SMILES2vec: An Interpretable General-Purpose Deep Neural Network for Predicting Chemical Properties

Anonymous Authors

Abstract

Chemical databases store information in text representations, and the SMILES format is a universal standard used in many cheminformatics software. Encoded in each SMILES string is structural information that can be used to predict complex chemical properties. In this work, we develop SMILES2vec, a deep RNN that automatically learns features from SMILES strings to predict chemical properties, without the need for additional explicit chemical information, or the "grammar" of how SMILES encode structural data. Using Bayesian optimization methods to tune the network architecture, we show that an optimized SMILES2vec model can serve as a general-purpose neural network for learning a range of distinct chemical properties including toxicity, activity, solubility and solvation energy, while outperforming contemporary MLP neural networks that uses engineered features. Furthermore, we demonstrate proof-of-concept of interpretability by developing an explanation mask that localizes on the most important characters used in making a prediction. When tested on the solubility dataset, this localization identifies specific parts of a chemical that is consistent with established first-principles knowledge of solubility with an accuracy of 88%, demonstrating that neural networks can learn technically accurate chemical concepts. The fact that SMILES2vec validates established chemical facts, while providing state-of-the-art accuracy, makes it a potential tool for widespread adoption of interpretable deep learning by the chemistry community.

1. Introduction

1.1 Machine Learning with Engineered Features

Since the 1980s, various machine learning (ML) algorithms ranging from simple linear regression to support vector machines (SVM) have been used to predict the properties of chemicals (Chersakov, 2014).

With the use of traditional ML models, comes the prerequisite of having an appropriate set of features. Feature engineering in chemistry is a sophisticated science that stretches back to the late 1940s (Platt, 1947). Molecular descriptors, as they are termed by chemists, are basic computable properties or sophisticated descriptions of a chemical's structure, and these engineered features were developed based on first principles knowledge. To date, over 5000 molecular descriptors have been developed (Todeschini, 2008). In addition, molecular fingerprints have also been designed, which instead of computing a basic property, provides a description of a specific part of the chemical's structure (Rogers, 2010). Modern *in silico* modeling in chemistry is therefore predicated on correlating these engineered features with the activity or property of the chemical, which is formally known as the field of Quantitative Structure-Activity or Structure-Property Relationship (QSAR/QSPR) modeling (Chersakov, 2014).

1.2 Deep Learning with Engineered Features

In chemistry, deep learning (DL) models using either molecular descriptors or fingerprints have emerged in recent years. Notably, DL models have won recent competitions, such as the 2012 Merck Kaggle challenge, and the 2014 NIH Tox21 challenge.

In addition, several research groups have started developing DL-based models to predict other properties, ranging from biochemical (Dahl, 2014; Mayr, 2016; Ramsundar, 2015; Unterthiner, 2014; Wallach 2016) to chemical (Hughes, 2015; Lusci, 2013) to physical and electronic properties (Montavon, 2013; Schutt 2017; Smith, 2017). In general, these models either perform at parity or slightly outperform prior state-of-the-art models based on traditional ML algorithms (Gawehn, 2015; Goh, 2016).

1.3 Limitations of Engineered Features

Compared to the computer science literature, ML/DL research in chemistry is based almost exclusively on engineered features, which has both benefits and drawbacks. The benefit is that it utilizes existing knowledge, and this accelerates the learning process, in the sense that the ML/DL algorithm will not need to develop appropriate representations in order to construct a predictive model. At the same time, this is also a drawback, as using engineered features will limit the search space of potentially developable representations, and if the features are suboptimal, it could reduce the accuracy of the resulting models.

With the growth of chemical data (Goh, 2016), it may be desirable to fully leverage representation learning of deep neural networks, which will enable us to predict novel chemical properties for which little or no feature engineering research has been performed. In computer vision and natural language research, this is achieved by providing little or no transformation to the original data. For example, unaltered images are used as the input in various CNN models (He, 2015) and similarly unaltered text is used in LSTM-based models (Wu, 2016).

1.4 Related Work

In the chemistry context, the original data would be data that describes the structure and orientation of the chemical. In basic chemistry education, students are taught how to draw a 2D diagram of a chemical (i.e. an image), which serves as the primary medium of communication amongst chemist. Alternatively, the same structural information can be encoded as graphs. To date, there has been limited representation learning research in any of these data formats. Perhaps the most developed is the use of molecular graphs in deep neural networks (Duvenaud, 2015; Kearnes, 2016). Image data has also been recently reported to work effectively when using deep CNN models (Goh, 2017).

