

RESEARCH

Open Access



Combining sequence and network information to enhance protein–protein interaction prediction

Leilei Liu^{1†}, Xianglei Zhu^{1,2†}, Yi Ma¹, Haiyin Piao³, Yaodong Yang¹, Xiaotian Hao¹, Yue Fu¹, Li Wang^{1*} and Jijie Peng⁴

From Biological Ontologies and Knowledge bases workshop 2019 San Diego, CA, USA. 18-21 November 2019

*Correspondence:
wangli@tju.edu.cn

[†]Leilei Liu and Xianglei Zhu contributed equally to this work

¹ College of Intelligence and Computing, Tianjin University, No.135 Yaguan Road, Tianjin 300350, China Full list of author information is available at the end of the article

Leilei Liu and Xianglei Zhu contributed equally to this work

Abstract

Background: Protein–protein interactions (PPIs) are of great importance in cellular systems of organisms, since they are the basis of cellular structure and function and many essential cellular processes are related to that. Most proteins perform their functions by interacting with other proteins, so predicting PPIs accurately is crucial for understanding cell physiology.

Results: Recently, graph convolutional networks (GCNs) have been proposed to capture the graph structure information and generate representations for nodes in the graph. In our paper, we use GCNs to learn the position information of proteins in the PPIs networks graph, which can reflect the properties of proteins to some extent. Combining amino acid sequence information and position information makes a stronger representation for protein, which improves the accuracy of PPIs prediction.

Conclusion: In previous research methods, most of them only used protein amino acid sequence as input information to make predictions, without considering the structural information of PPIs networks graph. We first time combine amino acid sequence information and position information to make representations for proteins. The experimental results indicate that our method has strong competitiveness compared with several sequence-based methods.

Keywords: Protein–protein interactions, Amino acid sequence, Graph convolutional networks

Background

PPIs play an important role in cellular systems of organisms, most proteins perform their functions by interacting with other proteins, so information about the PPIs can help us better understand the function of proteins [1]. Many basic cellular processes involve PPIs, for example metabolic cycles, DNA transcription and replication, and



signaling cascades [2]. Dysfunction in the PPIs will affect people's health and cause diseases, research shows that many diseases are the result of abnormal PPIs involving endogenous proteins, proteins from pathogens or both [3]. Accurately predicting protein interactions is very important for us to study the properties of cellular systems, improve the understanding of disease and provide a basis for the development of novel therapeutic approaches [4].

In recent years, high-throughput biological techniques and large-scale experimental approaches for PPIs identification have achieved tremendous development, lots of PPIs data from different organisms has been discovered by researchers [2]. And *Yeast* predominantly provides PPIs data so far. But the coverage of PPIs data is still very low and there is lots of noisy data in the PPIs dataset, since experimental methods inevitably produce false-positive results [5]. According to previous researches, 50% of the *Yeast* PPIs map and only 10% of the *Human* PPIs networks have been characterised [4]. Moreover, biological techniques and large-scale experiments are often expensive, time-consuming and labor-intensive [6, 7]. Calculation-based methods can solve the problem to a certain extent, and provide reference and guidance for the biological experiment design, which are helpful for laboratory validation.

These computational methods are mainly composed of two phases, representation phase and prediction phase. In representation phase, the methods generate a vectorized representation for each protein using its attribute information. And in the prediction phase, they use traditional machine learning techniques or deep learning to make predictions based on the representation generated in the previous phase. Many characteristic properties of proteins can be used to generate representation, including protein structure information, protein domains, gene neighborhood, phylogenetic profiles, gene expression and literature mining knowledge [8]. In bioinformatics fields, STRINGDB is the most commonly used database, which collects a lot of PPIs data from different species and provides online querying and API for users to retrieve data. For PPIs annotations, STRINGDB computes a combined score by combining the probabilities from the different evidence channels, including fusion evidence, neighborhood evidence, cooccurrence evidence, experimental evidence, textmining evidence, database evidence and coexpression evidence [9]. However, the design of above representation methods requires strong domain knowledge, and some information is difficult to obtain, which limits the practicality of the method to a certain extent [8]. Recently, protein amino acid sequence data has a rapid growth. Compared with the limited number of protein structures, it is undeniable that the number of protein sequences is much larger. Computational methods that make predictions only based on amino acid sequence are arousing great interest of researchers. The experimental results in previous works show that only using protein sequence information can also achieve high prediction accuracy [2, 5, 7, 8, 10–13].

