovric2020	0.720	0.898 (0.101)	0.720	0.854 (0.0672)	151.4/166
Llinas2020 set1	0.780	1.119 (0.163)	0.952	1.004	79/100
Llinas2020 set2	1.060	1.811 (0.328)	1.243	1.295	18/32

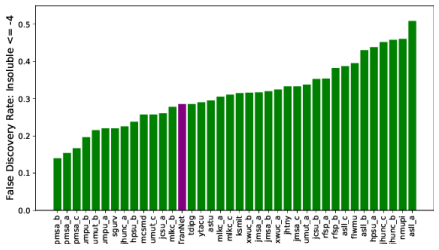
[a]The first two rows are testing the original MAT implementation on ESOL and then testing SolTranNet's implementation. For both the original MAT and SolTranNet implementation, there is no difference between our training and the reported result ($p = 0.073$ and $p = 0.26$, respectively). For SolTranNet Training, we trained five different seeds of our final architecture on the provided training and testing splits and report the mean and standard deviation (in parentheses). The SolTranNet Deployed column is using our final deployed model to predict the provided test set. The final column is the overlap of the provided test set with our deployed model's training set, since there was no attempt to remove molecules present in the test set from the training set. Notably the Lovric set had five different randomly selected splits for training and testing, which is why there is a mean in the Deployed and Overlap columns.



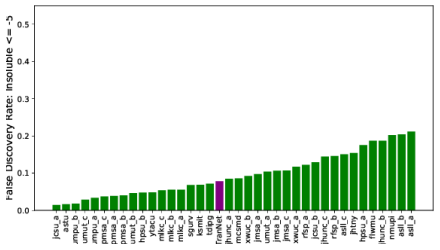
(a) SC2 solubility distribution, bin width of 0.5.



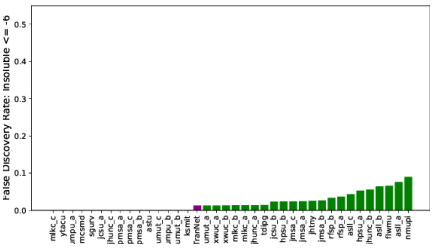
(b) SC2 Sensitivity analysis. SolTranNet is shown in purple.



(c) False discovery rates where insoluble is $LogS <= -4$. SolTranNet is shown in purple.



(d) False discovery rates where insoluble is $LogS <= -5$. SolTranNet is shown in purple.



(e) False discovery rates where insoluble is $LogS <= -6$. SolTranNet is shown in purple.

Figure 2. Recontextualizing the SC2 data. (a) Distribution of both SC2 test sets. We classified solubility as a molecule is soluble if its log $S > -4$. We then calculated the sensitivity (b) and false discovery rates (c–e). SolTranNet performs near state of the art in terms of sensitivity (94.8%) and false discovery rate (1.30% in e).

performance than the top submitted method and ranks in the lower quarter of the submitted methods.

This prompted us to analyze how useful SolTranNet is at classification of soluble compounds, which is a typical use case in a virtual screening pipeline. To refactor the prediction as a classification task, we define a soluble compound as a compound with log $S > -4$, i.e., being able to obtain a 100 μM solution. We then calculated the sensitivity and false discovery rate of SolTranNet and each method submitted to the SC2 competition (Figure 2). For this threshold, SolTranNet has a sensitivity of 94.8% and a false discovery rate of 28.6%. However, as the threshold for misclassification is relaxed (i.e., molecules predicted to have a log $S > -4$ but have an actual log S greater than a relaxed threshold), the false discovery rate quickly drops, with FDRs of 7.8% and 1.3% for -5 and -6 thresholds, respectively (Figure 2). This suggests that SolTranNet is useful at screening out insoluble compounds.

DISCUSSION

We described SolTranNet and evaluated it on a variety of data sets. During model optimization, we show that SolTranNet outperforms linear ML approaches (Table S1) and is competitive with other ML methods (Table 2 and Figure 2). Of particular note, we observe that smaller ML models perform better than their larger counterparts (Table S1). This goes

https://doi.org/10.1021/acs.jcim.1c00331
J. Chem. Inf. Model. 2021, 61, 2530–2536

contrary to the observations of Cui et al.[12] where deeper models performed better. Yet, SolTranNet's best performing model with the same training and testing data achieved a better performance than their 20-layer ResNet architecture[12] (Table 2). We suspect this effect is due to the small training set.

The available solubility data is quite small, with "large data sets" being thousands of data points. Small training set size makes it easier for ML methods to overfit their training data and limits their generalization. We observed this phenomenon with our larger models during our hyperparameter sweep (Table S1). This is why we selected our final architecture by its performance on our withheld test set, and why models with dropout tended to perform better as both of these techniques help to reduce overfitting to the training set. We hypothesize that the smaller models are more suited to solubility prediction until more training data becomes available. Data augmentation is a potential approach to expand the available data, but we leave this to future work.