This structural information can also be encoded in a text format, such as SMILES (Weininger, 1988). In addition, the SMILES format is also the basis for inter-operability between various cheminformatics software. In terms of text representations, we acknowledge the existing work of others (Bjerrum, 2017).

Lastly, this raises an important question: *Is SMILES sufficient to capture the first order distinction between different chemical properties?* Assuming that the hypothesis is correct, *is it possible to validate it with established first principles knowledge on simple properties (e.g. solubility)?*

This work distinguishes itself from related work in the following manner, and the paper is as organized as follows: (i) we perform extensive experiments to determine the optimal neural network architecture for interpreting the SMILES “chemical language”, (ii) we developed an explanation mask to explain *why* the neural network makes a particular prediction using the solubility dataset as an example, (iii) we show how the same network architecture can be used to predict broad categories of chemical properties, and (iv) we demonstrate that the resulting SMILES-based models have better accuracy than contemporary multi-layer perceptron (MLP) models that uses engineered features.

2. Methodologies

Here, we document the methodologies used in the development of SMILES2vec. First, we provide a brief introduction to SMILES. Next, we detail the datasets used, data preparations and pre-processing steps. Then, we examine the details of optimizing the neural network using Bayesian methods, as well as the training protocol. Lastly, we document the steps used in constructing the explanation mask.

2.1 Introduction to SMILES

SMILES, a “chemical language” (Weininger, 1988) encodes structural information of a molecule into a compact text representation. There is a regular “grammar” to SMILES. For example, the alphabets denote atoms, and in some cases also what type of atoms. For example, *c* and *C* denote aromatic and aliphatic carbons respectively. Special characters like -, = denote type of bonds. Rings are denoted by encapsulating numbers, and side chains by round brackets. Thus, with sufficient training a chemist can read SMILES and infer the structure of the chemical. From this structural information, more complex properties can be predicted, as it has been done in QSAR/QSPR modeling.

Inspired by language translation RNN work (Wu, 2016), we do not encode information about the “grammar” of SMILES. Instead, we expect that the RNN should learn these patterns and if necessary use them to develop intermediate features that would be relevant for predicting a variety of chemical properties.

2.2 Datasets Used

Our work creates a RNN model for general chemical property prediction, and ideally it should work effectively for different types of properties. To make our results comparable with contemporary DL-based models reported in the literature, we used the Tox21, HIV, and FreeSolv dataset from the MoleculeNet benchmark (Wu, 2017). In addition, we also used the ESOL solubility dataset as a proof-of-concept for interpreting SMILES2vec.

Description of Dataset.

Dataset	Property	Task	Size
Tox21	Non-Physical (Toxicity)	Multi-task binary classification	8014
HIV	Non-Physical (Activity)	Single-task binary classification	41,193
FreeSolv	Physical (Solvation)	Single-task regression	643
ESOL	Physical (Solubility)	Single-task regression	1128

Table 1: Characteristics of the 4 datasets examined in this study.

This dataset as summarized in Table 1, comprises of a mix of large vs small datasets, physical vs non-physical chemical properties and regression vs classification problems.

Data Preparation.
The length of the SMILES string directly impacts the compute resources required to train RNN models. To maintain a balance between maximum compatibility with existing SMILES data, but also rapid training time, we surveyed the ChEMBL database, a collection of industrially-relevant chemicals that has over 1 million entries (Gaulton, 2011). Using this database as a proxy for scientifically-relevant chemicals, we calculated that setting a maximum length of 250 characters would encompass 99.9% of existing entries. Therefore, we excluded entries of more than 250 characters in the dataset.

Next, we created a dictionary that mapped the unique characters as one-hot encodings. Zero padding was applied to ensure that shorter strings had a uniform size of 250 characters. In addition, extra padding of 10 zeroes were added both to the left and right of the string. Apart from the above-mentioned steps, no additional data augmentation steps were performed.

Data Preprocessing.
We used a dataset preprocessing step similar to that reported in previous work (Wu, 2017; Goh, 2017). A separate test set was partitioned out to serve as a test for generalizability. For the Tox21 and HIV dataset, 1/6th was partitioned out to form the test set, and for the FreeSolv and ESOL dataset, 1/10th was used to create the test set. The remaining 5/6th or 9/10th of the dataset was then used in the random 5-fold cross validation approach for training.

Model performance and early stopping criterion was determined by validation loss. Lastly, we oversampled the minority class to mitigate class imbalance. This was achieved by computing the ratio of both classes, and appending additional data from the smaller class by that ratio.

