Deep Learning Methods to Help Predict Properties of Molecules from SMILES

Gretchen Bonilla-Caraballo[1] and Manuel Rodriguez-Martinez[2]

[1] University of Puerto Rico, Mayagüez PR, USA
gretchen.bonilla@upr.edu,
[2] University of Puerto Rico, Mayagüez PR, USA

Abstract. Machine learning methods have been proposed in lieu of simulations to predict chemical properties of molecules. The trade-off here is paying for the training time once, in exchange for instant predictions on the input data. However, many of these methods rely heavily on feature engineering to prepare the data for these models. Moreover, the use of molecular structural information has been limited, despite having such information encoded in the Simplified Molecular Input Line Entry System (SMILES) format. In this paper we present a framework that relies on SMILES data to predict molecular properties. Our methods are based on 1-D Convolutional Networks and do not require complex feature engineering. Our methods can be applied to learn molecular properties from base data, thus making them accessible to a wider audience. Our experiments show that this method can predict the molecular weight and XLogP properties without any encoding of complex chemical rules.

Keywords: Deep Learning, SMILES, PubChem

1 Introduction

Understanding and predicting physical and chemical properties of molecules are important tasks for novel drug discovery, materials and battery design, particularly in the context of the development of medical and wearable devices. This process can be tedious as it often entails time-consuming experiments to characterize the molecules in the compounds. Simulation methods are often used to reduce the search space of options and experiments focus on the most promising molecules and target properties. But, these simulation methods often require large computational capabilities (e.g., clusters), parallel processing, and might take hours or even days to complete.

Machine learning (ML) methods have been proposed in lieu of simulations to predict chemical properties (Chakravarti, 2018), (Montavon et al, 2012), (Bepler and Berger, 2019), (Sendek et al, 2017), (Xie and Grossman, 2018), (Tshitoyan et al, 2019), (Schmidt et al, 2019). These methods include regression, support vector machines (SVM), decision trees, and neural networks. The goal in these works is to train a model that can predict the target chemical properties in a

very short time. The trade-off here is to pay for the training time once, and then reap the benefits of making instant predictions on the input data.

However, many of these methods rely heavily on *feature engineering* to prepare the data for these models. This limits their impact since targeted expertise in chemistry or related fields is needed to prepare the data. Moreover, the use of molecular structural information have been somewhat limited, despite having such information encoded in the Simplified Molecular Input Line Entry System (SMILES) format.

In this paper we present a framework that relies on SMILES data to help predict molecular properties. Our methods are based on 1-D Convolutional Networks (1-D Convs) and do not require complex feature engineering. Our methods can be applied to learn molecular properties from base data (e.g., first principle data), thus making them accessible to a wider audience. Our experiments show that this method can predict molecular weight and XLogP properties without any encoding of complex chemical rules.

1.1 Contributions

This paper provides the following original contributions:

- Describe deep learning (DL) architectures based on character embedding and convolutional neural networks (CNN) that can be used to predict chemical properties of compounds from their SMILES representation.
- Implement DL models that predict the molecular weight of chemical compounds based exclusively on the compound's SMILES.
- Implement DL models that predict the XLogP of chemical compounds using SMILES and in some instances using the compound's fragment representation as well.
- Describe an evaluation of these models on a data set obtained from the PubChem database (Kim et al, 2021) and identify the models with the highest precision per chemical property.

1.2 Paper Organization

The rest of this paper is organized as follows. In section 2 we provide background and motivation material to set the context for the remainder of the paper. In section 3 we present our deep learning models that can predict properties of molecules, using SMILES representations. Experimental setup and results are presented in section 4. Related works are presented and discussed in section 5. Finally, section 6 contains our summary of conclusions, and future works.

2 Motivation and Background

2.1 Motivation

As we mentioned in the section 1, a key goal of our work is to develop DL methods for predicting properties of molecules from first principles. That is,

rather than relying on complex and often *intuition-based* (Rajan, 2005) feature engineering efforts that require experienced domain scientists, we aim to utilize readily available chemical and physical data as input to the models. This approach leverages the capacity of deep neural networks to *generate features* in their early layers, which are then used by mid and late layers in the network. Properly trained and tuned deep models have been shown to rival or even surpass human ability in domains such as language translation and computer vision (Vaswani et al, 2017), (Devlin et al, 2019), (Long et al, 2015), (Bahdanau et al, 2014), (Krizhevsky et al, 2012), (Szegedy et al, 2014), (Conneau et al, 2018). Thus, a successfully trained model based on first principles data could help a wider range of scientists and practitioners study and predict properties of novel molecules.

Publicly available databases, such as PubChem (Kim et al, 2021) and OQMD (Saal et al, 2013), (Kirklin et al, 2015) provide the data from which to draw the set of basic features to serve as input to the models. These features are: a) chemical properties, b) physical properties, c) 2D and 3D imaging on spatial structures of molecules, and d) 2D imaging on molecular spectroscopy, to name a few.

2.2 Problem Formulation

We can state our problem as follows. Suppose we have a data set $\mathcal{S}$, consisting of a collection of records $s^{(0)}, s^{(1)}, ..., s^{(n)}$ containing information about some group of molecules. Each record $s^{(i)}$ contains information about a specific molecule in the data set. Let $\mathcal{P} = \{p_0, p_1, ..., p_r\}$ be a collection of molecular properties that we want to predict or estimate from $\mathcal{S}$. Our problem is then to:

1. *Transform* $\mathcal{S}$ into an input data set X, where

   $$X = \{x^{(0)}, x^{(1)}, ..., x^{(m)}\}.$$

   Here each element $x^{(i)}$ is an example that represents some set of base properties extracted from an item $s^{(j)} \in \mathcal{S}$. In this case, each example $x^{(i)}$ is a multi-dimensional vector with d dimensions

   $$x^{(i)} = (x_0^{(i)}, x_1^{(i)}, ..., x_d^{(i)})$$

   and X can be modeled as an $m \times d$ matrix containing these examples.
2. *Define* one or more DL models $M_0, M_1, ...M_p$ that predict the properties in $\mathcal{P}$. Thus, the prediction $y^{(i)}$ for a given model M_k on input $x^{(i)}$ might contain one or more of the properties in $\mathcal{P}$. In other words, in theory, a prediction y can be a multi-dimensional vector $y = (y_0, y_1, ..., y_l)$, with $l \leq r$, where each component represents some property in $\mathcal{P}$. For simplicity, in this paper we only consider models that predict one property at time from the input data X. Hence the model's output is one-dimensional (e.g., an integer or real label).

