Mathematical Biosciences and Engineering

MBE, 20(1): 269–282.
DOI: 10.3934/mbe.2023012
Received: 01 August 2022
Revised: 06 September 2022
Accepted: 07 September 2022
Published: 30 September 2022

http://www.aimspress.com/journal/MBE

Research article

Drug-target binding affinity prediction method based on a deep graph neural network

Dong Ma[1,†], Shuang Li[2,†] and Zhihua Chen[1,*]

[1] Institute of Computing Science and Technology, Guangzhou University, Guangzhou, China
[2] Beidahuang Industry Group General Hospital, Harbin, China

† The authors contributed equally to this work.

* **Correspondence:** Email: czhgd@gzhu.edu.cn.

Abstract: The development of new drugs is a long and costly process, Computer-aided drug design reduces development costs while computationally shortening the new drug development cycle, in which DTA (Drug-Target binding Affinity) prediction is a key step to screen out potential drugs. With the development of deep learning, various types of deep learning models have achieved notable performance in a wide range of fields. Most current related studies focus on extracting the sequence features of molecules while ignoring the valuable structural information; they employ sequence data that represent only the elemental composition of molecules without considering the molecular structure maps that contain structural information. In this paper, we use graph neural networks to predict DTA based on corresponding graph data of drugs and proteins, and we achieve competitive performance on two benchmark datasets, Davis and KIBA. In particular, an MSE of 0.227 and CI of 0.895 were obtained on Davis, and an MSE of 0.127 and CI of 0.903 were obtained on KIBA.

Keywords: artificial intelligence; biological sequence analysis; deep learning; binding affinity prediction; graph convolution network

1. Introduction

With the rapid development of machine learning in recent years, artificial intelligence has also been applied to various fields, most of the time achieving valuable results [1–3]. Computer-aided drug

design is one of these areas and is of great interest. The high cost and time-consuming nature of drug development makes the research of new drugs extremely difficult [4]. Drug-target affinity (DTA) prediction is one of the important subtasks that helps to reduce the time-consuming pre-drug selection phase of drug development [5,6].

There are three main types of computational approaches for DTA prediction, including molecular docking [7], traditional machine learning [8–10], and deep learning [11–13]. Molecular docking is based on protein structure to explore the main binding modes of ligands when binding to proteins [14], but it requires the crystallized structure of proteins that are difficult to obtain, which also indirectly affects its final performance [15]. Traditional machine learning, on the other hand, uses one-dimensional sequence representations in drug and protein sequences to train neural networks; however, these models represent drugs as strings. Such representations can reflect the atomic composition of molecules and the chemical bonds between atoms, but they cannot retain the structural information of molecules [16]. The structural information of the molecule, in turn, affects its chemical properties, which may impair the predictive power of the model as well as the functional relevance of the learned potential space. Deep learning models that have been widely used in recent years also perform well in the DTA prediction task, and learning using deep molecular modeling functions is gradually becoming more common because it can capture hidden information that is difficult to model by human experience.

One of the models that may be most suitable for the DTA prediction task is the graph neural network (GNN). The GNN can directly process graph data that can preserve structural information, and this approach has already made research progress. GraphDTA [17] introduces GNN into the DTA prediction task by constructing a drug molecule graph and performs feature extraction for drug molecules based on the graph data, while protein molecules as organic macromolecules can still only use CNN to extract features. DGraphDTA [18] builds graph data for protein molecules based on protein structure prediction, which allows both molecules to be represented by a graph and enables the application of GNN to extract features. However, it only compares two models, GCN [19] and GAT [20], and cannot fully evaluate the performance of GNN in the DTA prediction task.

2. Materials and methods

2.1. Datasets

The data for training generally contains the dataset of drug and protein molecules and the values of corresponding drug-target affinity. This research applies Davis [21] and KIBA (Kinase Inhibitor BioActivity) datasets for specific experiments, and the two datasets are typically used as benchmarks. The Davis dataset completely covered the binding affinity between all 68 drugs and 442 targets included, which was measured by K_d values (kinase dissociation constants). The KIBA dataset integrates information from IC_{50}, K_i (inhibition constant) and K_d (dissociation constant) measurements into a single bioactivity score containing a bioactivity matrix of 2111 kinase inhibitors and 229 human kinases. The drug and protein molecule entities in the two datasets are shown in Table 1. The PDBBind [22,23] dataset is a comprehensive database of drug-target 3D structure binding affinities in the Protein Data Bank (PDB). The PDBBind dataset provides 3D coordinates of target proteins and ligand structures. We use the general set as the training set and the refined set of the PDBBind dataset as the test set. In order to have a stable training process, we only select samples from the general set

with a protein sequence length less than 1000 amino acids. The general set is divided into a training set and a validation set with a ratio of 4:1. Finally, we have 8391 training samples, 1680 validation samples and 3940 test samples. To reduce random errors, we trained and tested the model for performance evaluation using a 5-fold training set and test set of the benchmark dataset, while introducing various methods and metrics for comparison.

Table 1. The Davis and KIBA datasets.

