Bioinformatics, 2019, 1–7
doi: 10.1093/bioinformatics/btz307
Advance Access Publication Date: 9 May 2019
Original Paper

Structural bioinformatics

FP2VEC: a new molecular featurizer for learning molecular properties

Woosung Jeon and Dongsup Kim*

Department of Bio and Brain Engineering, Korea Advanced Institute of Science and Technology, Yuseong-gu, Daejeon 34141, Republic of Korea

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on November 28, 2018; revised on March 28, 2019; editorial decision on April 22, 2019; accepted on April 24, 2019

Abstract

Motivation: One of the most successful methods for predicting the properties of chemical compounds is the quantitative structure–activity relationship (QSAR) methods. The prediction accuracy of QSAR models has recently been greatly improved by employing deep learning technology. Especially, newly developed molecular featurizers based on graph convolution operations on molecular graphs significantly outperform the conventional extended connectivity fingerprints (ECFP) feature in both classification and regression tasks, indicating that it is critical to develop more effective new featurizers to fully realize the power of deep learning techniques. Motivated by the fact that there is a clear analogy between chemical compounds and natural languages, this work develops a new molecular featurizer, FP2VEC, which represents a chemical compound as a set of trainable embedding vectors.

Results: To implement and test our new featurizer, we build a QSAR model using a simple convolutional neural network (CNN) architecture that has been successfully used for natural language processing tasks such as sentence classification task. By testing our new method on several benchmark datasets, we demonstrate that the combination of FP2VEC and CNN model can achieve competitive results in many QSAR tasks, especially in classification tasks. We also demonstrate that the FP2VEC model is especially effective for multitask learning.

Availability and implementation: FP2VEC is available from https://github.com/wsjeon92/FP2VEC.

Contact: kds@kaist.ac.kr

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

In recent years, there has been a growing interest in developing machine learning methods to predict the various properties of chemical compounds (Lavecchia, 2015). Quantitative structure–activity relationship (QSAR) models represent one of the most successful methods. The principle behind the QSAR methods is that structurally similar chemicals should have similar properties (Tropsha, 2010). QSAR methods have played a vital role in drug discovery, especially in lead compound generation by virtual screening (Shoichet, 2004) and the drug's ADME (adsorption, distribution, metabolism and excretion) property optimization (Lipinski *et al.*, 2001). Another important application of QSAR methods is computational toxicity prediction, which has been attracting substantial attention recently for an attempt to replace expensive and controversial toxicology experiments on animal models (Luechtefeld *et al.*, 2018).

The prediction accuracy of QSAR models has recently been greatly improved by employing deep learning technology (Capuzzi *et al.*, 2016; Duvenaud *et al.*, 2015; Kearnes *et al.*, 2016; Mayr *et al.*, 2016; Wójcikowski *et al.*, 2018). The advent of deep learning in the drug development field occurred in 2013 when the QSAR machine learning challenge on chemical compound activity in drug discovery organized by Merck (Kaggle challenge) was won by Hinton's group, a pioneer in deep learning technology. They achieved 14% better prediction accuracy over conventional QSAR methods. Since then, deep learning methods for drug development have attracted

huge attention from both researchers and pharmaceutical companies. For example, deep learning models made better predictions than the random forest method on a set of large diverse QSAR datasets (Ma *et al.*, 2015). Other examples include boosting docking-based virtual screening with deep learning (Pereira *et al.*, 2016), multitask deep neural networks in QSAR studies (Xu *et al.*, 2017), generating focused chemical libraries using a recurrent neural network (Segler *et al.*, 2018) and the *de novo* generation of new molecules using generative models (Kadurin *et al.*, 2017; Sanchez-Lengeling and Aspuru-Guzik, 2018). In addition, there is also an open-source programming tool called DeepChem that aims to popularize the deep learning methods in drug discovery and computational biology (Wu *et al.*, 2018).

Natural language processing (NLP) is a field that studies human languages and imitates them using various computational methods. In NLP, one expresses words as some type of numerical values such as *n*-dimensional vectors (Collobert *et al.*, 2011). For example, using a popular NLP model such as the Word2Vec model (Mikolov *et al.*, 2013a,b), one can compute the semantic relationship between words after those words are represented by trainable vectors (word embedding vectors). There are many attempts to apply NLP techniques to biological data. ProtVec (Asgari and Mofrad, 2015) uses a Word2vec model to model biological sequences such as DNA and protein sequences. ProtVec converts the biological sequence into vector representations and predicts the protein property using NLP techniques. Mol2vec (Jaeger *et al.*, 2018) also applies the Word2vec model to predict the various molecular properties from molecular structure information. Mol2vec expresses a molecular structure as a vector representation similar to the molecular fingerprint vector using a list of molecular substructures. The SMILES2VEC model introduces a direct conversion from SMILES representation to embedding vectors (Goh *et al.*, 2017). The NLP technique is also a powerful tool for expressing biological data in numerical form and can be used to calculate the semantic meaning of the data.