These sequence-based methods have achieved certain results, but the method of generating vectorized representation for proteins based on protein sequence information is complicated. And some methods need extra statistical information, computation complexity and time complexity are high. Moreover, they did not use the structural information of the PPIs networks graph. PPIs data can be represented in the form of graph, where nodes represent proteins and edges represent protein interactions. So the

position information of protein in the graph, which can also be said to be the relationship between proteins, can reflect the properties of proteins to some extent, which is an important supplement to protein sequence information. Combining amino acid sequence information and position information can help to make a more accurate prediction. In this paper, we first time use GCNs to capture the protein's position information in the PPIs networks graph and combine the amino acid sequence information and position information to make representations for each protein. In prediction phase, we use deep neural network (DNN) modules composed of fully connected neural network layers to extract high-level feature information and make predictions. By designing this architecture, we can generate stronger vectorized representation for proteins and make more accurate predictions.

Our main contributions can be summarized as follows: (1) We use GCNs to capture the position information of the proteins in the PPIs networks graph, which can reflect the properties of proteins to some extent. (2) We propose a novel representation method that combines amino acid sequence information and position information. (3) We test our method on several benchmark datasets and the experimental results demonstrate the validity of our method. To the best of our knowledge, this is the first study that combines amino acid sequence information and position information to make representations for proteins.

Experiments and results

In this section, we did two experiments to verify our model. In the first experiment, we compare our method with two state-of-the-art sequence-based PPIs prediction methods, including DPPI [5] and DeepFE-PPI [11]. DPPI takes a probabilistic sequence profile generated by PASI-BLAST as inputs and in prediction phase it uses 5 convolutional modules, 1 random projection module and 1 prediction module to extract features and make predictions. DeepFE-PPI adopts Word2vec to learn feature representation from a large protein database and in prediction phase it uses 4 fully connected layers to extract high-level feature. Through this experiment, it is confirmed that our method can achieve higher prediction accuracy than state-of-the-art methods.

In the second experiment, we did an ablation experiment by only using amino acid sequence information or position information to make predictions. The results of this experiment illustrate the effectiveness of our representation method, that is, combining amino acid sequence information and position information can generate a stronger representation for the protein.

Dataset description

We test our method on three different benchmark datasets, including *Human* dataset, *Yeast* dataset and *S. cerevisiae* core dataset. *Human* and *Yeast* datasets are described by Profppikernel [14], which only contains the top-scoring physical interactions. To get the fair comparison results, we follow the same strategy as DPPI and remove redundancy of the *Human* and *Yeast* datasets such that no two PPIs are similar at sequence level. Two PPIs are considered similar if at least two sequences, each of one PPIs, have a sequence identity greater than 40%. *S. cerevisiae* core dataset is described by You et al. [2], which has a total of 11188 interactions, including 5594 positive interactions and 5594 negative

interactions. The protein pairs which contain a protein with fewer than 50 residues or greater than 40% sequence identity are removed from the dataset. The amino acid sequences are retrieved from the Uniprot database (<http://www.uniprot.org/>).

Evaluation criteria

There are three common evaluation indicators in the classification problem, including accuracy, precision and recall. These indexes are defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

TP (true positive) is the number of samples that labels and predictions are both positive; TN (true negative) is the number of samples that labels and predictions are both negative; FP (false positive) is the number of samples that labels are negative but predictions are positive; FN (false negative) is the number of samples that labels are positive but predictions are negative.

And when label classes are not balanced, precision–recall curves are mainly used. The precision–recall curve shows the tradeoff between precision and recall. A large area under the curve represents both high recall and precision, the best case scenario for a classifier, showing a model that returns accurate results for the majority of classes it selects. Precision–recall curves give a more informative picture of an algorithm’s performance [15].