3. *Train* one or more DL models $M_0, M_1, ...M_p$ with the data in X. For each $x^{(i)}$ in X, a given model M_k will output a prediction $y^{(i)}$. The collection of predictions $y^{(0)}, y^{(1)}, ..., y^{(m)}$ from each of the m examples in X forms a column vector Y, where

$$Y = (y^{(0)}, y^{(1)}, ..., y^{(m)}).$$

4. *Use* the trained models $M_0, M_1, ...M_p$ to predict properties in $\mathcal{P}$ for previously unseen molecules.

2.3 PubChem Database

The PubChem[3] Database is one of the largest databases of chemical molecules and their properties. It is freely available and is maintained by the National Library of Medicine of the U.S. National Institutes of the Health (NIH). The database contains hundreds of millions of entries and provides search capabilities based on the name of the chemical, its molecular formula, or its compound identifier. Programmatic access is available through PUG-REST (Kim et al, 2018), a Python-based REST API.

For the purpose of this paper, we shall work with a sub-set of PubChem that contains the at least the following basic molecular properties:

1. *Canonical SMILES* - a string that specifies the chemical structures and bonds in a molecule. For example, the Canonical SMILES string for acetaminophen is CC(=O)NC1=CC=C(C=C1)O. We shall discuss more about the SMILES notation is section 2.4.
2. *Molecular Weight* - this is real number that represents the mass of a given molecule and its measured in *daltons*, or in molar mass (g/mol). The molecular weight for acetaminophen is 155.19 g/mol.
3. *XLogP* - this is real number representing the logarithm of the partition coefficient $\boldsymbol{P}$ of the molecule with respect to 1-octanol and water, and computed with the XLogP3 method (Cheng et al, 2007). The XlogP3 for acetaminophen is 0.5.

2.4 SMILES Representations

The Simplified Molecular Input Line Entry System (SMILES) is a character-based notation for describing chemical structures in a way that is easier for computers to interpret. SMILES represent a molecule's spatial organization model using a series characters without white spaces, employing a straightforward vocabulary and minimal grammar rules. In this notation, atoms are depict with their atomic symbols while bonds and branches are denoted by special characters. Figure 1, obtained from PubChem, presents the 2-D chemical structure of Cyclamic acid with a molecular formula of C6H13NO3S, and a canonical SMILES of C1CCC(CC1)NS(=O)(=O)O.

[3]PubChem URL: https://pubchem.ncbi.nlm.nih.gov

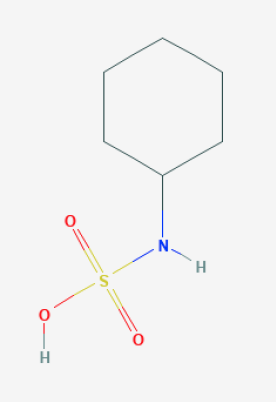

Fig. 1: 2-D chemical structure of Cyclamic acid.

One complication we found is that straightforward application the rules for generating a SMILES string can yield more than one string representation per molecule. To ensure consistency, chemical databases use a *canonicalization* algorithm so that each compound has a unique representation. In PubChem, this representation is called *Canonical SMILES*. Similarly, *Isomeric SMILES* are a variation of SMILES in which information on isotopes and stereochemistry (i.e., 3-D structures) is included within the string. A detailed description of the specifics of these canonicalization algorithms, isomeric variations, and stereochemistry is beyond the scope of this paper. The original SMILES framework was a proprietary framework, but since then a new open-standard called OpenSMILES has been developed by the chemistry research community. The interested reader can find more details in the OpenSMILES specification web site.

2.5 Applying NLP Ideas

A SMILES string can be seen as a "sentence" that describes the structural organization of a molecule in 2-D. The composition as well as the spatial structure of molecules are key determinants to the function of a molecule. Hence, our **key idea** is to apply *Natural Language Processing* (NLP) techniques to process this sentence and find patterns that help predict molecular properties. Word embedding (Mikolov et al, 2013), (Pennington et al, 2014), character-level embedding (Ling et al, 2015), (Zhang and LeCun, 2015), 1-D Convolutional Neural Networks 1-D (1-D Convs) (Zhang and LeCun, 2015), (Kim et al, 2016), (Jozefowicz et al, 2016), Long short-term memory networks (Conneau et al, 2018), (Ling et al, 2015), (Kim et al, 2016), Attention (Vaswani et al, 2017), Transformers (Devlin et al, 2019), and many other NLP techniques become key tools that can be applied to this domain.

3 Deep Learning Architectures

We now turn our attention to the DL models that we have developed to help predict molecular properties.

3.1 Overview

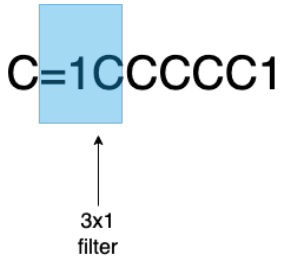

Fig. 2: A 3x1 filter.

Our main objective is to apply DL methods to SMILES strings to learn meaningful patterns from the atoms and bonds. The models created should predict molecular properties from first principles and without explicitly encoding rules from chemistry. Figure 2 shows this idea in the context of a 1-D Convs . In this figure, a 3×1 filter is used to scan the string and find patterns in the sequence of atoms and bonds. This filter moves over the string looking at groups of three characters at a time. It shifts one character to the right to look at the next group. The figure depicts the placement of the filter after the first movement, which covered 'C='. In practice, this filter does not run directly over the characters of the string but instead over the *character-level embedding* of the SMILES string. This approach will be explained in section 3.2

In this paper, we shall focus on the use of 1-D Convs as the core methods used to build our models. Our work is still early, and we have also experimented with LSTM networks and Transformers. However, 1-D Convs have so far provided us with the best trade off between training speed and prediction error. Comparing all these various methods in detail is part of our future work and is therefore outside the scope of this paper.

Figure 3 shows the end-to-end ML framework that we propose. We start out with a sample of the data from PubChem that contains 1,000,000 examples. We follow the standard practice of splitting this data set into training, validation and test sets. During the training phase, the training data is fed as tensor input to the model. Next, a character-level embedding layer takes care of mapping the characters in the SMILES string into an n-dimensional vector space. These transformed data are then passed through several 1-D Conv layers to build and detect features in the SMILES strings. Later on, the tensors generated in these

layers are "flattened" from their multi-dimensional shapes and fed into fully-connected layers, with the last layer producing the predictions of the model.

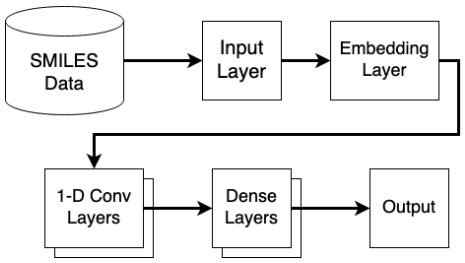

Fig. 3: Proposed end-to-end ML framework.