Dataset	Drugs	Proteins	Binding entities
Davis	68	442	30,056
KIBA	2111	229	118,254

The drug molecules are represented in the sequence format called SMILES (Simplified Molecular Input Line Entry System) [24], and this form is a chemical notation language that uses the element symbols of atoms and chemical bonds between them to represent molecules. However, because the sequence data only retain the structural information of molecules, the performance may be worse if the original SMILES string is directly applied. In addition, graph data can store more 3D features by comparison. Therefore, in this method, the molecular graph is constructed by sequence to meet the requirement for the model. Specifically, the molecular graph is constructed according to the drug SMILES string, with atoms as nodes and bonds as edges. In order to ensure that the features of the nodes are fully considered in the graph convolution process, self-loops are also added to the graph construction to improve the feature performance of the drug molecules. Protein molecules are also processed into graph data similarly. The raw data in the datasets are also sequence strings. However, the above idea that processing drugs does not work while proteins are biological macromolecules, the protein graphs will be too large to satisfy model conditions if atoms are constructed as nodes and chemical bonds are designed as edges. Owing to the development of protein structure prediction, it is feasible to utilize predicted structural information to approximate the real 3D structure of proteins. In this paper, the Pconsc4 [25] method was used to output contact map [26] is used to predict protein structure. The method assumes the residue of protein as nodes, and edges are determined by the Euclidean distance between the C_β atoms (C_α atoms for glycine) of residue pairs below a specified threshold [26]. Because the graph is constructed with residuals as nodes, features are selected around the residuals, which exhibit different properties due to the R group. These features include polarity, chargedness, aromaticity, etc. And in this paper, we use PSSM [27] to represent the characteristics of the residue nodes. The selected node features of drug and protein molecules and detailed data preprocessing measures are invariable as those in DGraphDTA [18].

2.2. Method

Almost every drug development study focuses on how to deal with drugs and target molecules, and they preprocess data by applying other algorithms into special forms to fit their approach [28]. However, the models in these studies are relatively simple. DeepDTA only equips three CNN layers to extract molecular features, and DGraphDTA employs two GNN layers and 2 fully connected layers to extract node representations. With the aim of representing a large amount of knowledge and obtaining high accuracy, a model with high expressiveness requires a larger training set. To this extent, a model with more parameters and higher complexity is advantageous. On the other hand, however, an overly

complex model may be difficult to train and may lead to unnecessary resource consumption [29]. So there is a need to balance the model complexity so that the model has a high expressive power while reducing unnecessary consumption.

CNNs and RNNs generally have advanced performance in handling Euclidean data, but the structural information of molecular graphs cannot be expressed in Euclidean space. This leads to CNN and RNN methods rarely achieving optimal results. Therefore, most researchers apply graph neural networks to extract the features of graph data.

Let $G = (V, E)$ denote a simple and connected undirected graph with n nodes and m edges. Let $\mathbf{A}$ be the adjacency matrix, $\mathbf{D}$ be the diagonal degree matrix, and $\mathbf{I}_N$ be an n-order identity matrix. Let $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ denote the adjacency matrix with self-loop added, and the corresponding diagonal degree matrix is denoted by $\widetilde{\boldsymbol{D}}$. Subsequently, let $\mathbf{X} \in R^{n*d}$ denote the node feature matrix, where the ith row of the matrix represents the d-dimensional feature vector of the ith node.

The GNN is a network designed to work directly with graph data and exploit its structural information, and there are now many variants available after years of development. The most well-known one is the GCN, which is widely used in graph data problems. For the GCN, each layer will implement a convolution operation through Eq (1):

$$H^{l+1} = f(H^l, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^l W^{l+1}), \tag{1}$$

where H^l represents the lth layer of node embedding with $H^0 = \mathbf{X}$, σ () is a nonlinear activation function, and W denotes a learnable parameter. Qimai Li et al. [30] prove that graph convolution is essentially Laplacian smoothing [31]; however, repeated application of Laplacian smoothing may mix the features of vertices from different clusters and make them indistinguishable [32]. In the case of symmetric Laplacian smoothing, they will converge to be proportional to the square root of the vertex degree [33]. This is why a deeper GCN will lead to performance decrease.

However, we find that the models that apply GCN commonly set their number of GNN layers to 2 or 3, and consequently these models will have no ability to extract high-order neighborhood information. This is attributed to the notion that stacking more layers tends to degrade the performance of these models, and such a phenomenon is called "oversmoothing" [30]. In other words, this situation refers to the fact that during the training process of graph neural networks, as the number of network layers and iterations increase, the hidden layer representations of each node within the same connected component will tend to converge to the same value. This will result in the final node representation having no actual meaning in the case of deeper layers. Even the addition of residual connectivity, an effective technique widely used in deep CNNs, merely mitigates the oversmoothing problem in GCNs [19].

The purpose of the GNN is to extract the graph embedding of the entire molecular graph. However, if only two or three layers of GCN are used, then for a certain node, just 2-hop or 3-hop neighbor node information can be perceived, which is unfavorable for the information of the whole graph that needs to be obtained to construct the graph embedding [34]. Thus, the GCN operations of several layers can only be regarded as local information aggregation, and the obtained graph embedding is not sufficiently accurate to represent the graph.