2.3 Designing the Neural network

The neural network design was optimized using a Bayesian optimization method that adjusted the model's hyperparameters. A standardized training protocol was used in training, and the model was trained in a supervised manner. The validation metrics was used as feedback to the Bayesian optimizer for suggesting new network designs.

Bayesian Optimization of Neural Network Design.
We used a Bayesian optimizer, SigOpt (Dewancker, 2016) to optimize the hyperparameters related to the neural network topology. Each different set of network hyperparameters are defined as a separate "trial", and for each trial, we trained the model to completion and used the validation metric (AUC for classification tasks, RMSE for regression tasks) as input to perform Bayesian optimization. To prevent overfitting during this process, the splitting of the dataset between training and validation sets was governed by a random seed. However, a fixed test set was maintained throughout, which is not used in the Bayesian optimization process. By comparing the difference in validation and test set metrics, it would thus allow us to determine if the network design was being overfitted to the training/validation data.

No hyperparameters optimization was performed for the learning protocol. Instead, we used a standardized protocol utilizing adaptive solvers (see next section).

Lastly, it should be noted that only a subset of the dataset was used in the Bayesian optimization. Specifically, we used only a single task (nr-ahr toxicity) from the Tox21 dataset and the Freesolv dataset.

Supervised Training Protocol.
SMILES2vec was trained using a Tensorflow backend (Abadi, 2016) with GPU acceleration using NVIDIA CuDNN libraries.(Chetlur, 2014) The network was created and executed using the Keras 2.0 functional API interface (Chollet, 2015). We use the RMSprop algorithm (Hinton, 2012) to train for 250 epochs using the standard settings recommended (learning rate = 10^{-3}, $\rho = 0.9$, $\varepsilon = 10^{-8}$). We used a batch size of 32, and also included an early stopping protocol to reduce overfitting. This was done by monitoring the loss of the validation set, and if there was no improvement in the validation loss after 25 epochs, the last best model was saved as the final model.

For classification tasks, we used the binary cross-entropy loss function for training. The performance metric reported in our work for the model performance is area under the ROC-curve (AUC). For regression tasks, we used the mean average error as the loss function for training. The performance metric reported in is RMSE.

Unless specified otherwise, the reported results in our work denote the mean value of the performance metric, obtained from the 5 runs in the 5-fold cross validation.

3. Experiments

In this section, we first conduct several Bayesian optimization experiments to tune the l network architecture and hyperparameters of SMILES2vec. Then, we conduct experiments to develop an explanation mask for improving interpretability, and demonstrate proof-of-concept using the ESOL solubility dataset. Lastly, we demonstrate generalizability of SMILES2vec by evaluating its performance on other datasets.

3.1. SMILES2vec Neural Network Design

RNNs, particularly those based on LSTMs (Hochreiter, 1997) or GRUs (Cho, 2014) are effective neural network designs for learning from text data. Its effectiveness has been demonstrated in examples like the Google Neural Translation Machine that uses an architecture of 8+8 layers of residual LSTM units (Wu, 2016). Most of other reported application of RNNs in natural language research are similarly used to model sequence-to-sequence predictions, and often fewer (i.e. 2 to 4) layers have been found to be sufficiently accurate for their tasks.

Our work is different because we are modeling sequence-to-vector predictions, where the sequence is a SMILES string (i.e. text), and the vector is a measured chemical property. To the best of our knowledge, there has been limited related work in the chemistry domain (Bjerrum, 2017) and existing model designs from natural language research (Wu, 2016) may not be transferable to a chemical language. Therefore, a substantial component of our work is in the design of the RNN architecture.

Architectural Class Exploration.
We first explore the RNN model's architecture class, which primarily includes high-level design choices, such as the type of units used, type of layers, arrangement of layers, etc.

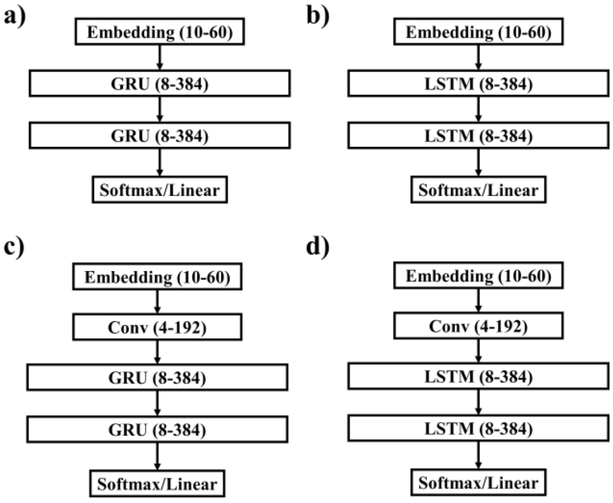

Figure 1: Illustration of the 4 architectural classes investigated, (a) GRU, (b) LSTM, (c) CNN-GRU and (d) CNN-LSTM. Number of units explored is indicated in parenthesis.