3.2 Character-level Embeddings

DL models expect their input as tensors, which the SMILES strings are not. Hence, we need a method to represent SMILES in tensor form. In the context of NLP, word embedding and character-level embedding have become the preferred way to accomplish this, replacing one-hot encoding and bag-of-words models. The reason being that embeddings can retain the information about relationships, context, and meaning that exists between adjacent words or characters. Thus, embeddings provide a dense representation of words, characters, and their meaning within a sentence.

In our work we will focus on character-level embeddings (Ling et al, 2015), (Zhang and LeCun, 2015). This is mainly because the SMILES' structure does not align with the typical sentence structure, where words are separated by spaces. Also, character level embeddings solve the *out-of-vocabulary (OOV) words* issue that often plagues word embedding methods.

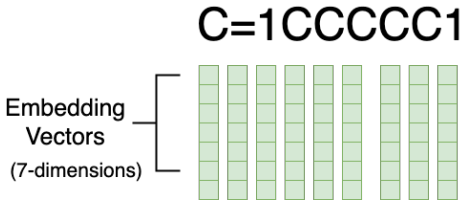

Fig. 4: SMILES Embedding.

Figure 4 depicts a SMILES string and a character-level embedding in a 7-D vector space. Each character is mapped to a vector with 7 dimensions, and each

component in such vector is a real number. In practice, this vector space can have a high dimension, with some models reported in the literature using a dimension of 256. Finding the right dimension size is still much of an experimental issue. The SMILES string in Figure 4 can then be represented as a 9×7 matrix, with each row being a transposed embedding vector. The standard practice in DL is to treat the embedding as another layer in the neural work that can be trained to find the right values for the weights that produce the vectors. Some methods allow *pre-training* these weights on a very large corpus representative of the target data set, and then load them to form a pre-trained embedding layer in the model. This form of *transfer learning* enables researchers to benefit from these pre-trained embeddings and focus on the design and tuning of the followup layers (i.e., for the "downstream task").

3.3 Deep Learning to Predict Molecular Weight from SMILES

The first task that we discuss is the prediction of the molecular weight in a molecule. This is a baseline task since the molecular weight can be estimated as the sum of the weights in all the atoms in the molecule. Thus, our aim here is to determine if a DL model, without any knowledge of chemistry, can learn to predict the molecular weight from the SMILES string. Thus, the first model that we will discuss is a DL model that receives a SMILES string as input and returns its prediction of the molecular weight, as shown in Figure 5.

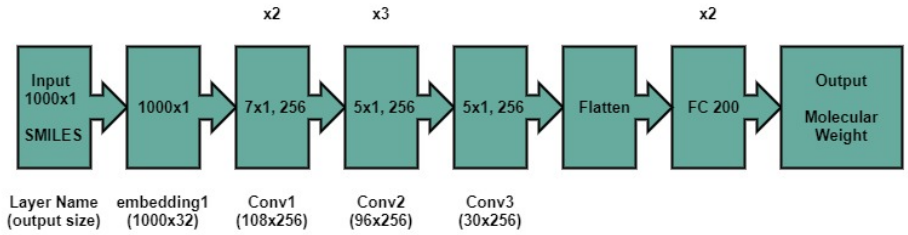

Fig. 5: Model to predict the molecular weight from SMILES.

This model receives as input the pre-processed SMILES strings, capped at a maximum length of 1000 characters[4]. Padding is used to make all SMILES equal in size to 1000 characters. The embedding contains a vocabulary of 25 characters used to represents atoms, bonds, branches, etc. In order to feed the input to the neural network, we first transform each string into an array of the same length. Each entry in this array is an integer representing the index of the character in the vocabulary.

[4]Molecules found in PubChem can be rather large in size.

The input layer of the network feeds the embedding layer, which maps strings into a 32-D vector space. This layer has an embedding matrix of 1000×32. The embedding is then passed to sequence of six (6) 1-D Convs layers with 256 filters per layer. In our figures the symbol of $x2$ and $x3$ mean that the same unit repeats itself two and three times respectively. Notice that the first two 1-D Convs layers are identical; they had a kernel of size 7x1 and were followed by 1-D max pooling of size 3. The next three 1-D Convs layers had kernels with size 5x1 and no pooling. The sixth 1-D Convs layer had kernels of size 5x1 and a 1-D max pooling layer of size 3. The result of the convolutions are then passed to a flatten layer, which in turn is passed to the final three dense layers. The first two dense layers consisted of 200 neural units and the final (output) one had only 1 neural unit. All of the dense layers had a ReLU activation function. The final dense layer gives us the model's prediction of the molecular weight.

3.4 Deep Learning to Predict XLogP from SMILES

The next task that we discuss is the prediction of the XLogP values from the SMILES strings. This task is more challenging than the previous one, and the interested reader can refer to (Cheng et al, 2007) for a discussion on the computational methods currently used in practice to obtain this quantity.

The architecture of our second model can be seen in Figure 6 . This architecture is very similar to the one used for predicting the molecular weight. In fact, the input and embedding layers are the same. The difference consists in the units and the kernel sizes used in the 1-D Convs and dense layers. The embedding layer is once again followed by six (6) 1-D Convs with 256 filters in each. The kernel sizes differ in this model; the first 2 convolutional layers have a kernel size of 5x1 and the kernel size for the remaining four (4) is 3x1. A 1-D max pooling layer of size 3 is used after second and sixth 1-D Conv layers. The two (2) dense layers that follow the flatten layer were changed with respect to the first model, and now consist 100 and 200 neural units respectively. The final dense layer gives us the model's prediction of the XLogP. As in the previous model for molecular weight, all dense layers used ReLU activation.

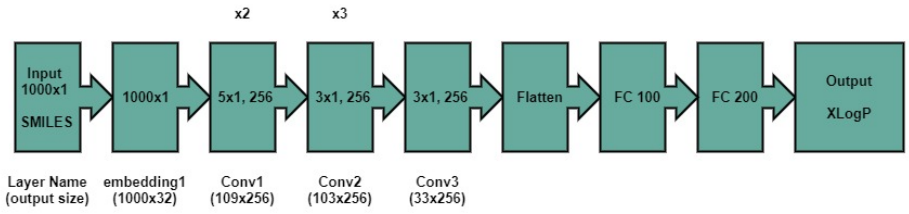

Fig. 6: Model to predict the XLogP from SMILES.