The convolutional neural network (CNN) is one of the neural network models specially designed to capture local features of data via convolution operation (Goodfellow *et al.*, 2016). The CNN model is powerful in analyzing grid-like data such as images (He *et al.*, 2016; Lecun *et al.*, 1998). In addition, CNN has been shown to be effective in analyzing sequential data such as sentences and DNA sequences (Alipanahi *et al.*, 2015; Collobert *et al.*, 2011; Kalchbrenner *et al.*, 2014; Shen *et al.*, 2014; Yih *et al.*, 2014). Moreover, the CNN method can be extended to model graph-like structures such as protein–protein interaction networks. Utilizing the fact that a chemical structure can be represented as a graph, a new molecular featurizer called the graph convolution featurizer has been developed (Duvenaud *et al.*, 2015; Kearnes *et al.*, 2016). Several featurizers based on graph convolutions outperform the conventional extended connectivity fingerprints (ECFP) feature (Rogers and Hahn, 2010) in both classification and regression tasks (Wu *et al.*, 2018), indicating that it is critical to develop more effective featurizers to fully realize the power of deep learning techniques.

In this study, we introduce a new molecular featurizer, FP2VEC, which represents a chemical compound as a set of trainable embedding vectors. This work is motivated by the fact that there is a clear analogy between chemical compounds and natural languages (Cadeddu *et al.*, 2014). The chemical compounds can be expressed as a set of the molecular substructures that are analogous to the words in texts. Like other vector embedding methods, our method converts the nonvectorial data (set of molecular substructures) into numerical vectors in Euclidean space (embedding vectors). A particular molecular substructure is represented as a specific integer by the molecular fingerprint generation algorithm.

To implement and test our new featurizer, we built a QSAR model using a CNN architecture. Especially, we adopt a simple CNN architecture that has been successfully used for NLP classification tasks such as sentence classification (Kim, 2014). In our QSAR model, the CNN is specialized in capturing the important features from the embedding vectors. By testing our new method on several benchmark datasets, we demonstrate that the combination of the FP2VEC and the CNN can improve the prediction accuracy of many QSAR tasks, especially classification tasks.

This article is organized as follows. First, Section 2 explains the benchmark models, the datasets, the algorithm of the fingerprint embedding method, the network structure of the QSAR model and the evaluation criteria. Next, Section 3 analyzes the prediction performance of our new featurizer on several benchmark datasets and compare it with those of other neural network-based featurizers. Finally, Section 4 summarizes the results of the research.

2 Materials and methods

2.1 Benchmark featurizers and datasets

2.1.1 Benchmark featurizers and models

To evaluate the performance of a new featurizer, FP2VEC, we compare our QSAR model with the benchmark results of several neural network-based prediction models that use different molecular featurizers: ECFP (Rogers and Hahn, 2010), graph convolution (Duvenaud *et al.*, 2015) and the weave featurizer (Kearnes *et al.*, 2016). For the classification tasks, four benchmark models are derived from MoleculeNet (Wu *et al.*, 2018). Those prediction models include the fully connected neural network (FCNN) model, bypass multitask network model (Bypass), graph convolution (GC) model and weave (Weave) model. The FCNN and Bypass models use the ECFP featurizer. The GC model uses the GC featurizer, and the weave model uses the Weave featurizer. For the regression tasks, the benchmark results are obtained from two studies: MoleculeNet and the graph convolution study (Duvenaud *et al.*, 2015). From MoleculeNet, three benchmark results (FCNN, GC and Weave models) are obtained. Additionally, we use four benchmark results from the graph convolution study: two GC featurizer models (GraphConv) with the linear/neural network and two ECFP models (ECFP) with the linear/neural network.

2.1.2 Datasets

We use four datasets (Tox21, HIV, BBBP and SIDER) for the classification tasks and five datasets (Malaria, CEP, ESOL, FreeSolv and Lipophilicity) for the regression tasks. All of the chemical compounds in the datasets are represented as SMILES codes. The datasets are prepared in the same way as the benchmark studies for fair comparison. We divide the datasets into the training set, validation set and a test set with an 8:1:1 ratio. We train the prediction models with training sets and optimize the models by choosing the model hyperparameters that maximize the prediction accuracies on validation sets. Finally, the prediction performances are measured using those optimized models on the test sets. We repeat the same procedures five times and report the average and standard deviation of each task. Detailed information on the datasets is provided in Supplementary Table S1.

Tox21 dataset. The Tox21 dataset is experimentally measured toxicity against 12 targets (Supplementary Table S2). The Tox21 dataset contains 8014 compounds and corresponding toxicity data against 12 targets. The label, toxicity, is described as binary: 1 if the compound has toxicity or 0 otherwise. The dataset is divided by the

random split method, which randomly splits the dataset into training, validation and test sets.

SIDER dataset. The SIDER dataset contains marketed drugs and their adverse drug reactions (ADR) against 27 System-Organs Class (http://www.meddra.org/). The ADR is recorded with binary labels. The dataset has a total of 1427 data points for 27 targets. For the SIDER dataset, we use the random split method.

HIV dataset. The HIV dataset is an experimental measurement of the ability to inhibit HIV replication. The HIV dataset contains 41 127 compounds and their ability of inhibition with binary labels. The HIV dataset is divided using the scaffold split method. The scaffold split method divides the dataset into a training set, a validation set and a test set according to their two-dimensional molecular structures (Bemis and Murcko, 1996). It provides a more difficult task than the random split dataset. The scaffold split dataset is implemented by the RDKit: Open-source cheminformatics (http://www.rdkit.org/) library.

BBBP dataset. The BBBP dataset is blood–brain barrier penetration with binary labels. The dataset has a total of 2050 compounds. For the BBBP dataset, we use the scaffold split method.