Graph Diffusion Convolution (GDC) [35] replaces the multilayer convolution operation in GCN by using graph diffusion, and the graph diffusion process is given by the generalized diffusion matrix:

$$S = \sum_{k=0}^{\infty} \theta_k T^k, \tag{2}$$

with the weighting coefficients θ_k and transition matrix T. The selection of θ_k and T must ensure that Eq (2) converges. One of the main options of weighting coefficients θ_k follow Eq (3) of Personalized PageRank [36]:

$$\theta_k{}^{PPR} = \alpha^{PPR}(1-\alpha^{PPR})^k, \alpha \in (0,1), \tag{3}$$

where α^{PPR} is the teleport probability, and the greater α^{PPR} is, the more information is diffused. T is the transfer matrix for a random walk in an undirected graph, and $T_{rw} = \mathbf{AD}^{-1}$. Considering the information about the node itself, GDC further adjusts the random walk by adding self-loops to the original adjacency matrix and replaces transition matrix T with $T_{sym} = (\omega_{\text{loop}}\ \mathbf{I}_N + \mathbf{D})^{-1/2}\ (\omega_{\text{loop}}\ \mathbf{I}_N + \mathbf{A})\ (\omega_{\text{loop}}\ \mathbf{I}_N + \mathbf{D})^{-1/2}$ where $\omega_{\text{loop}} \in \mathbf{R}^+$. Last, GDC takes over the normal adjacency matrix $\mathbf{A}$ with the sparse version $\tilde{S}$ of the generalized graph diffusion matrix S. The sparsification is done by selecting the k highest mass entries per column in S or by setting the entries below the threshold ε to zero and then performing the subsequent normal convolution operation.

Simple Graph Convolution (SGC) [37] believes that the complexity of GCNs inherited from neural networks is burdensome and unnecessary for less demanding applications. Thus, SGC reduces the additional complexity of GCNs by removing the nonlinearity between the layers of GCNs and simplifying the resultant function to a single linear transformation. The experiments show that the final obtained model is comparable to GCNs with higher computational efficiency and fewer fitted parameters.

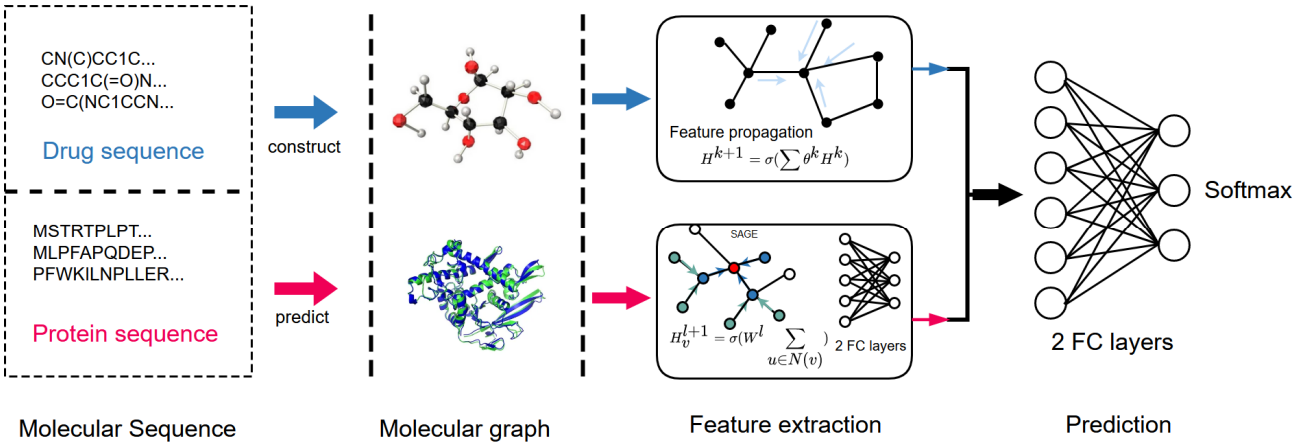

Figure 1. The pipeline of this paper.

Simple spectral graph convolution (S^{2}GC) [38] further analyzes the Markov Diffusion Kernel [39] and improves the GCN to obtain the following iterative function:

$$\hat{Y} = \text{softmax}(\frac{1}{K}\sum_{k=0}^{K}((1-\alpha)\tilde{T}^k X + \alpha X)W), \tag{4}$$

where $\tilde{T}$ is the special case of T_{sym} at $\omega_{\text{loop}} = 1$, i.e., $\tilde{T} = (I^N + \text{D})^{-1/2}(I^N + \text{A})(I^N + \text{D})^{-1/2}$, the $\alpha \in [0,1]$ parameter balances the self-information of the node with the consecutive neighborhoods, and the value of K represents the receptive field size of the model. For example, each node can update its own node feature representation based on the information of its farthest 4-hop neighbors if $K = 4$. The above iterative formula is designed by S^{2}GC for the node classification task. In this task, GNN aims to extract node features and subsequently add fully connected layers to fit channels, and the last

layer of the model uses softmax as a classifier. Therefore, when S^2GC is applied to this paper, the softmax layer needs to be removed, and the iterative function only retains the part within the softmax function. Figure 1 illustrates the flow of the process used in this paper, which consists of four main parts: the raw molecular sequence data, the processed drug molecule graph and the predicted protein contact map, feature extraction using graph models, and finally DTA prediction.