3.5 Deep Learning to Predict XLogP from SMILES and fragments

The third and final task that we discuss is the prediction of XLopP from SMILES and chemical fragments. In this context, fragments are molecules that result from simulated chemical transformations that mimic common reactions carried out in the lab to decompose a molecule into simpler parts. This decomposition can shed light into the functional structure and chemical properties of the molecule. The work in (Chakravarti, 2018) used this idea to explore word embedding for functional chemical fragments (i.e., fragments with well-known chemical properties). We used RDKit (Landrum, 2016) and the RECAP method to create these fragments.

This model, shown in Figure 7, differs from the previous two in that is receives two inputs: 1) the SMILES string for the original molecule and 2) a list of SMILES for the RECAP fragments of the molecule. Due to having two sets of string we needed one embedding layer for each type of input received. In this instance the embedding used a 57-D vector space. All the embeddings are then concatenated and supplied to the 1-D Conv layers that follow. After the concatenation layer, the models follows the exact same setup as the model used for predicting molecular weight. The output from the final dense layer is the predicted XLogP of the chemical compound.

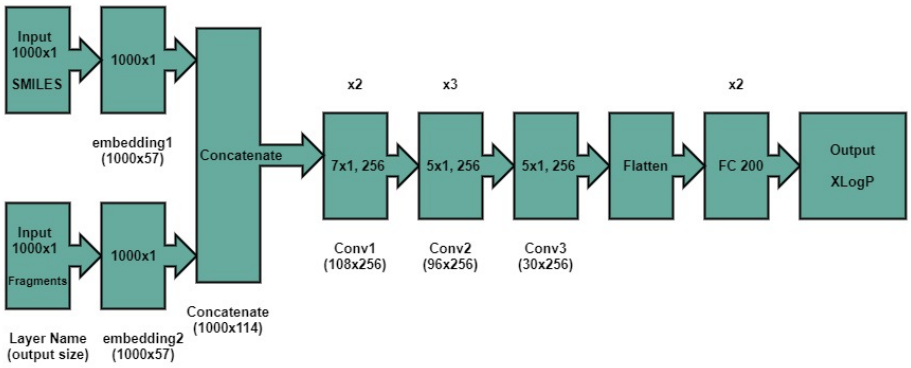

Fig. 7: Model to predict the XLogP from SMILES and RECAP fragments.

4 Experiments

We now present experimental results, including the environment, software and data sets used to obtain these results.

4.1 Computational Environment

All the code used to build the our DL models was written in Python 3.8. We build these models using Tensorflow 2.4, Keras 2.4, Talos (tal, 2019), and Jupyter Notebooks to aid with the visualization. We trained the models on a Dell PowerEdge R940XA Server that had 2 Intel Xeon Gold 5122 processors, 256GB of RAM, 2 NVIDIA Tesla V100 32G Passive GPU, and a 2TB HDD. Only one Tesla V100 GPU was allocated to us for training the models. On this server, we ran our software on a Docker container running atop the Ubuntu 18.04 LTS operating system and with CUDA 11.3 for GPU processing.

4.2 Datasets

All the data used to train and test the models were obtained from the PubChem database (Kim et al, 2021). We used the PubChem REST API (Kim et al, 2018) to capture a data set with ∼1.33 million entries. These entries were randomly chosen from the database. For our training, we then randomly chose 600,000 entries. Each entry contained the following information: CID (the unique identifier used by PubChem), molecular formula, canonical SMILES, isometric SMILES, molecular weight, XLogP, exact mass, TPSA, and complexity.

We used the the canonical SMILES representation for training all the models. It was preferred since it gave us enough information about the molecular structure and ensured that each compound had a unique SMILES string. Recall that the SMILES consisted of the symbols of the elements and special characters with no white-space. Given this, we considered more effective to tokenize the strings at the character level before being given then as input to the model.

To generate the fragments we used the RECAP algorithm from the RDKit (Landrum, 2016) Python library. After generation, the fragments for a molecule were stored as a string where each fragment was separated by a white-space. Most of the SMILES strings had a length of less than 200 characters, with the average length being around 56 characters. But a few of them had length of 1000 characters of more. Most of the fragments generated also had a length of less than 200, with an average length of 144 characters.

The molecular weight and XLogP of the compounds were also important elements as they were necessary to establish how good the predictions of the models were. The molecular weights in the dataset ranged from ∼1 to ∼10,000 g/mol, with the average being at ∼439.44 g/mol. The XLogP in the dataset ranged from ∼-70 to ∼161, with the average being at ∼4.58.

The training set consisted of 600,000 elements of the data and was used for training the models. The validation set consisted of 200,000 elements of the data and was used while training to help fit the hyperparameters of the model. Finally, the test set consisted of 200,000 elements of the data and was used to evaluate the models after training.

4.3 Baseline

As baseline cases for comparing our architectural choices, we developed two basic DL models, one for each property being predicted. Each model contained LSTMs layers instead of 1D Convs. We decided to use LSTMs since it is quite common for them to be used for NLP models. Both baseline models consisted of an embedding layer, 5 LSMT layers and a final Dense layer that outputs the prediction. The embedding layer used worked the same as described in sections 3-C and 3-D for our target models. These models were also trained using the same datasets as our new models proposed in this paper. In Figure 8 we can see the results obtained from the baseline model that predicted molecular weight. The graph on the left side is the loss of the model during training and validation. The graph on the right represents the predictions by the model when given a separate test set.

In the results we can observe that the loss of the model goes down with each epoch suggesting that its predictions are improving in accuracy. However, when we look at the model's performance with the test dataset we notice that regardless of the input it received it would return a similar prediction. This suggests that the model is not recognizing a pattern within the data, but instead it just learned an approximate average value that reduces the error of prediction. Likewise, if we observe the results of the baseline model for XLogP in Figure 9 we can identify a similar trend. In this case we can see that the loss during both training and validation never improved and remained constant. Whereas the results with the test dataset, much like the previous model, just returned a nearly fixed result regardless of input. It would seem that this model also just learned an average value that allows it to minimize the validation loss.

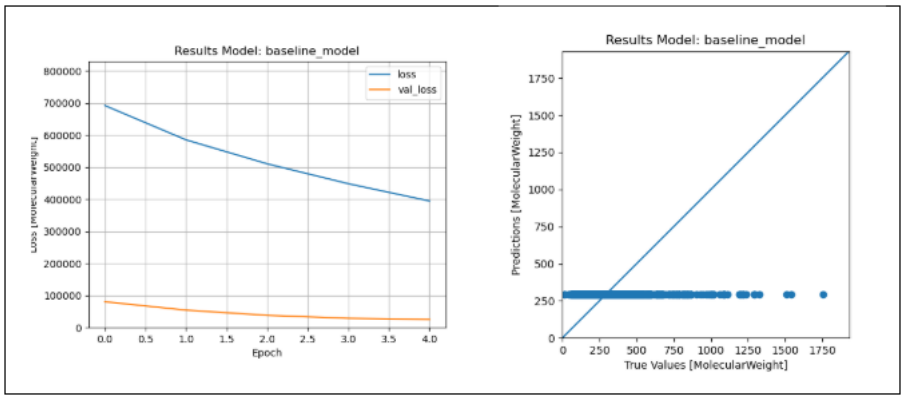

Fig. 8: Loss graph and results of the baseline model for predicting the molecular weight.