ESOL dataset. The ESOL dataset includes measurements of the water solubility of small compounds (Delaney, 2004). Water solubility is represented as a measured log solubility in moles per liter. The ESOL dataset includes 1128 compounds and their water solubility. For the ESOL dataset, we use the random split method.

FreeSolv dataset. The FreeSolv dataset contains the hydrogen-free energy of small compounds in a water environment measured by experiment and computer simulation. The dataset contains 642 molecules and their hydrogen-free energy. For the FreeSolv dataset, we use the random split method.

Lipophilicity dataset. The Lipophilicity dataset contains an octanol/water distribution coefficient at pH 7.4 measured experimentally. The dataset has 4200 compounds and their corresponding values. For the Lipophilicity dataset, we use the random split method.

Malaria dataset. The Malaria dataset includes the experimentally measured half-maximal effective concentration (EC50) values of a sulfide-resistant strain of *Plasmodium falciparum*, which is the source of malaria (Gamo *et al.*, 2010). The Malaria dataset has 9998 compounds and their EC50 values. For the Malaria dataset, we use the random split method.

CEP dataset. The CEP (Clean Energy Project) dataset includes the candidate molecules that are suitable for solar cell materials (Hachmann *et al.*, 2011). The CEP dataset has 29 978 compounds and corresponding CEP values. For the CEP dataset, we use the random split method.

2.2 Featurizer and QSAR model

2.2.1 Fingerprint embedding featurizer

The key concept of the fingerprint embedding featurizers is derived from NLP. There is a clear analogy between chemical compounds and natural languages (Cadeddu *et al.*, 2014). In NLP, a text (sentence or document) can be thought of as a collection of words, and each word in a text is expressed as a numerical vector (word embedding vector). Analogously, a chemical compound can be represented as a set of molecular substructures (molecular fingerprints), and each substructure is expressed as a vector (fingerprint embedding vector). Thus, in this work, we assume that a chemical compound can be represented by a set of fingerprint embedding vectors just as a text can be represented by a collection of word embedding vectors. Once this representation of chemical compounds analogous to the text representation in NLP is established, many useful techniques developed in NLP, such as CNN, can be utilized to predict the various molecular properties.

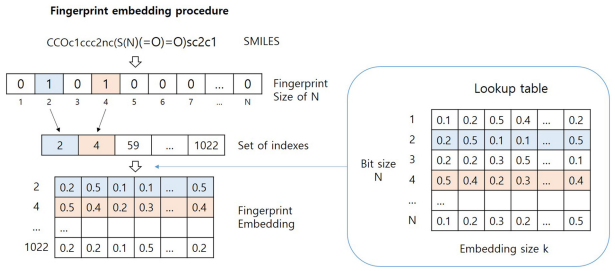

Fig. 1. Schematic overview of the FP2VEC featurizer generation process

Figure 1 shows a schematic overview of the whole fingerprint embedding process. First, we extract the molecular substructures from a SMILES representation of a chemical compound. Specifically, we generate the 1024 bit Morgan (or circular) fingerprint of a radius of 2 by using the RDKit. We have tried 2048 bit or full-size ('unfolded') fingerprints but found that the size of the fingerprint vectors does not affect the model's performance. After that, we collect the fingerprint indices, which are marked as '1' in the fingerprint vector. We then express the features of the molecular structure as a list of integers where each integer represents a specific molecular substructure. These integers are analogous to the word indices of texts. Next, we build the lookup table to represent each integer index as a vector of finite size (embedding size). The lookup table is a two-dimensional matrix whose size is the bit size times embedding size. Each row of the lookup table provides a unique embedding vector corresponding to each integer of the Morgan fingerprint. At the initial state, the lookup table is initialized with random values. Through the training process, the values of the lookup table are fine-tuned to maximize the specific objective of the training. For example, as shown in Figure 1, the embedding vector for fingerprint 2 is initialized with random values such as [0.2, 0.5, 0.1, …, 0.5]. These random fingerprint embedding vectors are fed into a particular QSAR model and then optimized through the training process by adjusting their values to maximize the model prediction accuracy. These changes are also reflected in the lookup table. Through this process, we obtain the task-specific vector representation of compounds, called the fingerprint embedding matrix, for the given tasks. This task-specific lookup table can provide more useful information than conventional circular fingerprint itself. During the test sessions, we use the fingerprint embedding matrix for the predictions.

2.2.2 Structure of the QSAR model using a simple CNN architecture

The QSAR model is built with a simple CNN architecture. The network structure of the model is inspired by the sentence classification model (Kim, 2014). Figure 2 shows the summary of the network structure of our QSAR model. Before training the model, the SMILES representations are converted into the fingerprint embedding matrix. Let $x \in \mathbb{R}^{lk}$ be the fingerprint embedding matrix, where l is the number of 1 bits in a fingerprint vector and k the embedding size. For the mini batch operation, we pad the fingerprint embedding matrix by 0 with a max length m ($\geq l$) along the fingerprint dimension. Thus, the padded fingerprint embedding matrices are all the same size, which is $x_{\text{pad}} \in \mathbb{R}^{mk}$. Then, the padded fingerprint embedding matrix x_{pad} is fed into the model as input data.