Another concern is the small size of the testing sets used in the community. With test sets in the low hundreds of molecules, there is a limit on the power of conclusions we can draw about model performance. Even when training on several thousand molecules, we observe a worse performance when test set distributions are substantially different from the training set (Figure S3, Fold1 column TableS1). This indicates that our models are not learning generalizable molecular features to make their predictions. The large range of the possible values in AqSolDB (14 log units) makes it easier to achieve high correlation statistics, which is why we favored analysis of the RMSE of the given predictions to compare model performance.

SolTranNet had a worse performance on the SC2 data set than the best performing group (Table 2). However, it should be noted that the test sets for SC2 were not blind. Thus, the most relevant comparison to the reported metrics is the STN best model in Table 2, as it is selected with knowledge of performance on the test set. SolTranNet exhibited the largest amount of training variability on these test sets, which indicates that we could potentially increase the performance by optimizing the SolTranNet architecture for these test sets. Even the best reported models have an RMSE over half a log unit, which raises the question "What level of performance makes a model useful for drug discovery?"

.

We restructured the evaluation into a classification task to investigate this. We chose to analyze the sensitivity and false discovery rate, as we desire a model that correctly identifies soluble compounds (high sensitivity) and avoids falsely identifying insoluble compounds (low false discovery rate). SolTranNet achieves a comparable classification performance to the other methods submitted to SC2 (Figure 2).

We have shown that SolTranNet is capable of predicting aqueous solubility and outperforms linear ML approaches. During our hyperparameter optimization, we demonstrate that models with more parameters do not perform better in our scaffold-based CCV splits and exhibit a worse performance on a withheld test set (Tables S1 and S2). SolTranNet's smaller size, removal of the distance matrix, and implementation of multiprocessing makes it run 118 times faster than the original MAT implementation (Table 1). SolTranNet has a comparable performance to current ML models (Tables 2 and S5, Figure 2). We have deployed SolTranNet via PIP for easy integration into drug discovery pipelines, and its source code is available under an Apache open source license at https://github.com/gnina/SolTranNet.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jcim.1c00331.

Figure S1, two-stage hyperparameter sweep for SolTranNet; Figure S2, Effect of dynamic stopping on SolTranNet performance; Figure S3, distributions of various training set data; Figure S4, RMSE and R^2 performance for first 10 models trained during architecture sweep; Figure S5, RMSE of five different SolTranNet trained on scaffold split of AqSolDB; Figure S6, R^2 of five different SolTranNet trained on scaffold split of AqSolDB; Figure S7, plots of the Deployed SolTranNet model predicting test sets for each random ESOL split; Figure S8, plots of SolTranNet trained on random splits of ESOL predicting AqSol; Figure S9, training distributions of AqSol and ESOL; Table S1, SolTranNet model hyperparameter search; Table S2, SolTranNet hyperparameter search correlations; Table S3, featurization used to embed atoms in SolTranNet; Table S4, performance of five different seeds of SolTranNet; Table S5, performance of SolTranNet on other published data sets; and Table S6, SolTranNet comparison of performance utilizing ESOL or AqSol (PDF)

■ AUTHOR INFORMATION

Corresponding Author

Paul G. Francoeur − *Department of Computational and Systems Biology, University of Pittsburgh, Pittsburgh, Pennsylvania 15260, United States;* orcid.org/0000-0002-1440-567X; Email: paf46@pitt.edu

Author

David R. Koes − *Department of Computational and Systems Biology, University of Pittsburgh, Pittsburgh, Pennsylvania 15260, United States;* orcid.org/0000-0002-6892-6614

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jcim.1c00331

Funding

This work is supported by R01GM108340 from the National Institute of General Medical Sciences and a GPU donation from the NVIDIA corporation.

Notes

The authors declare no competing financial interest.

Data and Software: SolTranNet is open source and available under the Apache2.0 license. SolTranNet is available via PIP installation for PYTHON3, and the source code is available at https://github.com/gnina/SolTranNet. All data sets and code used for the analysis in this paper are available at https://github.com/francoep/SolTranNet_paper

■ ACKNOWLEDGMENTS

We thank Andrew McNutt and Jonathan King for their contributions during manuscript preparation.

REFERENCES

(1) Musha, D. Studies on Body Water in Man. *Tohoku J. Exp. Med.* **1956**, *63*, 309−317.

(2) Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug Delivery Rev.* **1997**, *23*, 3−25.

(3) Di, L.; Kerns, E. H. Biological assay challenges from compound solubility: strategies for bioassay optimization. *Drug Discovery Today* **2006**, *11*, 446−451.

(4) Ekins, S.; Rose, J. In silico ADME/Tox: the state of the art. *J. Mol. Graphics Modell.* **2002**, *20*, 305−309.

(5) Lyu, J.; Wang, S.; Balius, T. E.; Singh, I.; Levit, A.; Moroz, Y. S.; O'Meera, M. J.; Che, T.; Algaa, E.; Tolmachova, K.; Tolmachev, A. A.; Shoichet, B. K.; Roth, B. L.; Irwin, J. J. Ultra-large library docking for discovering new chemotypes. *Nature* **2019**, *566*, 224−229.