LSTMs and GRUs are the two major RNN units used in the literature, and form the basis of two architectural classes. The template design for each class starts with an embedding layer that feeds into a 2-layer bidirectional GRU or 2-layer bidirectional LSTM as illustrated in Figure 1. In addition, we explored the utility of adding a 1D convolutional layer between the embedding and GRU/LSTM layers. This design forms the template of the other two architectural classes investigated.

A separate Bayesian optimization was used to tune the hyperparameters of each architectural class. Specifically, we varied the size of the embedding from 10 to 60 in intervals of 10. The number of units in the GRU/LSTM layers ranged from 8 to 384 in intervals of 8, and the number of units in the convolutional layer ranged from 4 to 192 in intervals of 4. For the convolutional layer, a size of 3 and a stride of 1 was used, which is based on the design principles from modern convolutional neural network (Szegedy, 2015). No additional optimization was performed on the size or stride of the convolutional layers. In addition, no specific shape of the network topology was enforced.

Bayesian Optimization of Hyperparameters.
In order for Bayesian methods to be effective, a sufficient number of trials for different neural network design has to be performed. In practice, it has been recommended that a minimum of 10N trials be performed, where N is the number of tunable hyperparameters. In our work, we performed 60 trials for each of the 4 architectural class. In addition, we manually seeded 6 initial designs for each class. Specifically, we used initial designs that had an embedding size of 40, a convolution layer with 16 filters, and both LSTM/RNN layer with [8, 16, 32, 64, 128, 256] units.

In addition, because we are developing a general-purpose neural network design that can be used for a broad range of property prediction, it would not be feasible to include all training data to optimize the network design within the limits of available computing resources. Therefore, a subset of the datasets were used in the Bayesian optimization (see section 2.3 for details), and separate optimizations were performed for the Tox21 classification and the FreeSolv regression task.

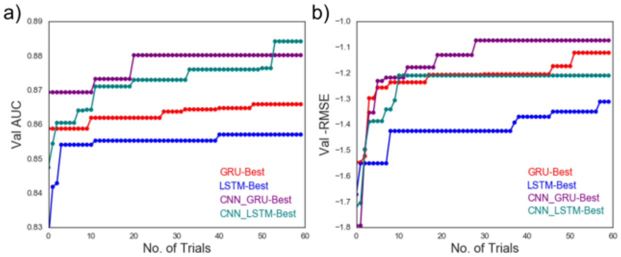

Figure 2: Results of Bayesian optimization of the hyperparameters of the 4 architectural classes for the (a) Tox21 classification and (b) Freesolv regression tasks.

The results of the Bayesian optimization across all 4 classes and 2 tasks are as indicated in Figure 2. For Tox21 classification, we observed that an additional convolutional layer between the embedding and RNN/LSTM layers improved model performance relative to their counterparts, and the best performing model was the CNN-LSTM class, with CNN-GRU trailing slightly behind. For FreeSolv regression, we observed that GRU-based networks outperform LSTM-based networks. Taking into considerations for generalization to other type of chemical properties, we selected the CNN-GRU architectural class for the remainder of this paper. Then, we selected the best network design of this class (see supplementary data for details), which is summarized in Table 2.

em_size	#conv	#rnn1	#rnn2
50	192	224	384

Table 2: Best CNN-GRU network design for the final SMILES2vec model.

Lastly, because the Bayesian algorithm uses the validation metric as a means to tune the network's hyperparameters, there is a possibility that as one progresses with the optimization, there may be overfitting towards the validation set. To determine the extent of overfitting, we examined the correlation between the validation metrics (whose validation set data would be changing during the Bayesian optimization) and the test metrics (whose test set data is fixed, and was never used in the Bayesian optimization). As illustrated in Figure 3, the correlation between validation and test metrics is 0.54 for the Tox21 dataset and 0.78 for the FreeSolv dataset. The lower correlation of the Tox21 dataset relative to the FreeSolv dataset can be explained, as the AUC performance metric on which the optimization was performed, is not the same as the cross-entropy loss function used for training.