The model is composed of three parts: a two-dimensional convolutional layer (Conv2d), a max pooling and a dropout layer, and a

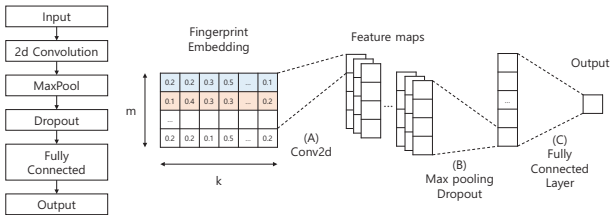

Fig. 2. (Left) The network structure and (right) data flow of the QSAR model. The model uses a padded fingerprint embedding matrix as input data. The model consists of (**A**) a two-dimensional convolutional layer, (**B**) a max pooling and dropout layer and (**C**) a fully connected layer.

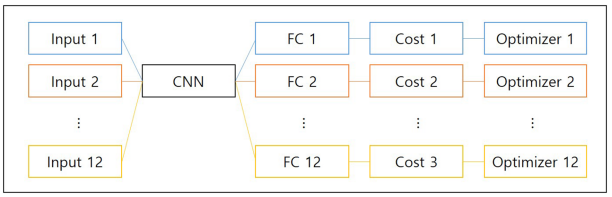

Fig. 3. The network structure of a multitask learning method for the Tox21 dataset. The Tox21 dataset has 12 different targets. Each (input, target) pair is grouped with the same color. Here, each target has its own input data (Input), a fully connected layer (FC), a cost function (Cost) and an optimizer while sharing the same CNN model parameters.

fully connected layer. First, the Conv2d layer is applied to the input data,

$$\mathbf{c} = \text{Conv2d}(x_{\text{pad}} \circledast w_{\text{conv}}),$$

where $\circledast$ stands for the convolution operator. Each filter w_{conv} with a window size $h \times k$ is denoted as $w_{\text{conv}} \in \mathbb{R}^{hk}$. The convolution filters only move along the substructure dimension, not the embedding dimension. By the convolutional operation, the n filters would capture the significant features from the input data. The convolution operation then produces the n feature maps $\mathbf{c}$ by n different filters, where each feature $c_i \in \mathbb{R}^{m-h+1}$, $i = 1, \ldots, n$. The entire feature map is expressed as $\mathbf{c} \in \mathbb{R}^{(m-h+1,n)}$. After the convolutional operation, we add the bias $b_{\text{conv}} \in \mathbb{R}^{(m-h+1,n)}$ to the feature maps and apply the rectifier linear unit (ReLU) function for the nonlinear activation (Glorot *et al.*, 2011),

$$\mathbf{c}_{\mathbf{relu}} = \text{ReLU}(\mathbf{c} + b).$$

Next, the max-over-time pooling operation (Collobert *et al.*, 2011) in the max pooling layer extracts important features from feature maps. The max pooling layer picks up the maximum value from the $\mathbf{c}_{\mathbf{relu}}$.

$$\mathbf{c}_{\mathbf{max}} = \max(\mathbf{c}_{\mathbf{relu}}), \quad \mathbf{c}_{\mathbf{max}} \in \mathbb{R}^n.$$

After that, the dropout (Srivastava *et al.*, 2014) is applied with a dropout rate of 0.5 to prevent overfitting during a training session,

$$\mathbf{c}_{\mathbf{drop}} = \text{dropout}(\mathbf{c}_{\mathbf{max}}).$$

During an evaluation session, the dropout is not applied to $\mathbf{c}_{\mathbf{max}}$. After that, we concatenate n $\mathbf{c}_{\mathbf{drop}}$ as a one-dimensional matrix. Finally, the fully connected layer yields the model output. The model output becomes the prediction of the QSAR model,

$$y = \mathbf{c}_{drop} \cdot w_{fc} + b_{fc},$$

where $w_{fc} \in \mathbb{R}^n$ is the weight for the fully connected layer and $b_{fc} \in \mathbb{R}$ is the bias for the fully connected layer. Through the training session, the values of the lookup table and the network parameters are adjusted for better prediction. During the test session, the model uses the trained values for prediction.

2.2.3 Output and evaluation

For the classification tasks, we apply the sigmoid activation function to the output. We also used the logarithmic loss function and Adam optimizer (Kingma and Ba, 2014) as an optimizer. The predictions are evaluated by the area under the receiver operating characteristic curve (ROC-AUC) scores. The ROC-AUC scores are measured by taking the average of ROC-AUC values from five independent trials. For the regression tasks, we use the mean-squared error loss function and the Adam optimizer as an optimizer. The predictions are evaluated by the root mean square error (RMSE) scores. The RMSE scores are measured by taking the average from the RMSE values of five independent trials.

2.2.4 Hyperparameters

FP2VEC has one hyperparameter, the embedding size k, while the QSAR model has two hyperparameters, the window size of the filter h, and the size of feature map n. We test many different combinations of hyperparameters to select the optimal choice. We have tested $k = 100, 200, 300$, $h = 1, .., 7$, $n = 512, 1024, 2048$, as well as the learning rate = 1e-3, 5e-4, 1e-4, 5e-5, 1e-5. We select the best set of hyperparameters for each task. The performance of the model is not sensitive to the changes in these hyperparameters. The hyperparameters are described in Supplementary Table S3.