(6) Boothroyd, S.; Kerridge, A.; Broo, A.; Buttar, D.; Anwar, J. Solubility prediction from first principles: a density of states approach. *Phys. Chem. Chem. Phys.* **2018**, *20*, 20981−20987.

(7) Skyner, R. E.; McDonagh, J. L.; Groom, C. R.; van Mourik, T.; Mitchell, J. B. O. A review of methods for the calculation of solution free energies and the modelling of systems in solution. *Phys. Chem. Chem. Phys.* **2015**, *17*, 6174−6191.

(8) Jorgensen, W. L.; Duffy, E. M. Prediction of drug solubility from structure. *Adv. Drug Delivery Rev.* **2002**, *54*, 355−366.

(9) Llinas, A.; Oprisiu, I.; Avdeef, A. Findings of the Second Challenge to Predict Aqueous Solubility. *J. Chem. Inf. Model.* **2020**, *60*, 4791−4803.

(10) Boobier, S.; Osbourn, A.; Mitchell, J. B. O. Can human experts predict solubility better than computers? *J. Cheminf.* **2017**, *9*, 63.

(11) Lovric, M.; Pavlovic, K.; Zuvela, P.; Spataru, A.; Lucic, B.; Kern, R.; Wong, M. W. Machine Learning in Prediction of Intrinsic Aqueous Solubility of Drug-like Compounds: Generalization, Complexity or Predictive Ability? *J. Chemom.* **2021**, DOI: 10.1002/cem.3349.

(12) Cui, Q.; Lu, S.; Ni, B.; Zeng, X.; Tan, Y.; Chen, Y. D.; Zhao, H. Improved Prediction of Aqueous Solubility of Novel Compounds by Going Deeper With Deep Learning. *Front. Oncol.* **2020**, *10*, 121.

(13) Maziarka, L.; Danel, T.; Mucha, S.; Rataj, K.; Tabor, J.; Jastrzebski, S. Molecule Attention Transformer. 2020, *arXiv*. https://arxiv.org/abs/2002.08264 (accessed 2021-05-21).

(14) Ramsundar, B.; Eastman, P.; Walters, P.; Pande, V.; Leswing, K.; Wu, Z. *Deep Learning for the Life Sciences*; O'Reilly Media: 2019.

(15) Yang, K.; Swanson, K.; Jin, W.; Coley, C.; Eiden, P.; Gao, H.; Guzman-Perez, A.; Hopper, T.; Kelley, B.; Mathea, M.; Palmer, A.; Settels, V.; Jaakkola, T.; Jensen, K.; Barzilay, R. Analyzing Learned Molecular Representations for Property Prediction. *J. Chem. Inf. Model.* **2019**, *59*, 3370−3388. PMID: 31361484

(16) Sorkun, M. C.; Khetan, A. AqSolDB, a curated reference set of aqueous solubility and 2D descriptors for a diverse set of compounds. *Sci. Data* **2019**, *6*, 143.

(17) RDKit: Open-Source Cheminformatics. http://www.rdkit.org (accessed 2017-11-06).

(18) Harris, C. R.; Millman, K. J.; van der Walt, S. J.; Gommers, R.; Virtanen, P.; Courna- peau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N. J.; Kern, R.; Picus, M.; Hoyer, S.; van Kerkwijk, M. H.; Brett, M.; Haldane, A.; del R', J. F.; Wiebe, M.; Peterson, P.; G'erard-Marchant, P.; Sheppard, K.; Reddy, T.; Weckesser, W.; Abbasi, H.; Gohlke, C.; Oliphant, T. E. Array programming with NumPy. *Nature* **2020**, *585*, 357−362.

(19) Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library; *NeurIPS 2019*; Curran Associates, Inc.: 2019; pp 8024−8035.

(20) Francoeur, P. G.; Masuda, T.; Sunseri, J.; Jia, A.; Iovanisci, R. B.; Snyder, I.; Koes, D. R. Three-Dimensional Convolutional Neural Networks and a Cross-Docked Data Set for Structure-Based Drug Design. *J. Chem. Inf. Model.* **2020**, *60*, 4200−4215. PMID: 32865404

(21) Delaney, J. S. ESOL: Estimating Aqueous Solubility Directly from Molecular Structure. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1000−1005. PMID: 15154768

(22) Mobley, D. L.; Guthrie, J. P. FreeSolv: a database of experimental and calculated hydration free energies, with input files. *J. Comput.-Aided Mol. Des.* **2014**, *28*, 711−720.

(23) Biewald, L. *Experiment Tracking with Weights and Biases*. 2020. https://www.wandb.com/ (accessed 2021-05-21). Software available from wandb.com.

(24) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blon-del, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Courna-peau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825−2830.

https://doi.org/10.1021/acs.jcim.1c00331
J. Chem. Inf. Model. 2021, 61, 2530−2536