3. Results and discussion

3.1. Experiment setting

In this paper, we alter the GCN layer of the model in DGraphDTA with other graph neural networks and evaluate the impact of GNNs on the final results. However, limited by computational resources, we first compare the parametric size of various networks and identify some algorithms with lower complexity for subsequent experiments. The algorithms are built based on the PyTorch [40] library, and Pytorch geometric (PyG) [41] provides a variety of portable access algorithms from which to choose graph data tasks. The first step is to compare the graph models used parametrically due to the large amount of data and to keep the models from becoming too complex. Assume the input dimension and output dimension are m and n, respectively, $k = m*n$, and the results of the partial comparison in the case of a single-layer network are shown in Table 2.

Table 2. Number of parameters of the single-layer model.

Model	The numbers of parameters
GCN [19]	$k + n$
GAT [20]	$k + 3n$
GraphConv [42]	$2k + n$
SAGE [43]	$2k + n$
SGC [37]	$k + n$

From Table 2, we can obtain these models with a small difference in complexity. There are other models with complexity far beyond these, such as Molecular Fingerprints (MFConv) [44] with the number of parameters up to 20 times k. It is obvious that using this kind of algorithm will reduce the training efficiency of the model. Similar to S^2GC, SGC is not applied to the model to compare the performance of various algorithms. As mentioned before, S^2GC takes the adjacency matrix of the graph to construct the transfer matrix when calculating $\tilde{T}$, and processing the transfer matrix will consume considerable time when applied to the DTA prediction task. Therefore, we extracted the features of drug molecules using S^2GC, while the SAGE algorithm was used for the feature extraction of protein molecules in this experiment.

Training the model requires setting several parameters and tuning details within the model. One of these details is to adjust the proportion of drug features and protein features in the final execution of classification to adjust the influence of both molecules on the final prediction results. Since drug molecules and protein molecules are encoded as 54- and 78-dimensional features, respectively, and their feature dimension ratio is approximately 2:3, the feature dimension ratio of the two molecules produced by the graph neural network can be set to 2:3. In addition, we set the value of K, which is the number of iterations of the network, to 4 and the value of α to 0.05 in the S^2GC model.

Since S^2GC requires processing adjacency and transfer matrices for graphs, DTA prediction is a task that includes multiple graph data. Therefore, model training is time-consuming and cannot be performed using a large number of different parameters, and the setting of various parameters needs to be based on experience. A few important hyperparameters used in the model are as follows: learning rate of 0.001, number of iterations of 2000, feature dimensions of 54 and 78 for drug and protein molecules, respectively, and corresponding output dimensions of 112 (16*7) and 144 (16*9) for the two types of molecules in the feature extraction phase, where 112:144 can be approximated as 54:78.

3.2. Metrics

The same metrics as in the benchmark are implemented to calculate the concordance index (*CI*) [45] and the mean squared error (MSE) [46]. The CI is essentially an estimate of the probability that the predicted value will be consistent with the actual value; it measures whether the predicted affinity values for two random drug-target pairs are predicted in the same order as the true values and is calculated by Eq (5).

$$CI = \frac{1}{Z} \sum_{\alpha_x > \alpha_y} h(b_x - b_y) \tag{5}$$

where b_x is the predicted value of the larger affinity α_x, by is the predicted value of the smaller affinity α_y, Z is the normalization constant, and $h(x)$ is the step function:

$$h(x) = \begin{cases} 1, x > 0 \\ 0.5, x = 0 \\ 0, x < 0 \end{cases} \tag{6}$$

MSE measures the difference between the prediction and the label, and the formula is as follows:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (p_i - y_i)^2, \tag{7}$$

where p_i is the predicted value, y_i is the corresponding actual label, and a smaller MSE means that the predicted value of the model is closer to the true value.

In addition, another metric, Pearson correlation coefficient, is also used in some articles for performance comparison, calculated by Eq (8). where *cov* is the covariance between the predicted value *p* and the real number *y*, and σ () is the standard deviation.

$$Pearson = \frac{cov(p,y)}{\sigma(p)\sigma(y)} \tag{8}$$

3.3. Experimental results

In this study, we introduced S2GC and SAGE to extract drug and protein features, respectively, and we proceeded to use several layers of FC to make predictions. Figures 2 and 3 show the distribution between the labels and predicted values of the samples in the two datasets. There is a straight line with a slope of 1. When the sample points are closer to this line, the corresponding DTA predicted by the model is closer to the true value, which indicates better model performance. The distribution of sample

points on these two plots also shows that our method is able to predict the DTA values more accurately, with the vast majority of the predicted values and actual labels having a difference of 1 or less.