2.2.5 Multi-task learning

In case of the Tox21 and the SIDER datasets, a single compound is associated with multiple targets. In such cases, the multitask learning model shows a better prediction than the single task learning model (Mayr *et al.*, 2016; Ramsundar *et al.*, 2015). In case of the single task learning, each target has a separate CNN model. For example, for the tox21 dataset, there are 12 individual CNN models for 12 different targets. Thus, each CNN model is trained by different input data. For the multi-task learning, however, there is only one CNN model for all 12 targets (Fig. 3). In the multitask learning scheme, the 12 different targets share the parameters of the CNN model architecture. By sharing parameters, the CNN model can capture the general features of chemical compounds across the targets. The separated fully connected layers for each target then learn specific features of the compounds for each target. Using both a single CNN model and separated fully connected layer, a multitask learning model can improve prediction accuracy.

3 Results and discussion

3.1 Classification tasks

For the classification tasks, the prediction performances of our QSAR model using the FP2VEC featurizer are compared with those of four other neural network-based methods: the FCNN and Bypass model using the ECFP featurizer, the GC model using the Graph Convolution featurizer (GC) and the Weave model using the Weave featurizer (Weave). The datasets and performances of other methods are taken from MoleculeNet (Wu *et al.*, 2018). For the multiple target datasets (Tox21, SIDER), we also apply multitask learning. For fair comparison, the datasets are prepared in the same way as the benchmark studies. We measured the ROC-AUC scores of the test sets to evaluate the prediction accuracy of the QSAR model.

The means and standard deviations of the ROC-AUC scores are measured by five independent trials.

The detailed results, including the validation/test sets, are described in Supplementary Table S4. For the Tox21 and SIDER datasets, the ROC-AUC scores of our model are assessed with the random split method. The ROC-AUC scores of the validation sets on the Tox21 and SIDER datasets show similar results compared to those of the test sets (Supplementary Table S4). The HIV and BBBP datasets are divided by the scaffold split method, which increases the structural differences of the compounds among the training, validation and test sets. Scaffold split offers a more difficult evaluation setting. The ROC-AUC scores of the validation sets on the HIV and BBBP datasets (0.800 and 0.960, respectively) are much higher than those of the test sets (0.754 and 0.713, respectively).

Table 1 shows the mean ROC-AUC scores and standard deviations on test sets for the Tox21, HIV, BBBP and SIDER datasets. The Tox21 dataset has 12 targets, and the SIDER dataset has 27 targets. Therefore, these two datasets are ideal for testing how much performance gain we can achieve by applying multitask learning methods. For the single task on the Tox21 dataset, the ROC-AUC score of our method (0.825) is nearly identical to that of the highest score by the GC method (0.829), while our method achieved a much higher score (0.732) on the SIDER dataset than that of the second highest score (0.673) by Bypass. However, the FP2VEC model with the multitask learning achieved much higher ROC-AUC scores on both the Tox21 and SIDER datasets compared to the single task learning. To compare those scores with those of other featurizers and methods, we obtain the results from a recent publication (Feinberg *et al.*, 2018) where multitask learning scores on Tox21 by various featurizers, including a new featurizer, PotentialNet, are reported. According to that paper, PotentialNet, Weave and GC achieve ROC-AUC scores of 0.857, 0.831, and 0.838, respectively. Compared to those scores, FP2VEC achieves the highest ROC-AUC score (0.876), which suggests that FP2VEC is highly effective for the multitask learning tasks. The results of the prediction accuracy of individual targets on the Tox21 dataset are provided in Supplementary Table S5, and the results of the prediction accuracy of the SIDER dataset against individual targets are provided in Supplementary Table S6. As shown in Supplementary Tables S5 and S6, the ROC-AUC scores of the individual targets in the multitask model are primarily increased compared to the single task models in the Tox21 and SIDER datasets.

Both the HIV and BBBP datasets have only one target; thus, the prediction model is a single task learning model. On the HIV dataset, the FP2VEC model shows second-best performance among the benchmark results. The ROC-AUC score of our model (0.754) is slightly lower than the best-performing GC model (0.753). For the BBBP dataset, our model shows the highest ROC-AUC score. The HIV and BBBP datasets use a scaffold split method. The scaffold split dataset is more difficult than the random split dataset because the training, validation and test sets have different types of two-dimensional molecular structures. These results indicate that even if the training and the test set have different characteristics related to the molecular structures, the FP2VEC method can learn the general feature of the compounds.

Overall, benchmark tests on the classification tasks demonstrate that our model achieves superior prediction performance when compared with the ECFP, Graph Convolution and Weave featurizer models. We also demonstrate that the FP2VEC model is especially effective for the multitask learning tasks. In addition, the GC and Weave display weakness when the dataset is relatively small, such as in the BBBP and SIDER datasets. However, the FP2VEC featurizer consistently achieves reliable performance regardless of the size of the dataset.

3.2 Regression tasks

Regarding the regression tasks, we compare the prediction performance of our model (FP2VEC) with two different studies. For the ESOL, FreeSolv and Lipophilicity datasets, the results are compared with the three benchmark models (FCNN, GC and Weave model) from MoleculeNet. For the other two datasets, the Malaria and CEP, the results are compared with the four models published in the original graph convolution paper by Duvenaud *et al.* (2015). The four models are the neural network and linear models with the graph convolution featurizer (GraphConv) and the neural network and linear models with the ECFP featurizer. We measure the RMSE scores on the test sets to evaluate the prediction accuracy of the models. The means and standard deviations of the RMSE scores are also measured by five independent trials. Detailed results, including the validation/test sets, are described in Supplementary Table S7. Regarding the classification tasks, all of the tasks are evaluated by the random split method. The results of the validation sets and test sets are very similar (Supplementary Table S7).