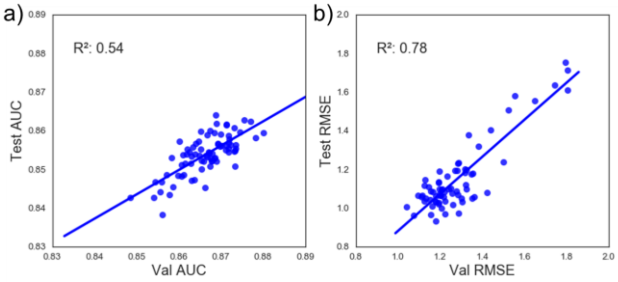

Figure 3: Correlation plot between validation and test metrics for all trials enumerated for the (a) Tox21 and (b) Freesolv datasets.

3.2. SMILES2vec Interpretation

To gain a better insight into the SMILES2vec model, we developed a method to gain some level of interpretability. Here, our objective is to identify the part(s) of the SMILES string that is responsible for the neural network's decision.

Methods for making "black box" models exist (Ribeiro, 2016), but most of these methods tend to require explicit combinatorial analysis. Our approach based on the work of [redacted], provides insight into how the neural network analyzes the data, without combinatorically probing the input. This is achieved by training an explanation mask, whereby a separate explanation network learns to mask input data to produce near identical output from the original unmasked input.

Training the Explanation Mask.

We train a neural-network generated mask to identify the important characters of the input. The procedure is as follows: First, we use the final SMILES2vec model as the base network. Next, we construct another neural network to produce a mask over the input data, such that the output of the base neural network remains the same. We freeze the base neural network, and train the explanation mask end-to-end, as shown in Figure 4.

With the weights of the base network frozen, the mask being learned will be specific to the SMILES2vec model. To avoid the mask being trivial, we added two forms of regularization, a small L2 regularization, and also penalized the mask for having high entropy. The overall loss function for a single element of each mini batch is as follows:

$$Loss_i = ||f(SMILES_i, \theta) - Sol(SMILE_i)||_2 \\ +1e^{-6}||\, MASK_i||_2 + 0.05\, H(MASK_i)$$

where $f(SMILES_i,\ \theta)$ is the base neural network applied to the ith SMILES, $Sol(SMILE_i)$ is the solubility, H is the entropy over the mask (with weights normalized), and $MASK_i$ is the value of the calculated explanation mask at each entry in the input SMILES string.

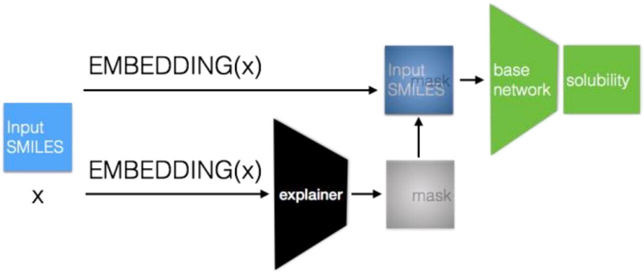

Figure 4: Structure for training the explanation network. The SMILES input is passed through the embedding layer, then into the explainer. This produces a mask which is placed over the original embedding and sent through the pre-trained base model.

The explanation network used to create the mask was a 20 layer residual network with SELU (Klambauer, 2017) activations. The padding was fixed such that the length of the input remained the same at each layer. The input to the network is the embedding of the SMILES string. The last layer is a 1D convolution of length 1, followed by a batch normalization then by a softplus activation. We observed that the final batch normalization layer to be very important for final trainability. We trained with Adam (Kingma, 2014) until convergence. We started the learning rate at 10^{-2} and divided by 10 as the training error plateaued, ultimately training down to 10^{-6}.

Interpreting Chemical Solubility.

We demonstrate proof-of-concept for an interpretable SMILES2vec network using the ESOL solubility dataset (Wu, 2017). Chemical solubility as it is a well-studied and simple property where there is established first principles knowledge. Briefly, parts of a chemical (i.e. functional groups) can usually be classified as either hydrophilic or hydrophobic. Hydrophilic groups like alcohols, amines and carboxyl form strong interactions with water and increase the overall solubility of a compound, and they typically contain non-carbon atoms like nitrogen and oxygen. The reverse is true for hydrophobic groups, which tend to make chemicals more insoluble, and they typically are carbon-based chains/rings and halogens.