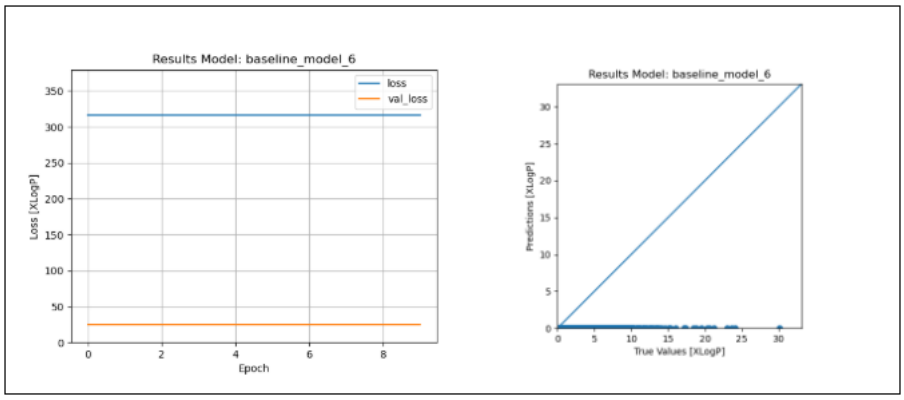

Fig. 9: Loss graph and results of the baseline model for predicting the XLogP.

4.4 Training Our Models

The first model implemented was for predicting the molecular weight of a compound. Initially the padding used for the input data was 300, but was later changed to 1000 since to better accommodate for some of the much longer SMILES strings present in the data set. The results of this model, shown in Figure 10, were promising and led us to proceeded with tuning it. The first plot shows the mean squared error (MSE) of the model for the training and validation data sets. The second plot maps predictions to the true values. To tune the model we used the Python library Talos (tal, 2019), which permitted us to tune various hyperparameters. Some of the initial hyperparameters tuned were: learning rate, epochs, and batch size. Later on, model parameters were included such as the amount of units per layer and the kernel size in the 1D Convs layers. Our main metric for comparison was the resulting validation loss of the models. The overall best model is the one described in section 3-C, which gave us the results shown in Figure 11.

From the training and tuning, we observed that models performed better with smaller molecular weights, but had higher errors with larger weights. This was likely due to the limited data beyond 2,000 g/mol. To address this, we trained models specialized in the 0 to 2,000 g/mol range to see if we could enhance precision in this limited range. The results of the best of those models can be seen in Figure 12.

Using the architecture of the molecular weight model as a reference we proceeded to develop the next two models for predicting the XLogP value. Initially the model had the same architecture as the previous one, hence why they are so similar. But after tuning we reached a model better suited for XLogP. The results can be seen in Figure 13.

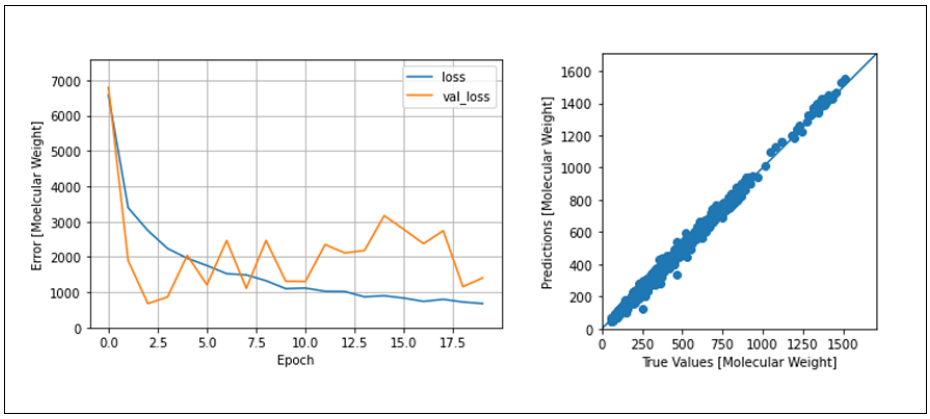

Fig. 10: Results of the first trained model for predicting molecular weight.

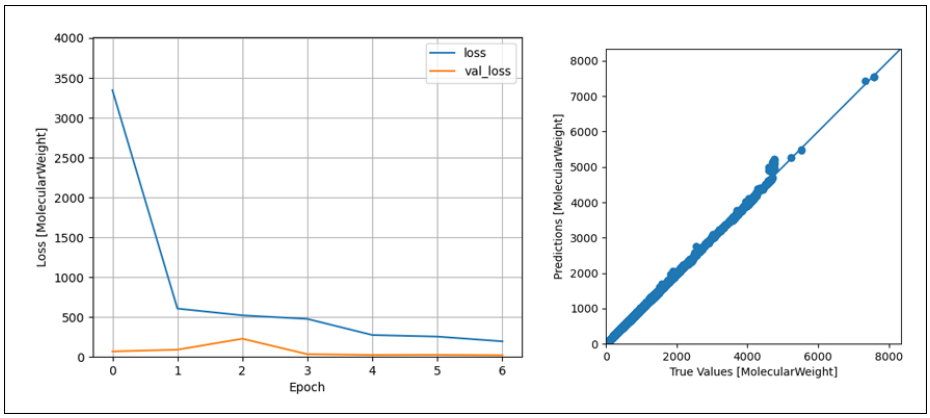

Fig. 11: Results of the best overall tuned model for predicting molecular weight.

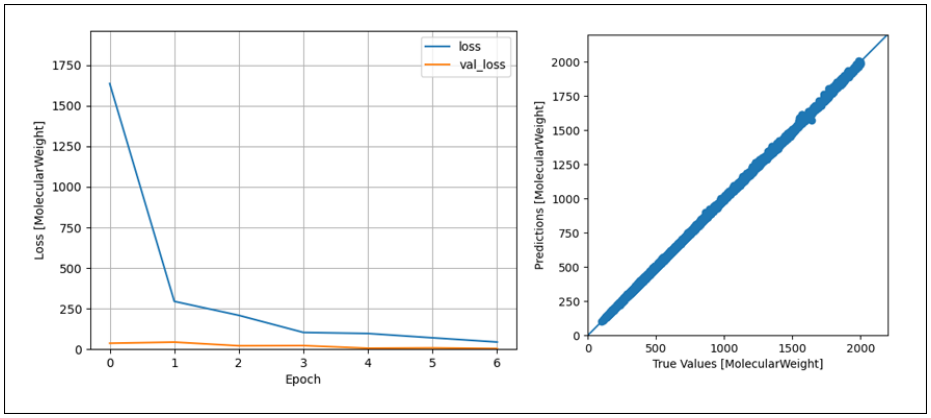

Fig. 12: Results of the best tuned model for predicting molecular weights smaller than 2,000.