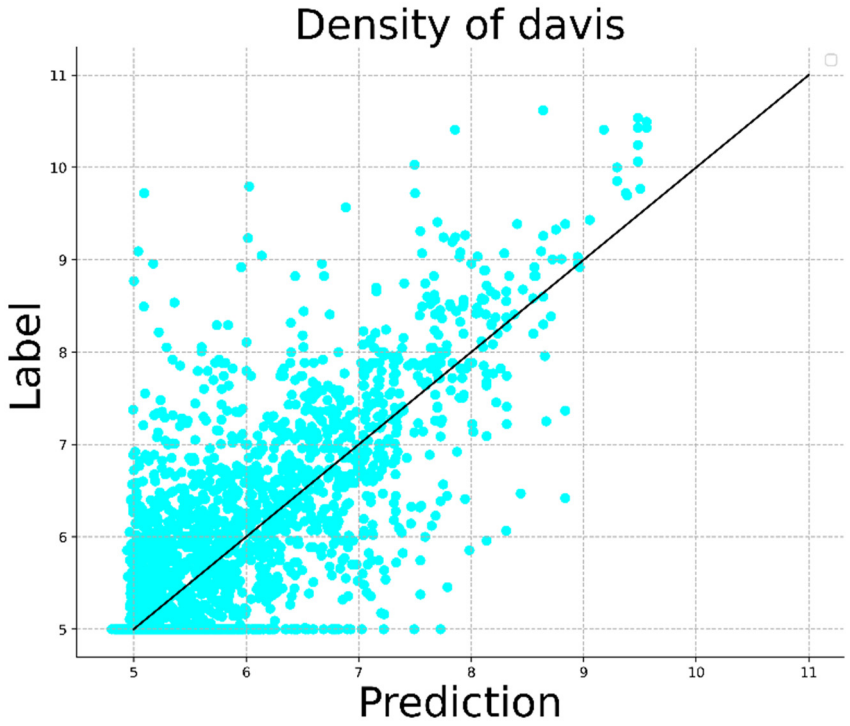

Figure 2. Labels and predicted values in Davis.

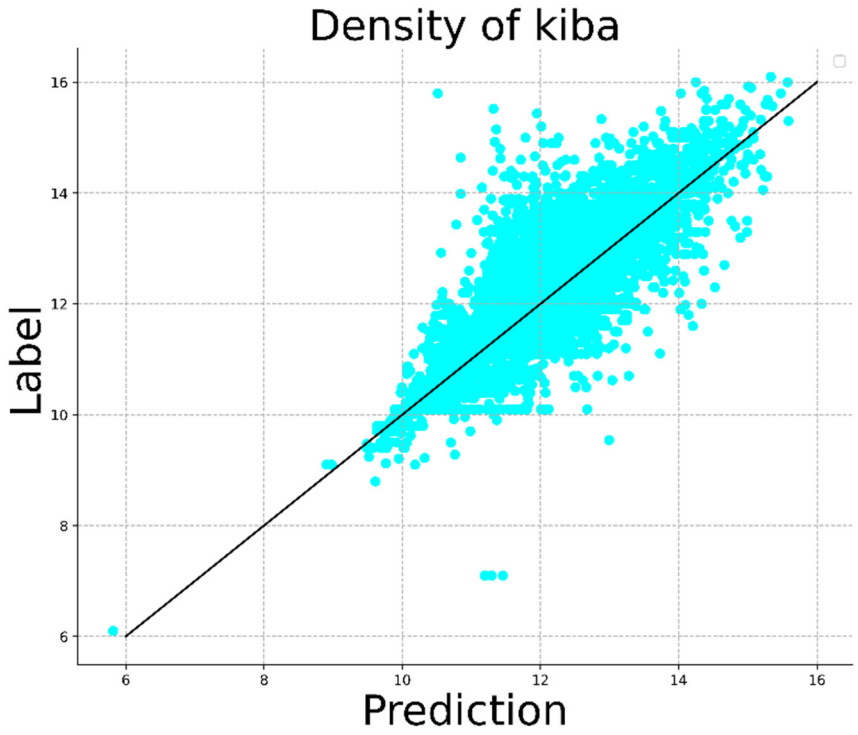

Figure 3. Labels and predicted values in KIBA.

We conducted several experiments to predict DTA and report the performance of certain models on two metrics, MSE and CI. Tables 3 and 4 show the MSE and CI values on the independent test sets of the two benchmark datasets, respectively. It can be observed in the Davis dataset that our model has the best performance on both MSE and CI metrics. GraphDTA, another model based on graph neural networks, also performs well, obtaining an MSE of 0.229, which is second only to the method in this paper. The rest of the models extract features mainly based on sequential data. These models are weaker than the graph model in terms of performance because they do not handle structural information well.

For the KIBA dataset, our model performs slightly worse than Affinity2Vec, obtaining only the second-best result. However, our model also obtained an MSE of 0.127 and a CI of 0.903, which are not far from Affinity2Vec's MSE of 0.124 and CI of 0.91. We argue that the main reason for this is the type of dataset; although this dataset is large, it does not contain all binding values, and known DTA only accounts for 24.4% of the affinity. Training on this dataset uses semi-supervised learning and is more suitable for Affinity2Vec's seq2seq [47] and ProtVet [48], which are two unsupervised data-driven models. On the other hand, this may be due to the more complex method in Affinity2Vec; in terms of the model, it uses multilayer GRU [49] and ProtVec to extract sequence features and XGBoost [50] for the final DTA prediction. In terms of features, it applies not only embedding features, but also drug-target meta-path score features [51] and both hybrid features. This fits the characteristics of the large data volume in the KIBA dataset, and there is enough data in KIBA to ensure a smooth fit of the Affinity2Vec training process. In other words, the model and complexity of Affinity2Vec are more suitable for training in a large dataset such as KIBA, which can also explain why the performance of Affinity2Vec on the Davis dataset is not optimal, i.e., due to the insufficient amount of data in Davis. As the PDBBind dataset, the method in this paper achieves 1.231, 0.756 and 0.683 on MSE, CI and Pearson indicators, respectively, which also has some performance improvement relative to some other methods. The detailed comparison is shown in Table 5.