Table 2 shows the RMSE scores on the test sets against the ESOL, FreeSolv and Lipophilicity datasets. On three datasets, our model shows a better RMSE score than the FCNN model with the ECFP featurizer. Compared with the GC and Weave models, our model achieves similar or worse results. In this result, the graph-based model achieves better performance than our model, but our model still achieves reasonable performance compared with the GC model.

Table 3 shows the RMSE scores on the test sets against the Malaria and CEP datasets. For both datasets, our model achieves the lowest RMSE score compared to the other benchmark models. In this result, our model is even better than the GC model, which indicates that our model has competitive performance with the graph convolution featurizer models for regression tasks as well. These results indicate that although the performance of FP2VEC for the regression tasks are not as impressive as that for the

Table 1. The ROC-AUC scores of the test set against Tox21, HIV, BBBP and SIDER datasets

Model	Featurizer		Tox21	SIDER	HIV	BBBP
Our model	FP2VEC	Multitask	**0.876 ± 0.006**	**0.836 ± 0.009**	x	x
		Single task	0.825 ± 0.011	**0.732 ± 0.009**	0.754 ± 0.005	**0.713 ± 0.006**
FCNN	ECFP		0.803 ± 0.012	0.666 ± 0.026	0.698 ± 0.037	0.688 ± 0.005
Bypass	ECFP		0.810 ± 0.013	0.673 ± 0.025	0.693 ± 0.026	0.702 ± 0.006
GC	GraphConv		0.829 ± 0.006	0.638 ± 0.012	**0.763 ± 0.016**	0.690 ± 0.009
Weave	Weave		0.820 ± 0.01	0.581 ± 0.027	0.703 ± 0.039	0.671 ± 0.014

Note: The benchmark results are directly taken from Wu *et al.* (2018). The highest ROC-AUC score for each task is highlighted in bold.

classification tasks, our model can consistently achieve comparative performance compared to other featurizers and models.

3.3 Analysis of FP2VEC and QSAR model

3.3.1 Comparison with circular fingerprint featurizer

To ensure that a new featurizer FP2VEC provides a clear advantage over the circular fingerprint featurizer, we test the circular fingerprint itself as an input feature to our CNN-based QSAR model. If the QSAR model that only uses the circular fingerprint as a featurizer achieves a similar result with the FP2VEC model, we cannot conclude that the increased accuracies of our model are in fact the result of our new featurizer FP2VEC. We evaluate the circular fingerprint model on the Tox21 dataset. We use the same CNN-based QSAR model. Thus, only the featurizer, circular fingerprint or FP2VEC is the difference between the two models. The result shows that the prediction result of the circular fingerprint model is much worse than that for FP2VEC (Supplementary Table S8). The test set ROC-AUC score of the circular fingerprint model is 0.586. The circular fingerprint vectors are sparse; thus, the small window size of the convolution filters cannot appropriately capture the molecular features. With a larger window size of 11, the prediction accuracy improves slightly (ROC-AUC score of 0.624). A window size larger than 11 makes no significant difference. These results clearly indicate that the FP2VEC featurizer improves the prediction performance in QSAR tasks compared to the raw circular fingerprint.

Table 2. The RMSE scores on test sets for the ESOL, FreeSolv and Lipophilicity datasets

Model	Featurizer	ESOL	FreeSolv	Lipophilicity
Our model	FP2VEC	1.06 ± 0.10	1.56 ± 0.22	0.84 ± 0.02
FCNN	ECFP	1.12 ± 0.15	1.87 ± 0.07	0.86 ± 0.01
GC	GraphConv	0.97 ± 0.01	1.40 ± 0.16	**0.66 ± 0.04**
Weave	Weave	**0.61 ± 0.07**	**1.22 ± 0.28**	0.72 ± 0.04

Note: The benchmark results are directly taken from Wu *et al.* (2018). The lowest RMSE score for each task is highlighted in bold.

Table 3. The RMSE scores on test sets for the Malaria and CEP datasets

Featurizer	Network	Malaria	CEP
FP2VEC	CNN	**1.01 ± 0.02**	**1.34 ± 0.04**
ECFP	Linear	1.13 ± 0.03	2.63 ± 0.09
	Neural network	1.36 ± 0.10	2.00 ± 0.09
GraphConv	Linear	1.15 ± 0.02	2.58 ± 0.18
	Neural network	1.16 ± 0.03	1.43 ± 0.09

Note: The benchmark results are directly taken from Duvenaud *et al.* (2015). The lowest RMSE score for each task is highlighted in bold.

3.3.2 Analysis of CNN architecture for QSAR model

Here, we construct the QSAR model with a CNN architecture. In our QSAR model, the convolution filters with various window sizes are designed to detect a set of neighboring molecular substructure indices in fingerprint vectors that may be important for predicting a given molecular property. To optimize the window sizes of the filters, we test various window sizes from 1 to 7. In regression tasks, models with the window size of 1 result in the best results overall. It seems reasonable because the ordering of the substructure indices of the fingerprint vectors is completely arbitrary and there is no apparent relationship between neighboring indices in fingerprint vectors. However, in classification tasks, the window size of 5 produces the best results. We speculate that certain substructure indices that are important for a particular molecular property may be closely located in FP2VEC, which are captured by finite size convolution filters.