Finally, the model that receives as input both SMILES and fragments was implemented similar to the molecular weight model, but was adjusted to receive the additional input. This last model gave us the results shown in Figure 14.

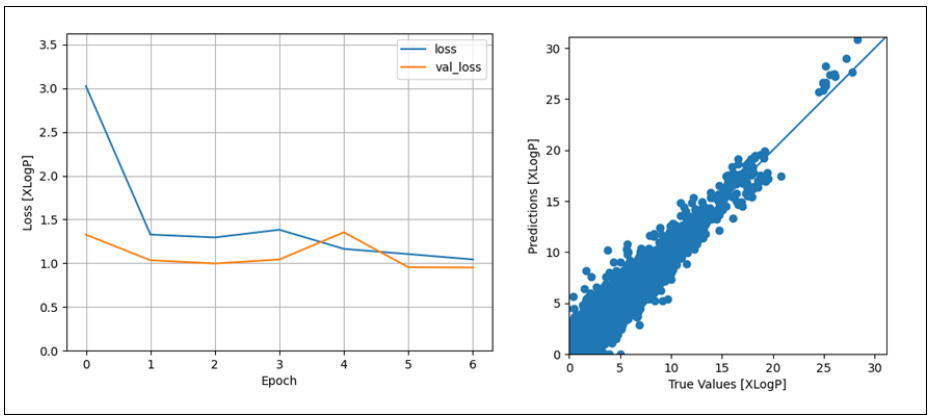

Fig. 13: Results of model trained for prediction XLogP

4.5 Discussion of Results

The graphs shown in Figures 10, 11, and 12, display results from the three molecular weight models, evaluated with the 200,000 sample test set. It can be observed that the second and third model had improved precision compared to the original with predictions closer to the true value represented by the center

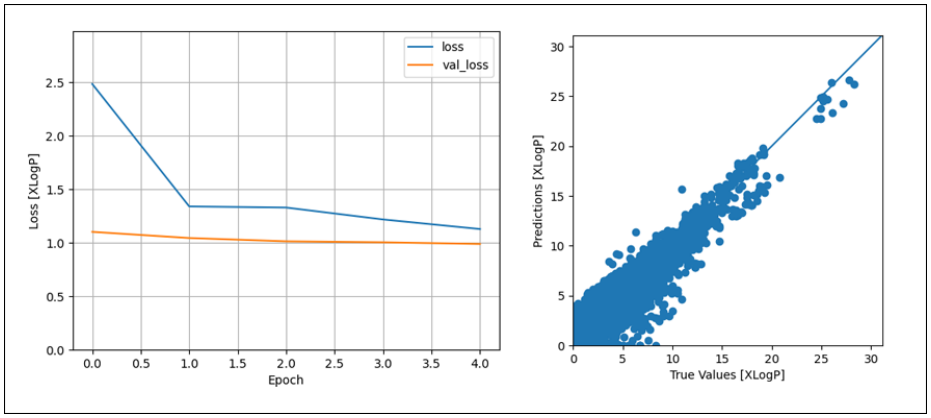

Fig. 14: Results of model trained for prediction XLogP using SMILES and fragments.

line. Notably, these models require relatively few training epochs since training loss dips quickly and the validation loss tends to stay low during training.

To further compare the newer models to the original, we provided each with a small test set containing 500 previously unseen SMILES, all with molecular weights under 2,000 g/mol. The results can be seen in Figure 15. The second model yielded the most accurate results, with an MSE of 37.527, outperforming both the original model and the third model, which was specifically trained for compounds under 2,000 g/mol. This suggests that limiting training data to a subset did not significantly improve performance.

We also compared the two XLogP prediction models. In the validation set results, shown in Figures 13 and 14, both performed similarly. The first model was slightly more distributed on its predictions, while the second tended to underestimate values. We can also observe that during training both models had validation losses hovering near 1, rarely dipping lower. Figure 16 further illustrates their behaviour with a new test set. These models were also given a small test set of 500 new SMILES which included its fragments for the second model. In this test, the model using only SMILES as input significantly outperformed the other, achieving an MSE of 1.996 versus 17.083. Notably, the model trained with fragments struggled with higher XLogP values preferring to stay close to the origin.

One final observation is that the models do not perform well on short SMILES strings. This can be observed in Figures 17 and 18. Here the models were tested using another test set of 500 SMILES strings, that were limited to having between 5 to 20 characters. For all the models the results were more scattered than in previous results. In Figure 17 we see that the second model seems to have a lower bound at around 100 g/mol. Regardless, it still outperformed the other two, but not by a large margin as in previous cases. For the XLogP models we

continue to see that the first model is more distributed in its prediction whereas the second underestimates the values.

If we revisit the baseline models discussed in section 4-D and their results shown in Figures 8 and 9, we see that both models failed to learn any useful information from the SMILES strings, both returning an average value. This becomes more apparent when we compare those results to the ones shown in Figures 10, 11, 12 and 13, which have more precise predictions. With this contrast we can establish that using 1D Convs for processing the embedded data provided an improvement in the results, as opposed to using LSTMs. The use of 1D Convs layers helped the model better understand the data it received allowing it to better correlate the input with the expected output.

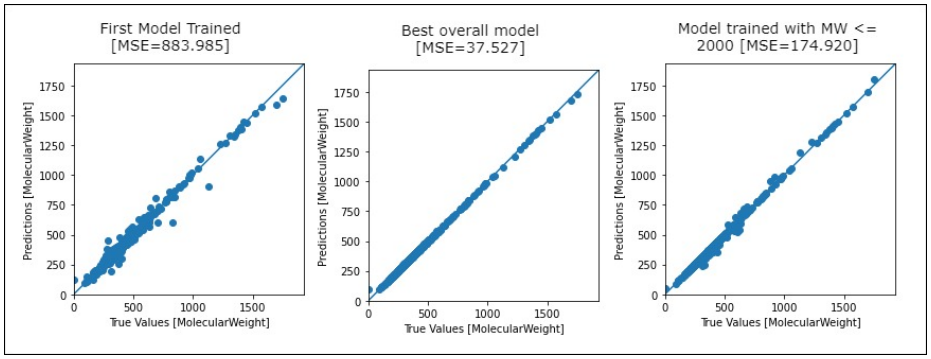

Fig. 15: Comparing the three models that predict the molecular weight.