So when evaluating the method’s performance on *Human* and *Yeast* dataset, in which the number of negative samples is greater than positive samples, we follow the same strategy as DPPI, plot the precision–recall curves and compare the area under the curves of different methods. For *S. cerevisiae* core dataset, the number of negative samples is identical to that of positive samples, we compare accuracy, precision and recall of different methods.

Parameter settings

When implementing our method, the model hyperparameters are determined by grid search and we get the best results with the following hyperparameters. The maximum length of protein amino acid sequence is set to 850 and when capturing proteins’ position information, we use one-layer GCNs to aggregate information from neighbor nodes. The number of neurons in the fully connected layers are 256, 128, 64, 32, 8 and 2 respectively. The dropout layer randomly drops neurons with a probability of 0.5. All parameters are updated by conducting stochastic gradient descent (SGD) and the learning rate of SGD is set to 0.01.

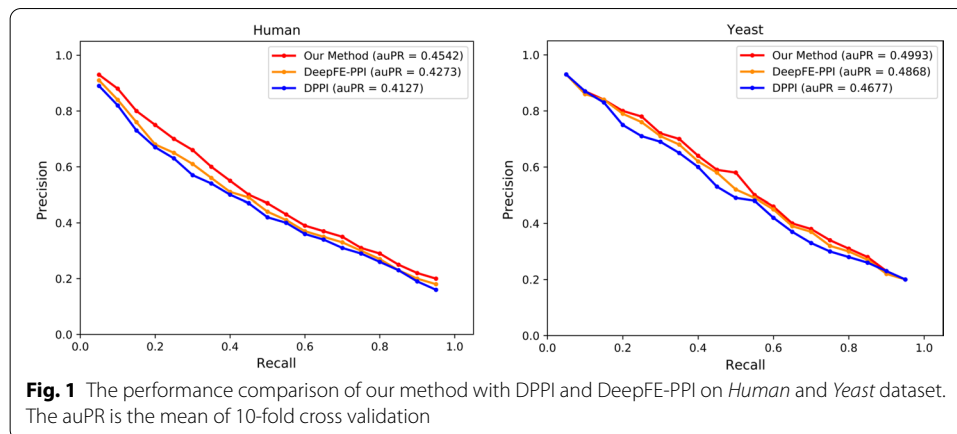


Table 1 The performance comparison of our method with DPPI and DeepFE-PPI on *S. cerevisiae* core dataset

	Precision	Recall	Accuracy
DPPI	0.9668	0.9224	0.9455
DeepFE-PPI	0.9645	0.9299	0.9478
Our method	0.9702	0.9355	0.9533

The results are obtained by 5-fold cross validation

Results

First, we compare our method with two state-of-the-art sequence-based methods, DPPI and DeepFE-PPI. We perform 10-fold cross validation on *Human* and *Yeast* datasets and compute the mean auPR (area under precision–recall curve), the results are shown in Fig. 1. In addition, following the same strategy as DPPI and DeepFE-PPI, we perform 5-fold cross validation on *S. cerevisiae* core dataset and report their average results in terms of precision, recall and accuracy, the results are shown in Table 1.

Next we conduct an ablation experiment to confirm the effectiveness of combining two types of information, we make predictions by only using protein amino acid sequence information or position information. Same as the first experiment, we perform 10-fold cross validation on *Human* and *Yeast* datasets and compute the mean auPR, the results are shown in Fig. 2. On *S. cerevisiae* core dataset, we perform 5-fold cross validation and count the precision, recall and accuracy, results are shown in Table 2.