As the benchmark model for this paper, DGraphDTA obtained better results on the Davis and KIBA datasets, achieving 0.904, 0.202, and 0.867 for CI, MSE, and Pearson metrics on Davis, and 0.904, 0.126, 0.903 for CI, MSE, and Pearson metrics on KIBA, respectively. However, DGraphDTA only uses three layers of GCNs for feature extraction, and it cannot continue to increase the number of layers subsequently, and simply increasing GCNs can also lead to oversmoothing problems and may also degrade the model performance. The S2GC algorithm used in this paper can be set up to $k = 16$, which is the effect of 16 layers of GCN. However, due to the lack of computational resources, this paper can only use $k = 4$ for experiments, and the difference is not obvious compared to the three-layer GCN of DGraphDTA, which is why the performance of this paper's method decreases instead of rising compared to DGraphDTA.

In general, GNN-based models commonly outperform other models in terms of performance with little difference in model complexity, and models that rely only on sequence information indeed struggle to outperform graph models in terms of feature extraction. Although other methods can stack models and features to obtain better results, graph models are certainly a more suitable class of architectures for the DTA prediction task. In addition, the performance of the same model may vary on different datasets. For the variation of molecular size and whether all DTA values are covered in both Davis and KIBA datasets, different model architectures can be subsequently set according to different datasets to meet the matching relationship between data volume and model complexity.

Table 3. Performance of multiple models on the Davis test set.

	MSE	CI	Pearson
GraphDTA [17]	0.229	0.886	-
DeepFusionDTA [52]	0.253	0.887	0.820
WideDTA [53]	0.262	0.886	0.820
Affinity2Vec [54]	0.242	0.886	-
S^2GC + SAGE	0.227	0.895	0.846

Table 4. Performance of multiple models on the KIBA test set.

	MSE	CI	Pearson
GraphDTA	0.139	0.891	-
DeepFusionDTA	0.176	0.876	0.856
WideDTA	0.179	0.875	0.856
Affinity2Vec	0.124	0.91	-
S^2GC + SAGE	0.127	0.903	0.887

Table 5. Performance of multiple models on the PDBBind test set.

	MSE	CI	Pearson
GCNConvNet	1.308	0.711	0.597
GINConvNet	1.323	0.690	0.551
S^2GC + SAGE	1.231	0.756	0.683

4. Conclusions

Predicting the strength of drug-target binding affinity is more informative and challenging than just classifying drug-target interactions. The prediction of DTA requires molecular graph data based on drugs and proteins. For such graph data, it is more appropriate to use graph neural networks instead of other neural network models to extract features. To solve the oversmoothing problem of the most widely used GCN when the model layers are deepened, we compared several networks used to alleviate the oversmoothing, from which we selected an S2GC model that performs well in the node classification task to extract molecular features. We also found that the graph neural network-based model outperformed the other algorithms with a small difference in model complexity, so we can continue to follow this idea in our subsequent work to explore the performance of graph neural networks in the DTA prediction task.

Acknowledgements

This work was supported by the Chinese National Natural Science Foundation under Grant 61876047 and the Municipal Government of Quzhou (Grant Number 2020D003 and 2021D004). Funding by Science and Technology Projects in Guangzhou (202201020120). We are grateful to Dr. Ding and all partners for their help and support during the research process.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. Y. Zhang, Artificial intelligence for bioinformatics and biomedicine, *Curr. Bioinf.*, **15** (2020), 801–802. https://doi.org/10.2174/1574893615082012210923 30
2. B. Jena, S. Saxena, G. K. Nayak, L. Saba, N. Sharma, J. S. Suri, Artificial intelligence-based hybrid deep learning models for image classification: The first narrative review, *Comput. Biol. Med.*, **137** (2021), 104803. https://doi.org/10.1016/j.compbiomed.2021.104803
3. H. Lin, Development and application of artificial intelligence methods in biological and medical data, *Curr. Bioinf.*, **15** (2020), 515–516. https://doi.org/10.2174/1574893615062006101123 45
4. R. C. Andrade, M. Boroni, M. K. Amazonas, F. R. Vargas, New drug candidates for osteosarcoma: Drug repurposing based on gene expression signature, *Comput. Biol. Med.*, **134** (2021), 104470. https://doi.org/10.1016/j.compbiomed.2021.104470
5. J. Wang, Y. Shi, X. Wang, H. Chang, A drug target interaction prediction based on LINE-RF learning, *Curr. Bioinf.*, **15** (2020), 750–757. https://doi.org/10.2174/1574893615666191227092453
6. M. Aslam, M. Shehroz, F. Ali, A. Zia, S. Pervaiz, M. Shah, et al., Chlamydia trachomatis core genome data mining for promising novel drug targets and chimeric vaccine candidates identification, *Comput. Biol. Med.*, **136** (2021), 104701. https://doi.org/10.1016/j.compbiomed.2021.104701
7. J. Yan, J. Huang, C. Zhang, H. Huo, F. Chen, Virtual screening of acetylcholinesterase inhibitors based on machine learning combined with molecule docking methods, *Curr. Bioinf.*, **16** (2021), 963–971. https://doi.org/10.2174/1574893615999200719234045
8. F. F. Ahmed, M. Khatun, M. Mosharaf, M. N. Mollah, Prediction of protein-protein interactions in Arabidopsis thaliana using partial training samples in a machine learning framework, *Curr. Bioinf.*, **16** (2021), 865–879. https://doi.org/10.2174/1574893616666210204145254
9. D. P. Boso, D. D. Mascolo, R. Santagiuliana, P. Decuzzi, B. A. Schrefler, Drug delivery: Experiments, mathematical modelling and machine learning, *Comput. Biol. Med.*, **123** (2020), 103820. https://doi.org/10.1016/j.compbiomed.2020.103820
10. Y. Ding, J. Tang, F. Guo, Q. Zou, Identification of drug-target interactions via multiple kernel-based triple collaborative matrix factorization, *Briefings Bioinf.*, **23** (2022). https://doi.org/10.1093/bib/bbab582
11. R. Su, X. Liu, L. Wei, Q. Zou, Deep-Resp-Forest: A deep forest model to predict anti-cancer drug response, *Methods*, **166** (2019), 91–102. https://doi.org/10.1016/j.ymeth.2019.02.009
12. Q. Bai, S. Liu, Y. Tian, T. Xu, A. J. Banegas-Luna, H. Pérez-Sánchez, Application advances of deep learning methods for de novo drug design and molecular dynamics simulation, *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, **12** (2022), e1581. https://doi.org/10.1002/wcms.1581
13. Q. Bai, S. Tan, T. Xu, H. Liu, J. Huang, X. Yao, MolAICal: A soft tool for 3D drug design of protein targets by artificial intelligence and classical algorithm, *Briefings Bioinf.*, **22** (2021). https://doi.org/10.1093/bib/bbaa161