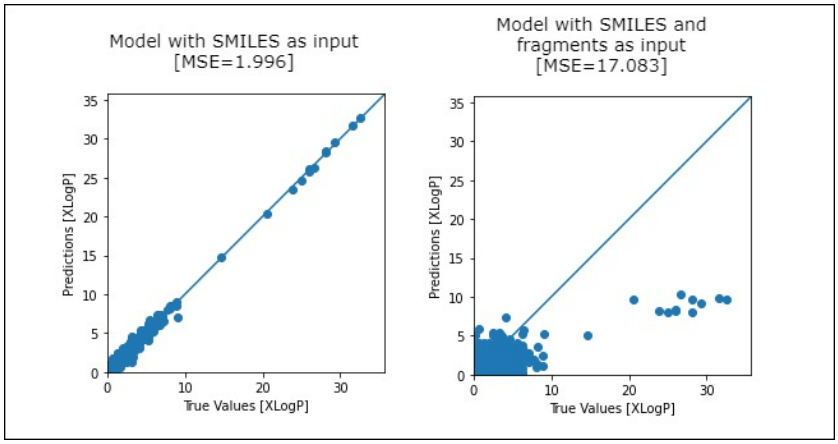

Fig. 16: Comparing the two XLogP architectures implemented.

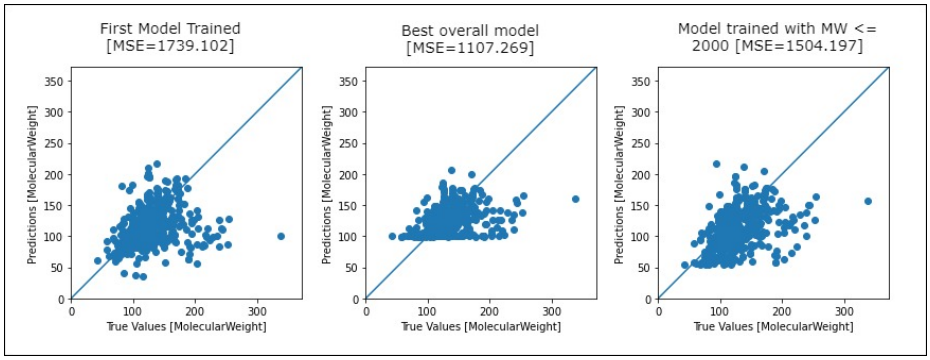

Fig. 17: Comparing results of the three models that predict the molecular on SMILES string between 5 - 20 characters

5 Related Works

The use of machine learning methods for predicting chemical properties of a compound has been previously explored in (Chakravarti, 2018), (Montavon et al, 2012), (Bepler and Berger, 2019), (Sendek et al, 2017), (Xie and Grossman, 2018), (Tshitoyan et al, 2019), (Schmidt et al, 2019). These studies differ greatly from our work in the input type, model architecture, and approach taken to extract the information and in the properties they wanted to predict. Specifically, they use basic regression, support vector machines (SVM), and decision trees to build their models. They also rely heavily on feature engineering to prepare and feed the data to these models. These works have used chemical fragments (Chakravarti, 2018), protein sequences (Bepler and Berger, 2019), Coulomb matrices (Montavon et al, 2012), crystal graphs/structures (Sendek et al, 2017), (Xie and Grossman, 2018), and text (Tshitoyan et al, 2019) as input to their models. The properties predicted have also varied vastly, depending on the application the researchers were aiming for; some properties predicted in these works are the similarity amongst compounds (Chakravarti, 2018), (Tshitoyan et al, 2019), the best material for batteries (Sendek et al, 2017), and energy stored (Montavon et al, 2012), (Xie and Grossman, 2018), among others. NLP techniques based on fully-connected neural networks (Chakravarti, 2018), (Bepler and Berger, 2019), (Tshitoyan et al, 2019) have also been explored previously to predict properties of chemical compounds or to find embeddings for the compound fragments, yielding results centered around their specified domain. Our research architectures were influenced by these previous attempts at using NLP. Our work differs from these approaches because we use convolutional neural networks as the main building block for our models and embeddings. Moreover, our models predict properties using first principle data as input, with no reliance on manual feature engineering efforts. We focused on using SMILES as input, and capitalize on the structural information that is encoded in these strings. Our

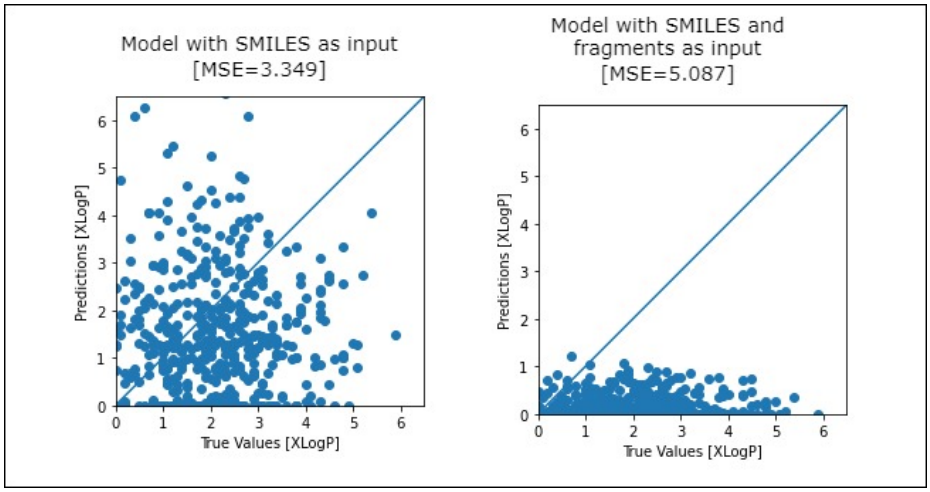

Fig. 18: Comparing results of the two XLogP architectures implemented on SMILES string between 5 - 20 characters.

method can be combined with other input types to explore new models that predict more complex properties. This is the novelty and our key contribution.