Discussion

The results of the first experiment show that our method has an improvement over the previous methods. From Fig. 1, we can see that our method gets the best results on *Human* and *Yeast* datasets. On *Human* dataset, it achieves the largest mean auPR with the value of 0.4542 while DPPI is 0.4127 and DeepFE-PPI is 0.4273. Our method has 10.06% and 6.30% improvement over DPPI and DeepFE-PPI respectively. On *Yeast* dataset, our method also gets the largest mean auPR with the value of 0.4993 compared with DPPI's 0.4677 and DeepFE-PPI's 0.4868. It has 6.76% and 2.57% improvement over DPPI and DeepFE-PPI respectively. In addition, from Table 1, we can see that our method also

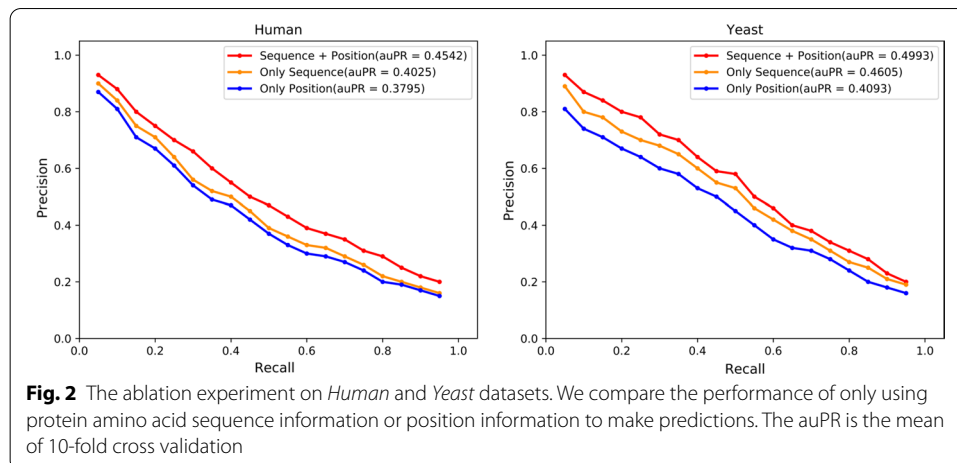


Table 2 The ablation experiment on *S. cerevisiae* core dataset

	Precision	Recall	Accuracy
Only Position	0.9004	0.9325	0.8992
Only Sequence	0.9587	0.9231	0.9372
Sequence + Position	0.9702	0.9355	0.9533

The results are obtained by 5-fold cross validation

achieves the best performance with 97.02% average precision, 93.55% average recall and 95.33% average accuracy on *S. cerevisiae* core dataset.

The first experiment proves the effectiveness of our method. From the above results, we can draw the conclusion that our method can make more accurate predictions. And compared with previous works, our method is simpler in representation phase. We use one-hot encoding to encode sequence information of protein and GCNs to capture position information, then we get the final representation matrix by combining them. While DPPI needs to generate a probabilistic sequence profile for each protein using PASI-BLAST and DeepFE-PPI needs to train a Word2vec model firstly and then uses the pre-trained model to generate representation vector for each protein, which is complicated and time-consuming.

In the second experiment, we conduct an ablation experiment to compare different representation methods. We only use amino acid sequence information or position information to make predictions, and compare the prediction results with the method of combining these two types of information. Figure 2 shows that combining sequence information and position information achieves the best prediction results on *Human* dataset and *Yeast* dataset. The same conclusion holds on the *S. cerevisiae* core dataset, as can be seen from Table 2.

In addition, we can also observe that the prediction results obtained using only amino acid sequence information are more accurate than the prediction results obtained using only position information, indicating that amino acid sequence information is more important when representing proteins. Combining these two types of information gets the best prediction results, it confirms the correctness of our method that position

information of protein in the PPIs networks graph can reflect the properties of proteins to some extent and is an important supplement to protein amino acid sequence information.

Conclusion and future work

In this paper, we propose a novel method that combining sequence information and position information to generate representations for proteins. When capturing position information of proteins in the PPIs networks graph, we use GCNs to aggregate feature information of neighbor nodes. In prediction phase, we design DNN modules to extract high-level features and make predictions. We conduct extensive experiments on three different benchmark datasets to verify the effectiveness of our method, and carry out in-depth analysis of the experiment results. Continuing work will improve the design of DNN architecture to get a better prediction performance.

Methods

PPIs prediction is essentially a classification problem and we need to identify that the given two proteins whether to interact. We train our model in a supervised way, it takes the representations of protein pairs as inputs and outputs a score representing the interaction probability. In order to illustrate our model clearly, we first introduce the principles and applications of GCNs. The following subsections will give further illustrations on the overall framework of our model, the method of encoding protein amino acid sequence information, the method of capturing proteins' position information in the PPIs networks graph and the design of DNN modules in the prediction phase.