14. J. Li, A. Fu, L. Zhang, An overview of scoring functions used for protein-ligand interactions in molecular docking, *Interdiscip. Sci.: Comput. Life Sci.*, **11** (2019), 320–328. https://doi.org/10.1007/s12539-019-00327-w
15. Y. Ding, J. Tang, F. Guo, Protein crystallization identification via fuzzy model on linear neighborhood representation, *IEEE/ACM Trans. Comput. Biol. Bioinf.*, **18** (2019), 1986–1995. https://doi.org/10.1109/TCBB.2019.2954826
16. Y. Ding, J. Tang, F. Guo, Human protein subcellular localization identification via fuzzy model on kernelized neighborhood representation, *Appl. Soft Comput.*, **96** (2020), 106596. https://doi.org/10.1016/j.asoc.2020.106596
17. T. Nguyen, H. Le, T. P. Quinn, T. Nguyen, T. D. Le, S. Venkatesh, GraphDTA: Predicting drug-target binding affinity with graph neural networks, *Bioinformatics*, **37** (2021), 1140–1147. https://doi.org/10.1093/bioinformatics/btaa921
18. M. Jiang, Z. Li, S. Zhang, S. Wang, X. Wang, Q. Yuan, et al., Drug-target affinity prediction using graph neural network and contact maps, *RSC Adv.*, **10** (2020), 20701–20712. https://doi.org/10.1039/D0RA02297G
19. T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, preprint, arXiv:1609.02907.
20. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, preprint, arXiv:1710.10903.
21. M. I. Davis, J. P. Hunt, S. Herrgard, P. Ciceri, L. M. Wodicka, G. Pallares, et al., Comprehensive analysis of kinase inhibitor selectivity, *Nat. Biotechnol.*, **29** (2011), 1046–1051. https://doi.org/10.1038/nbt.1990
22. R. Wang, X. Fang, Y. Lu, S. Wang, The PDBbind database: Collection of binding affinities for protein-ligand complexes with known three-dimensional structures, *J. Med. Chem.*, **47** (2004), 2977–2980. https://doi.org/10.1021/jm030580l
23. R. Wang, X. Fang, Y. Lu, Y. C. Yang, S. Wang, The PDBbind database: Methodologies and updates, *J. Med. Chem.*, **48** (2005), 4111–4119. https://doi.org/10.1021/jm048957q
24. D. Weininger, SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules, *J. Chem. Inf. Comput. Sci.*, **28** (1988), 31–36. https://doi.org/10.1021/ci00057a005
25. M. Michel, D. Menéndez Hurtado, A. Elofsson, PconsC4: Fast, accurate and hassle-free contact predictions, *Bioinformatics*, **35** (2019), 2677–2679. https://doi.org/10.1093/bioinformatics/bty1036
26. Q. Wu, Z. Peng, I. Anishchenko, Q. Cong, D. Baker, J. Yang, Protein contact prediction using metagenome sequence data and residual neural networks, *Bioinformatics*, **36** (2020), 41–48. https://doi.org/10.1093/bioinformatics/btz477
27. J. C. Jeong, X. Lin, X. W. Chen, On position-specific scoring matrix for protein function prediction, *IEEE/ACM Trans. Comput. Biol. Bioinf.*, **8** (2010), 308–315. https://doi.org/10.1109/TCBB.2010.93
28. Y. Ding, P. Tiwari, Q. Zou, F. Guo, H. M. Pandey, C-loss based higher-order fuzzy inference systems for identifying DNA N4-methylcytosine sites, *IEEE Trans. Fuzzy Syst.*, **2022** (2022). https://doi.org/10.1109/TFUZZ.2022.3159103
29. X. Hu, L. Chu, J. Pei, W. Liu, J. Bian, Model complexity of deep learning: A survey, *Knowl. Inf. Syst.*, **63** (2021), 2585–2619. https://doi.org/10.1007/s10115-021-01605-0