3.3.3 Comparison with MACCS-based FP2VEC

The FP2VEC algorithm uses a molecular fingerprint to convert the molecular substructures to integer numbers. FP2VEC can be implemented with any fingerprint method, not only with the circular fingerprint. To verify this, we replace the circular fingerprint with MACCS in the FP2VEC algorithm. MACCS is a fingerprint with 166 bit-size keys in which each key represents a certain molecular substructure. MACCS is implemented by RDKit. We evaluated the MACCS-based FP2VEC against various QSAR tasks. The QSAR model is the same as we described in Section 2. The results are shown in Table 4.

In classification tasks, the MACCS-based FP2VEC model achieves slightly lower prediction accuracies than the circular fingerprint-based FP2VEC model. The multitask learning models are also found to be better than the single task models. Regarding regression tasks, some tasks achieve better results than the circular fingerprint-based FP2VEC. Remarkably, for the FreeSolv dataset, the MACCS-based FP2VEC model (1.00) exceeds the prediction accuracy of the best benchmark results (1.22 by the Weave model). In the ESOL and Lipophilicity datasets, the MACCS-based FP2VEC model is slightly better than the circular fingerprint-based FP2VEC model, while the prediction accuracy is reduced in the Malaria and CEP datasets. These results suggest that the FP2VEC algorithm can also adopt not only the circular fingerprint but also any other type of fingerprint such as MACCS.

Table 4. The QSAR prediction results of the MACCS-based FP2VEC model on the Tox21, SIDER, ESOL, FreeSolv, Lipophilicity, Malaria and CEP datasets

Dataset		Task	Metric	Validation set	Test set
Tox21	Multitask	Classification	ROC-AUC	0.863 ± 0.013	0.860 ± 0.006
	Single task	Classification	ROC-AUC	0.820 ± 0.007	0.819 ± 0.011
SIDER	Multitask	Classification	ROC-AUC	0.822 ± 0.010	0.825 ± 0.013
	Single task	Classification	ROC-AUC	0.670 ± 0.011	0.650± 0.006
ESOL	Single task	Regression	RMSE	0.96 ± 0.09	0.99 ± 0.06
FreeSolv	Single task	Regression	RMSE	1.00 ± 0.12	1.08 ± 0.25
Lipophilicity	Single task	Regression	RMSE	0.84 ± 0.03	0.82 ± 0.05
Malaria	Single task	Regression	RMSE	1.04 ± 0.04	1.04 ± 0.02
CEP	Single task	Regression	RMSE	1.63 ± 0.06	1.67 ± 0.05

Note: The hyperparameter setting of MACCS-based FP2VEC is the same as Sections 3.1 and 3.2.

4 Conclusion

In this work, we show the possibility of implementing the techniques of the NLP research area for developing a molecular featurizer. Motivated by the apparent analogy between chemical compounds and natural languages, we develop a new molecular featurizer, FP2VEC, which represents a chemical compound as a set of task-specific, information-rich vectors for the given tasks. After that, we develop a QSAR model using a simple CNN architecture that has been successfully used for the sentence classification task in NLP. By testing our new method on several benchmark datasets, we demonstrate that the combination of FP2VEC and CNN can achieve competitive results in many QSAR tasks, especially classification tasks. In addition, the FP2VEC model has a number of additional attractive features. First, we show that the FP2VEC model is especially effective for multitask learning tasks. Second, unlike the Graph Convolution and Weave featurizers, FP2VEC consistently achieves competitive prediction accuracy even when trained on small datasets on the classification tasks. Third, the implementation of our model is trivially simple and easy, and the training of the model is straightforward and fast: it seldom takes more than a few minutes on a typical GPU processor to train the model. We expect that our new molecular featurizer, FP2VEC, as well as a QSAR prediction model based on a simple CNN architecture can provide a valuable tool for developing various computational methods in drug development and toxicology research.

Funding

This work was supported by the National Research Foundation of Korea (NRF) [grants 2017M3A9C4065952 and 2019R1A2C1007951] funded by the Korea government (MSIT).

Conflict of Interest: none declared.

References

Alipanahi,B. *et al.* (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.*, **33**, 831–838.

Asgari,E. and Mofrad,M.R. (2015) Continuous distributed representation of biological sequences for deep proteomics and genomics. *PLoS One*, **10**, e0141287.

Bemis,G.W. and Murcko,M.A. (1996) The properties of known drugs. 1. Molecular frameworks. *J. Med. Chem.*, **39**, 2887–2893.

Cadeddu,A. *et al.* (2014) Organic chemistry as a language and the implications of chemical linguistics for structural and retrosynthetic analyses. *Angew. Chem. Int. Ed. Engl.*, **53**, 8108–8112.

Capuzzi,S.J. *et al.* (2016) QSAR modeling of Tox21 challenge stress response and nuclear receptor signaling toxicity assays. *Front. Environ. Sci.*, **4**, 3.

Collobert,R. *et al.* (2011) Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, **12**, 2493–2537.