We first trained the SMILES2vec base model, which attained a validation RMSE of 0.63. Solubility values are reported on the $\log_{10}$ scale, with less soluble compounds having more negative number, and more soluble compounds having a less negative number. The mask outputs a normalized attention value that denotes the importance of a particular character in the network's decision. For each SMILES string, we identified the top-3 characters (see Table 3 for examples). Then, we separated the dataset into soluble (> -1.0) and insoluble (< -5.0) compounds. Using established knowledge of chemical solubility to establish the ground truth, we expect that soluble compounds should have higher attention on the atoms O, N, and insoluble compounds to have higher attention on atoms C, F, Cl, Br, I. With this ground truth labeling in expected atoms, we computed the top-3 accuracy of SMILES2vec interpretability, which is 88%.

SMILES	Solubility	Top-3 Chars
c1ccncc1	1.18	c,c,n
O=Cc1ccco1	-0.87	C,c,o
Clc1ccc(cc1Cl)c2ccccc2	-5.93	c,c,c
Cc1c2ccccc2c(C)c3ccccc13	-6.91	c,c,c

Table 3: Sample SMILES entries, their predicted solubility value, and the top-3 most important characters colored in blue.

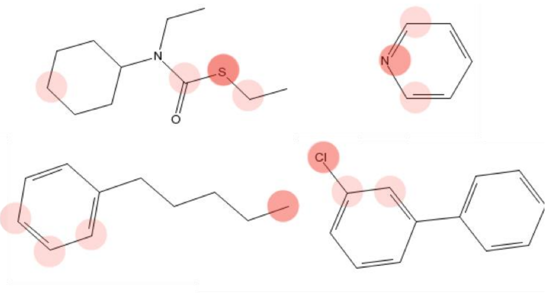

Figure 5: Colored circles of increasing darkness indicate the locations of increasing attention on the molecule. The explanation mask validates established knowledge by focusing on atoms of known hydrophobic and hydrophilic functional groups.

In addition, we also qualitatively examined the outputs of the masks by mapping the SMILES character to the corresponding atom(s) in the molecular structure, and examples are shown in Figure 5. For molecules with low solubility, the characters *c*, *C*, and *Cl* tend to receive more attention than others, which correspond to hydrophobic groups that typically have carbon and chlorine atoms. In contrast, molecules with low solubility have attention focused on the characters *O* and *N*, which correspond to oxygen and nitrogen atoms that are common in hydrophilic groups.

The localization of appropriate atoms on each functional group type depending on the chemical's predicted solubility value, thus indicate that SMILES2vec has learned representations that correspond to known chemistry concepts. Lastly, we emphasize that SMILES2vec has developed these representations *without* being provided any explicit chemical information. While chemical information is implicitly encoded in the SMILES string, no "decoding solution" was provided to the network, which demonstrates the effectiveness of representation learning from raw data in the chemical sciences.

Generalization of SMILES2vec Models

Thus far, we have only evaluated the performance of SMILES2vec on the ESOL solubility dataset. We also note that the architectural optimization of SMILES2vec only included a small fraction of the 4 datasets identified for this work; the HIV dataset was not included, and 11 out of 12 toxicity tasks were not included. Hence, we need to determine how generalizable the Bayesian optimized network design will be to other chemical tasks.

Unlike the solubility example, the other 3 datasets (toxicity, activity, solvation energy) are more complex properties for which no simple rule-based methods exist in the chemistry literature. As such, quantifying the accuracy of the SMILES2vec interpretation is non-trivial, and is beyond the scope of this work. Nevertheless, the accuracy of the model's predictions can still be evaluated.

First, we determined the effectiveness of generalizing SMILES2vec to the 3 remaining datasets. The following validation performance metrics were obtained: AUC of 0.799 for the full Tox21 dataset, AUC of 0.781 for the HIV dataset, and RMSE 1.14 kcal/mol for the FreeSolv dataset. Furthermore, we note that in all models, the difference between the validation and test metrics is small, further confirming the generalization of the model to compounds it has not seen either during the model training, or in the Bayesian hyperparameter optimization.

Based on these results, we conclude that the Bayesian optimization of the network architectural design was effective in developing a general-purpose SMILES2vec network design for other chemical properties. We also note in recent literature there has been a trend towards using other "black box" approaches as a solution for network architecture design, for example using RNNs and reinforcement learning to optimize the design of a target neural network (Zoph, 2017). However, such methods typically require on the order of ~10,000 trials, which is much more than the ~500 trials used in our work. In addition, given that the template of each architectural class was fixed, adaptive methods that grow or shrink the neural network are also viable alternatives (Siegel, 2016).