30. Q. Li, Z. Han, X. M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in *Thirty-Second AAAI conference on artificial intelligence*, AAAI, New Orleans, USA, (2018), 3538–3545. https://doi.org/10.1609/aaai.v32i1.11604
31. G. Taubin, A signal processing approach to fair surface design, in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM , (1995), 351–358. https://doi.org/10.1145/218380.218473
32. Y. Ding, W. He, J. Tang, Q. Zou, F. Guo, Laplacian regularized sparse representation based classifier for identifying DNA N4-methylcytosine Sites via L2, 1/2-matrix Norm, *IEEE/ACM Trans. Comput. Biol. Bioinf.*, **2021** (2021). https://doi.org/10.1109/TCBB.2021.3133309
33. Y. Ding, J. Tang, F. Guo, Identification of drug-target interactions via dual laplacian regularized least squares with multiple kernel fusion, *Knowledge-Based Syst.*, **204** (2020), 106254. https://doi.org/10.1016/j.knosys.2020.106254
34. P. Tiwari, S. Dehdashti, A. K. Obeid, P. Marttinen, P. Bruza, Kernel method based on non-linear coherent states in quantum feature space, *J. Phys. A: Math. Theor.*, **55** (2022), 355301. https://doi.org/10.1088/1751-8121/ac818e
35. J. Klicpera, S. Weißenberger, S. Günnemann, Diffusion improves graph learning, preprint, arXiv:1911.05485.
36. L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank citation ranking: Bringing order to the web, *Stanford InfoLab*., **1999** (1999).
37. F.Wu, A. Souza, T. Zhang,C. Fifty, T. Yu, K. Weinberger, Simplifying graph convolutional networks, in *International conference on machine learning*, *PMLR*, **97** (2019), 6861–6871. https://doi.org/10.48550/arXiv.902.07153
38. H. Zhu, P. Koniusz, Simple spectral graph convolution, in *International Conference on Learning Representations*, (2020).
39. F. Fouss, K. Francoisse, L .Yen, A. Pirotte, M. Saerens, An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification, *Neural networks*, **31** (2012), 53–72. https://doi.org/10.1016/j.neunet.2012.03.001
40. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, Pytorch: An imperative style, high-performance deep learning library, in *Advances in neural information processing systems*, **32** (2019).
41. M. Fey, J. E. Lenssen, Fast graph representation learning with PyTorch Geometric, preprint, arXiv:1903.02428.
42. C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, et al., Weisfeiler and leman go neural: Higher-order graph neural networks, in *Proceedings of the AAAI conference on artificial intelligence*, AAAI, Honolulu, USA, **33** (2019), 4602–4609. https://doi.org/10.1609/aaai.v33i01.33014602
43. W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in *Advances in neural information processing systems*, **30** (2017).
44. D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, et al., Convolutional networks on graphs for learning molecular fingerprints, in *Advances in neural information processing systems*, **28** (2015). https://doi.org/10.48550/arXiv.1509.09292
45. M. Gönen, G. Heller, Concordance probability and discriminatory power in proportional hazards regression, *Biometrika*, **92** (2005), 965–970. https://doi.org/10.1093/biomet/92.4.965

46. D. M. Allen, Mean square error of prediction as a criterion for selecting variables, *Technometrics*, **13** (1971), 469–475. https://doi.org/10.1080/00401706.1971.10488811
47. Z. Xu, S. Wang, F. Zhu, J. Huang, Seq2seq fingerprint: An unsupervised deep molecular embedding for drug discovery, in *Proceedings of the 8th ACM international conference on bioinformatics, computational biology, and health informatics*, ACM, Boston, USA, (2017), 285–294. https://doi.org/10.1145/3107411.3107424
48. E. Asgari, M. R. Mofrad Continuous distributed representation of biological sequences for deep proteomics and genomics, *PloS one*, **10** (2015), e0141287. https://doi.org/10.1371/journal.pone.0141287
49. J. Chung, C. Gulcehre, K. Cho,. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, preprint, arXiv:1412.3555.
50. T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, ACM, San Francisco, USA, (2016), 785–794. https://doi.org/10.1145/2939672.2939785
51. G. Fu, Y. Ding, A. Seal, B. Chen, Y. Sun, E. Bolton, Predicting drug target interactions using meta-path-based semantic network analysis, *BMC Bioinf.*, **17** (2016), 1–10. https://doi.org/10.1186/s12859-016-1005-x
52. Y. Pu, J. Li, J. Tang, F. Guo, DeepFusionDTA: Drug-target binding affinity prediction with information fusion and hybrid deep-learning ensemble model, *IEEE/ACM Trans. Comput. Biol. Bioinf.,* **2021** (2021). https://doi.org/10.1109/TCBB.2021.3103966
53. H. Öztürk, E. Ozkirimli, A. Özgür, WideDTA: Prediction of drug-target binding affinity. preprint, arXiv:1902.04166.
54. M. A. Thafar, M. Alshahrani, S. Albaradei, T. Gojobori, M. Essack, X. Gao, Affinity2Vec: Drug-target binding affinity prediction through representation learning, graph mining, and machine learning, *Sci. Rep.*, **12** (2022), 1–18. https://doi.org/10.1038/s41598-022-08787-9

AIMS Press