Graph convolutional networks

Deep learning is a major advancement in the field of machine learning in recent years, which has aroused great interest of researchers and is widely used in several machine learning tasks, including computer vision, image analysis, speech recognition, information retrieval, natural language processing, reinforcement learning and multi-agent systems [16–23]. In addition, in the field of bioinformatics, deep learning is also widely used. For example, deep learning algorithms have been successfully applied to predict the association between *Human* diseases and microRNA, a type of non-coding RNA [24]. Compared with traditional machine learning methods, deep learning is suitable for processing and analyzing complex data, extracting and abstracting high-dimensional feature, which is helpful to process increasing amounts and dimensions of data generated by high throughput technique in bioinformatics.

However, in the real world, lots of data is generated from non-Euclidean domains and represented as graphs with complex relationships and interdependency between nodes. If we can make full use of the information of the graph structure, it will be of great help to solve the problem. Such as modeling the multiagent coordination problems in cooperative environments under the networked social learning framework under four representative topologies [25].

The characteristics of data with graph structure can be summarized as follows. Each node in the graph can be regarded as an object with its own unique attributes. And nodes are connected by edges, indicating that there is a certain relationship between

them. We need to comprehensively consider the attribute information of the node itself and the attribute information of its neighboring nodes to accurately make representations for nodes in graph-structured data.

In graph-structured data, the previous deep learning algorithm can not be directly applied. To solve the problem, researchers generalize the operation of convolution from traditional data to graph-structured data and propose GCNs, which generate the node's representation by aggregating feature information of its neighbors [26]. GCNs layer aggregates information from connected nodes to generate hidden representation for center nodes and then a non-linear transformation is applied to the hidden representation. By stacking multiple GCNs layers, the final hidden representation of each node receives message from further neighbors. The rule of aggregating neighbors' feature information in vector form can be described by Eq. 4,

$$h_i^{(l+1)} = \sigma \left(\sum_{j \in N_i} \frac{1}{c_{ij}} h_j^{(l)} W^{(l)} \right) \quad (4)$$

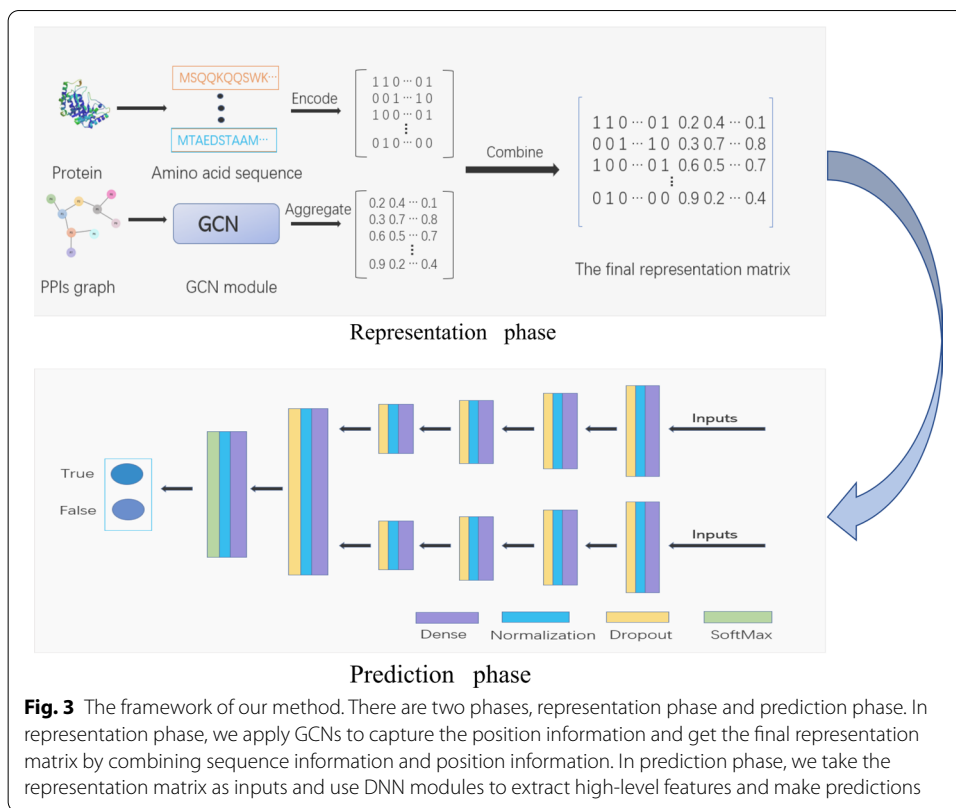
where $h_i^{(l+1)}$ is the hidden representation of node i in the $(l + 1)$ th layer, N_i is the set of node i 's neighbors, c_{ij} is an appropriately chosen normalization constant for the edge (v_i, v_j) , $W^{(l)}$ is a layer-specific weight matrix and $\sigma(\cdot)$ denotes a non-linear activation function.