6 Conclusions

Machine learning methods have been proposed in lieu of simulations to predict chemical properties. However, many of these methods rely heavily on feature engineering to prepare the data. This limits their impact since targeted expertise in chemistry or related fields is needed to prepare the data. Moreover, the use of molecular structural information have been somewhat limited, despite having such information encoded in the Simplified Molecular Input Line Entry System (SMILES) format. In this paper we have presented a framework that relies on SMILES data to predict molecular properties. Our methods are based on 1-D Convolutional Networks and do not require complex feature engineering. Our methods can be applied to learn molecular properties from base data, making them accessible to a wider audience. Our experiments show that out method can predict molecular weight and XLogP properties without any encoding of complex chemical rules.

Acknowledgements

This material is based upon work supported by the U.S. National Science Foundation under Grant OIA-1849243. Research reported in this publication was also supported by the National Library of Medicine, of the U.S. National Institutes

of Health under award number R15LM012275. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health

Bibliography

(2019) Autonomio talos. URL "http://github.com/autonomio/talos"

Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473, URL http://arxiv.org/abs/1409.0473, 1409.0473

Bepler T, Berger B (2019) Learning protein sequence embeddings using information from structure. CoRR abs/1902.08661, URL http://arxiv.org/abs/1902.08661, 1902.08661

Chakravarti SK (2018) Distributed representation of chemical fragments. ACS Omega 3(3):2825–2836, DOI 10.1021/acsomega.7b02045, URL https://doi.org/10.1021/acsomega.7b02045, https://doi.org/10.1021/acsomega.7b02045

Cheng T, Zhao Y, Li X, Lin F, Xu Y, Zhang X, Li Y, Wang R, Lai L (2007) Computation of octanol-water partition coefficients by guiding an additive model with knowledge. Journal of Chemical Information and Modeling 47(6):2140–2148, DOI 10.1021/ci700257y, URL https://doi.org/10.1021/ci700257y, pMID: 17985865

Conneau A, Lample G, Rinott R, Williams A, Bowman SR, Schwenk H, Stoyanov V (2018) : Evaluating cross-lingual sentence representations URL https://arxiv.org/pdf/1809.05053.pdf, 1809.05053

Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, pp 4171–4186, DOI 10.18653/v1/N19-1423, URL https://www.aclweb.org/anthology/N19-1423

Jozefowicz R, Vinyals O, Schuster M, Shazeer N, Wu Y (2016) Exploring the limits of language modeling. URL http://arxiv.org/abs/1602.02410, cite arxiv:1602.02410

Kim S, Thiessen PA, Cheng T, Yu B, Bolton EE (2018) An update on PUG-REST: RESTful interface for programmatic access to PubChem. Nucleic Acids Research 46:W563—W570, DOI doi.org/10.1093/nar/gky294, URL https://doi.org/10.1093/nar/gky294

Kim S, Chen J, Cheng T, Gindulyte A, He J, He S, Li Q, Shoemaker BA, Thiessen PA, Yu B, Zaslavsky L, Zhang J, Bolton EE (2021) PubChem in 2021: new data content and improved web interfaces. Nucleic Acids Research 49:D1388—D1395, DOI 10.1093/nar/gkaa971, URL https://doi.org/10.1093/nar/gkaa971

Kim Y, Jernite Y, Sontag D, Rush AM (2016) Character-aware neural language models. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI Press, AAAI'16, p 2741–2749

Kirklin S, Saal JE, Meredig B, Thompson A, Doak JW, Aykol M, Rühl S, Wolverton C (2015) The Open Quantum Materials Database (OQMD): assessing the accuracy of DFT formation energies. npj Computational Materials 1, DOI 10.1038/npjcompumats.2015.10, URL https://doi.org/10.1038/npjcompumats.2015.10

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, Curran Associates Inc., NIPS'12, p 1097–1105

Landrum G (2016) Rdkit: Open-source cheminformatics software URL https://github.com/rdkit/rdkit/releases/tag/Release$_2$016$_0$9$_4$

Ling W, Dyer C, Black AW, Trancoso I, Fermandez R, Amir S, Marujo L, Luís T (2015) Finding function in form: Compositional character models for open vocabulary word representation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, pp 1520–1530, DOI 10.18653/v1/D15-1176, URL https://www.aclweb.org/anthology/D15-1176

Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 3431–3440, DOI 10.1109/CVPR.2015.7298965

Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. CoRR abs/1301.3781, URL http://arxiv.org/abs/1301.3781, 1301.3781

Montavon G, Hansen K, Fazli S, Rupp M, Biegler F, Ziehe A, Tkatchenko A, von Lilienfeld OA, Müller KR (2012) Learning invariant representations of molecules for atomization energy prediction. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, Curran Associates Inc., Red Hook, NY, USA, NIPS'12, p 440–448

Pennington J, Socher R, Manning CD (2014) Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP), pp 1532–1543, URL http://www.aclweb.org/anthology/D14-1162

Rajan K (2005) Materials informatics. Materials Today 8(10):38–45, DOI https://doi.org/10.1016/S1369-7021(05)71123-8, URL https://www.sciencedirect.com/science/article/pii/S1369702105711238

Saal JE, Kirklin S, Aykol M, Meredig B, Wolverton C (2013) Materials design and discovery with high-throughput density functional theory: The open quantum materials database (OQMD). JOM 65:1501–1509, DOI 10.1007/s11837-013-0755-4, URL https://doi.org/10.1007/s11837-013-0755-4

Schmidt J, Marques MRG, Botti S, Marques MAL (2019) Recent advances and applications of machine learning in solid-state materials science. npj Computational Mathematics 5:83, DOI 10.1038/s41524-019-0221-0

Sendek AD, Yang Q, Cubuk ED, Duerloo KAN, Cui Y, Reed EJ (2017) Holistic computational structure screening of more than 12000 candidates for solid lithium-ion conductor materials. Energy Environ Sci 10:306–320, DOI 10.1039/C6EE02697D, URL http://dx.doi.org/10.1039/C6EE02697D

Szegedy C, Liu W, Jia Y, Sermanet P, Reed SE, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2014) Going deeper with convolutions. CoRR abs/1409.4842, URL http://dblp.uni-trier.de/db/journals/corr/corr1409.htmlSzegedyLJSRAEVR14

Tshitoyan V, Dagdelen J, Weston L, Dunn A, Rong Z, Kononova O, Persson KA, Ceder G, Jain A (2019) Unsupervised word embeddings capture latent knowledge from materials science literature. Nature 571:95–98, DOI 10.1038/s41586-019-1335-8

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser u, Polosukhin I (2017) Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA, NIPS'17, p 6000–6010

Xie T, Grossman JC (2018) Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. Physical Review Letters 120(14), DOI 10.1103/physrevlett.120.145301, URL http://dx.doi.org/10.1103/PhysRevLett.120.145301

Zhang X, LeCun Y (2015) Text understanding from scratch. URL http://arxiv.org/abs/1502.01710, cite arxiv:1502.01710