Next, we compare the performance of SMILES2vec against contemporary deep neural networks that have reported results on the same datasets (Tox21, HIV, FreeSolv) that we have evaluated our model on. We compare against a typical MLP network that uses engineered features (Wu, 2017), a chemistry-specific molecular graph convolutional neural network (Wu, 2017), and Chemception, a deep CNN that uses images (Goh, 2017).

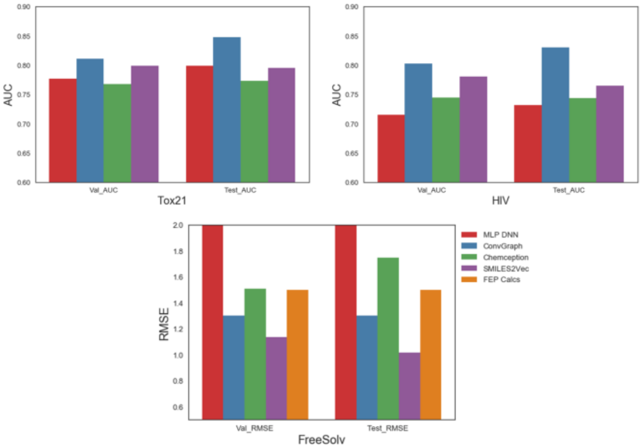

Figure 6: Performance of SMILES2vec against contemporary deep neural networks trained on engineered features, image and graph data. For Tox21 and HIV, higher AUC is better. For FreeSolv, lower RMSE is better.

The results are presented in Figure 6, and we use validation metrics to evaluate the quality of the model. In comparing the 4 methods, we observed tha the standard MLP DL models that uses engineered features (fingerprints) performed the worst. SMILES2vec outperformed image CNN models such as Chemception. In addition, it also outperformed first-principles models for computing solvation free energy (note: there are no first-principles models for computing toxicity/activity). Against convolutional graphs, which is the current state-of-the-art, SMILES2vec has mixed performance, it underperform slightly for classification tasks (Tox21: 0.799 vs 0.811, HIV: 0.781 vs 0.803) and outperform slightly for regression task (FreeSolv: 1.14 vs 1.30). While it cannot be concluded that SMILES2vec can consistently outperform the convolutional graph algorithm, any apparent loss in performance is arguably an acceptable trade off in lieu of a more interpretable model.

4. Conclusion

In this paper, we develop SMILES2vec, the first general-purpose deep neural network that uses chemical text data (SMILES) for predicting chemical property, with an explanation mask that improves interpretability. By performing extensive Bayesian optimization experiments, we identified a specific CNN-GRU neural network architecture that we showed to be effective in predicting a wide range of properties. SMILES2vec achieved a validation AUC of 0.799 and 0.781 for Tox21 toxicity and HIV activity prediction respectively, and a validation RMSE of 1.14 kcal/mol and 0.63 for solvation energy and solubility. Using a solubility dataset as an illustration of SMILES2vec interpretability, we construct explanation masks that indicate that SMILES2vec localizes on specific hydrophilic or hydrophobic functional groups when predicting chemical solubility, with a top-3 accuracy of 88%. Identification of such functional groups and their relationship to chemical solubility is a key first-principles concept in chemistry, which SMILES2vec was able to discover on its own. Compared to other deep learning models, SMILES2vec outperforms the typical MLP DL models that uses engineered features as input and CNN-based models that uses simple 2D images as input. Against the current state-of-the-art (convolutional graph networks), SMILES2vec outperforms on regression tasks and underperforms on classification tasks. These results indicate that SMILES2vec can accurately predict a broad range of properties and learn technically accurate chemical concepts, which suggest that it can be used as an interpretable tool for the future of deep learning driven chemical design.

References

Abadi, M., *et, al.* 2016. TensorFlow: A System for Large-Scale Machine Learning. In *OSDI,* 265-283.

Bjerrum, E.J. 2017. SMILES Enumeration as Data Augmentation for Neural Network Modeling of Molecules. *arXiv preprint arXiv:1703.07076*.

Cherkasov, A., *et, al.* 2014. QSAR modeling: where have you been? Where are you going to?. *J. Med. Chem. 57*(12): 4977-5010.

Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B. and Shelhamer, E. 2014. cudnn: Efficient primitives for deep learning. *arXiv preprint arXiv:1410.0759*.

Cho, K., Van Merriënboer, B., Bahdanau, D. and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

Chollet, F. 2015. Keras, https://github.com/fchollet/keras

Dahl, G.E., Jaitly, N. and Salakhutdinov, R. 2014. Multi-task neural networks for QSAR predictions. *arXiv preprint arXiv:1406.1231*.