GCNs take the graph structure information and node feature information as inputs and the outputs of GCNs can be different mechanisms according to different graph analytics task, including node-level, edge-level and graph-level [26].

GCNs are widely used in the processing of graph structure data, and have achieved excellent performance compared with previous methods. For example, in a citation network, partial nodes are labeled and others are unlabeled, using GCNs can learn an appropriate representation for each node, which is very important for predicting labels of the unlabeled nodes [27]. Similarly, there is also work proposing an improved spectral-based GCNs, which can work directly on directed graph data in semi-supervised nodes classification tasks [28]. In addition, GCNs are used to make multi-relational link prediction in a multimodal graph [29]. There is also work applying GCNs in the protein interface prediction problem, but different from ours, they represent a protein as a graph where each amino acid residue is a node whose features represent the properties of the residue [30]. In our paper, we represent the PPIs networks as a graph where each protein is a node. We use GCNs to capture position information of protein in the graph, which can be an important supplement to protein amino acid sequence information.

Design of the proposed model

In Fig. 3, we demonstrate the flow diagram of our method. The integrated flow diagram has two phases, representation phase and prediction phase. In representation phase, we process proteins' amino acid sequence information and PPIs networks graph information to generate representation for each protein. In prediction phase, we take the final representation matrix as inputs, which combines amino acid sequence information and position information, and use DNN modules to extract high-level features and make



predictions. Next, we will give details of the methods used in the representation and prediction phases.

One-hot encoding

One-hot encoding, also known as one-bit effective encoding, mainly uses bit status registers to encode each state. Each state is controlled by its own independent register bit, and only one bit is valid at any time. In practical machine learning applications, features are not always continuous values, and may be some categorical values. For such features, we usually need to digitize the features. For example, gender is a categorical attribute with two possible values, male or female. We can use [1, 0] for male and [0, 1] for female using one-hot encoding. One-hot encoding solves the problem that the classifier is not good at handling attribute data, and it also plays a role of expanding features to a certain extent.

Encode amino acid sequence information

Proteins are chains of amino acids that fold into a three dimensional structure that gives them their biochemical function. There are 20 different types of amino acids in organism. In this paper, we use one-hot encoding to encode amino acids, so each amino acid can be represented by a 20-dimensional vector consisting of 0 and 1.