Delaney,J.S. (2004) ESOL: estimating aqueous solubility directly from molecular structure. *J. Chem. Inf. Comput. Sci.*, **44**, 1000–1005.

Duvenaud,D.K. *et al.* (2015) Convolutional networks on graphs for learning molecular fingerprints. In: *Advances in Neural Information Processing Systems*, Montreal, Canada, 2015, pp. 2224–2232.

Feinberg,E.N. *et al.* (2018) PotentialNet for molecular property prediction. *ACS Cent. Sci.*, **4**, 1520–1530.

Gamo,F.J. *et al.* (2010) Thousands of chemical starting points for antimalarial lead identification. *Nature*, **465**, 305–310.

Glorot,X. *et al.* (2011) Deep sparse rectifier neural networks. In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, FL, pp. 315–323.

Goh,G.B. *et al.* (2017) SMILES2Vec: an interpretable general-purpose deep neural network for predicting chemical properties. arXiv: 1712.02034 [stat.ML].

Goodfellow,I. *et al.* (2016) *Deep Learning*. MIT Press, Cambridge.

Hachmann,J. *et al.* (2011) The Harvard Clean Energy Project: large-scale computational screening and design of organic photovoltaics on the World Community Grid. *J. Phys. Chem. Lett.*, **2**, 2241–2251.

He,K. *et al.* (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas Valley, NV, 2016, pp. 770–778.

Jaeger,S. *et al.* (2018) Mol2vec: unsupervised machine learning approach with chemical intuition. *J. Chem. Inf. Model.*, **58**, 27–35.

Kadurin,A. *et al.* (2017) druGAN: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Mol. Pharm.*, **14**, 3098–3104.

Kalchbrenner,N. *et al.* (2014) A convolutional neural network for modelling sentences. arXiv, 1404, 2188. [cs.CL].

Kearnes,S. *et al.* (2016) Molecular graph convolutions: moving beyond fingerprints. *J. Comput. Aided Mol. Des.*, **30**, 595–608.

Kim,Y. (2014) Convolutional neural networks for sentence classification. arXiv: 1408.5882 [cs.CL].

Kingma,D.P. and Ba,J. (2014) Adam: a method for stochastic optimization. arXiv: 1412.6980 [cs.LG].

Lavecchia,A. (2015) Machine-learning approaches in drug discovery: methods and applications. *Drug Discov. Today*, **20**, 318–331.

Lecun,Y. *et al.* (1998) Gradient-based learning applied to document recognition. *Proc. IEEE*, **86**, 2278–2324.

Lipinski,C.A. *et al.* (2001) Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug Deliv. Rev.*, **46**, 3–26.

Luechtefeld,T. *et al.* (2018) Machine learning of toxicological big data enables read-across structure activity relationships (RASAR) outperforming animal test reproducibility. *Toxicol. Sci*, **165**, 198–212.

Ma,J. *et al.* (2015) Deep neural nets as a method for quantitative structure-activity relationships. *J. Chem. Inf. Model.*, **55**, 263–274.

Mayr,A. *et al.* (2016) DeepTox: toxicity prediction using deep learning. *Front. Environ. Sci.*, **3**, 80.

Mikolov,T. *et al.* (2013a) Distributed representations of words and phrases and their compositionality. In: Burges,C.J.C. *et al.* (eds) *Advances in Neural Information Processing Systems*. Lake Tahoe, Nevada, Vol. 26, pp. 3111–3119.

Mikolov,T. *et al.* (2013b) Efficient estimation of word representations in vector space. arXiv preprint , arXiv:1301.3781.

Pereira,J.C. *et al.* (2016) Boosting docking-based virtual screening with deep learning. *J. Chem. Inf. Model.*, **56**, 2495–2506.

Ramsundar,B. *et al.* (2015) Massively multitask networks for drug discovery. arXiv: 1502.02072 [stat.ML].

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

Sanchez-Lengeling,B. and Aspuru-Guzik,A. (2018) Inverse molecular design using machine learning: generative models for matter engineering. *Science*, **361**, 360–365.

Segler,M.H.S. *et al.* (2018) Generating focussed molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.*, **4**, 120–131.

Shen,Y. *et al.* (2014) Learning semantic representations using convolutional neural networks for web search. In: *Proceedings of the 23rd International Conference on World Wide Web*, Seoul, Republic of Korea, pp. 373–374. ACM.

Shoichet,B.K. (2004) Virtual screening of chemical libraries. *Nature*, **432**, 862–865.

Srivastava,N. *et al.* (2014) Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, **15**, 1929–1958.

Tropsha,A. (2010) Best practices for QSAR model development, validation, and exploitation. *Mol. Inform.*, **29**, 476–488.

Wójcikowski,M. *et al.* (2018) Development of a protein-ligand extended connectivity (PLEC) fingerprint and its application for binding affinity predictions. *Bioinformatics*, **35**, 1334–1341.

Wu,Z. *et al.* (2018) MoleculeNet: a benchmark for molecular machine learning. *Chem. Sci.*, **9**, 513–530.

Xu,Y. *et al.* (2017) Demystifying multitask deep neural networks for quantitative structure-activity relationships. *J. Chem. Inf. Model.*, **57**, 2490–2504.

Yih,W-T. *et al.* (2014) Semantic parsing for single-relation question answering. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, MD, 2014, Vol. 2: Short papers, pp. 643–648.