Dewancker, I., McCourt, M., Clark, S., Hayes, P., Johnson, A. and Ke, G. 2016. Evaluation System for a Bayesian Optimization Service. *arXiv preprint arXiv:1605.06170*.

Duvenaud, D.K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A. and Adams, R.P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, 2224-2232.

Gaulton, A., *et, al.* 2011. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Res.* 40(D1): D1100-D1107.

Gawehn, E., Hiss, J.A. and Schneider, G. 2016. Deep learning in drug discovery. *Mol. Inf.* 35(1): 3-14.

Goh, G.B., Hodas, N.O. and Vishnu, A. 2017. Deep learning for computational chemistry. *J. Comput. Chem.* 38(16): 1291–1307.

Goh, G.B., Siegel, C., Vishnu, A., Hodas, N.O. and Baker, N.A., 2017. Chemception: A Deep Neural Network with Minimal Chemistry Knowledge Matches the Performance of Expert-developed QSAR/QSPR Models. *arXiv preprint arXiv:1706.06689*.

He, K., Zhang, X., Ren, S. and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision,* 1026-1034.

Hinton, G., Srivastava, N. and Swersky, K. 2012. RMSProp: Divide the gradient by a running average of its recent magnitude. *Neural networks for machine learning, Coursera lecture 6e*.

Hochreiter, S. and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735-1780.

Hughes, T.B., Dang, N.L., Miller, G.P. and Swamidass, S.J. 2016. Modeling Reactivity to Biological Macromolecules with a Deep Multitask Network. *ACS Cent. Sci.* 2(8): 529-537.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V. and Riley, P. 2016. Molecular graph convolutions: moving beyond fingerprints. *J. Comput. Aided Mol. Des.* 30(8): 595-608.

Kingma, D. and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Klambauer, G., Unterthiner, T., Mayr, A. and Hochreiter, S. 2017. Self-Normalizing Neural Networks. *arXiv preprint arXiv:1706.02515*.

Lusci, A., Pollastri, G. and Baldi, P. 2013. Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules. *J. Chem. Inf. Model.* 53(7): 1563-1575.

Mayr, A., Klambauer, G., Unterthiner, T. and Hochreiter, S. 2016. DeepTox: Toxicity Prediction using Deep Learning. *Front. Env. Sci.* 3(80): 1-15.

Montavon, G., Rupp, M., Gobre, V., Vazquez-Mayagoitia, A., Hansen, K., Tkatchenko, A., Müller, K.R. and Von Lilienfeld, O.A. 2013. Machine learning of molecular electronic properties in chemical compound space. *New J. Phys.* 15(9): 095003.

Platt, J.R. 1947. Influence of neighbor bonds on additive bond properties in paraffins. *J. Chem. Phys.* 15(6): 419-420.

Ramsundar, B., Kearnes, S., Riley, P., Webster, D., Konerding, D. and Pande, V. 2015. Massively multitask networks for drug discovery. *arXiv preprint arXiv:1502.02072*.

Ribeiro, M.T., Singh, S. and Guestrin, C. 2016. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135-1144.

Rogers, D. and Hahn, M. 2010. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* 50(5): 742-754.

Schütt, K.T., Arbabzadah, F., Chmiela, S., Müller, K.R. and Tkatchenko, A. 2017. Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* 8(13890): 1-8.

Siegel, C., Daily, J. and Vishnu, A. 2016. Adaptive neuron apoptosis for accelerating deep learning on large scale systems. In *IEEE International Conference on Big Data*, 753-762.

Smith, J.S., Isayev, O. and Roitberg, A.E. 2017. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* 8(4): 3192-3203.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1-9.

Todeschini, R. and Consonni, V. 2008. *Handbook of molecular descriptors* (Vol. 11). John Wiley & Sons.

Unterthiner, T., Mayr, A., Klambauer, G., Steijaert, M., Wegner, J.K., Ceulemans, H. and Hochreiter, S. 2014. Multi-Task Deep Networks for Drug Target Prediction. In *Advances in Neural Information Processing Systems*, 1-4.

Wallach, I., Dzamba, M. and Heifets, A. 2015. AtomNet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery. *arXiv preprint arXiv:1510.02855*.

Weininger, D. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comp. Sci.* 28(1): 31-36.

Wu, Y., *et, al.* 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

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

Zoph, B. and Le, Q.V. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.