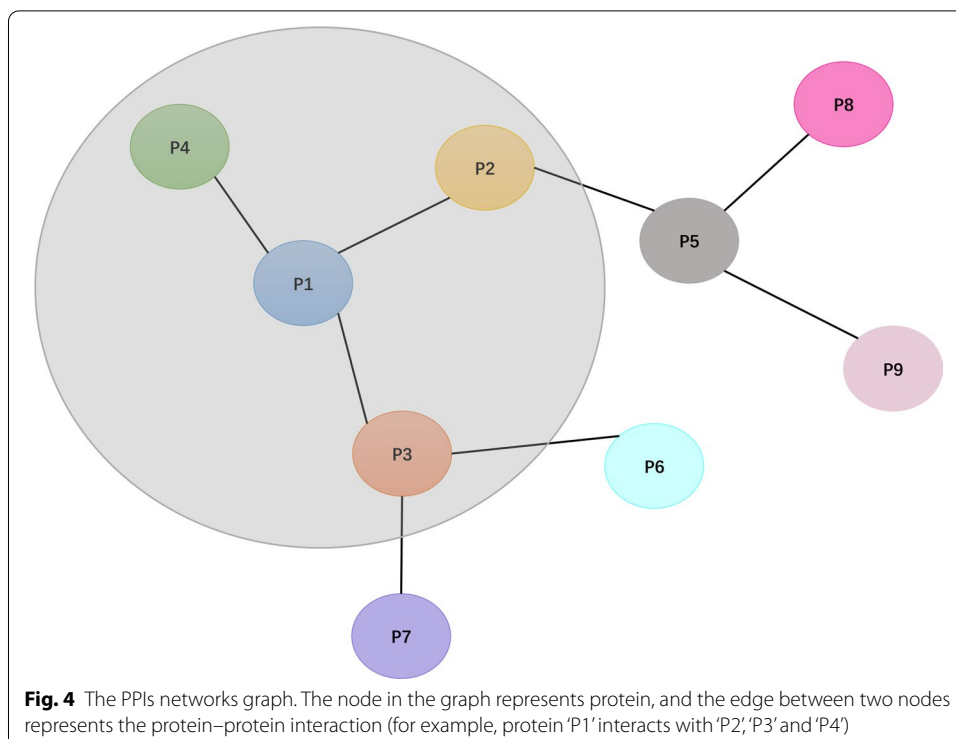
The detailed process is as follows, we use natural numbers ranging from 1 to 20 to give each unique amino acid an identity, and convert the original amino acid sequence to a vector of natural numbers. We construct an identity matrix with shape of 20 * 20, every

line in the matrix is the feature vector of a unique amino acid. Then we can convert the identity vector to feature vector by looking up in the identity matrix. After that, we get the amino acid sequence information of protein in the vector form, and for the convenience of subsequent processing, we set a fixed length for the obtained vector. If its length is less than the fixed length, we pad zeros to the front of the sequence and if its length is longer than the fixed length, we truncate the sequence in the front. In the previous methods, when encoding protein sequence information, the representation vector of the protein is usually a trainable parameter. Our method is more simple and efficient, and there are no parameters that need to be trained.

Capture proteins' position information

In our method, we model the PPIs networks as an unweighted and undirected graph where each protein is a node, as shown in Fig. 4. The edge between two nodes indicates that these two proteins can interact. The position information of protein in the PPIs networks graph reflects which proteins it can interact with, which is essentially a reflection of the protein's characteristics. Take protein 'P1' as an example, it interacts with 'P2', 'P3' and 'P4'. So when capturing position information of protein 'P1', we use GCNs to aggregate information about its neighbor nodes 'P2', 'P3' and 'P4'.

In our model, we take the graph structure information and node feature information as inputs. The graph structure information mainly includes the adjacency matrix and degree matrix of the graph, where the adjacency matrix describes the relationship between the nodes in the graph, and the degree matrix describes the number of connecting nodes of each node. The dimensions of these two matrices are determined by the number of nodes in the graph. Adjacency matrix and degree matrix are constructed



based on the training set, without using data of testing set, so there isn't a label leakage problem. The node feature information is also a matrix, where each row represents the feature information of a node, the number of rows is determined by the number of nodes, and the number of columns is the dimension of the feature vector of each node. Here we use one-hot encoding to encode each unique protein, so the dimension of the feature vector of each protein is the same as the number of proteins.

The way of GCNs capturing position information for each protein in the graph can be expressed by Eq. 5.

$$X_1^{N*f} = \sigma \left(\tilde{D}^{-1} \tilde{A} X_0^{N*N} W_0 \right) \quad (5)$$

In our work, one-layer GCNs achieve the best performance. Here, $\tilde{A} = A + I_N$ is the adjacency matrix of the PPIs networks graph with added self-connections, I_N is the identity matrix. $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ is the degree matrix, W_0 is the trainable weight matrix in the first layer and $\sigma(\cdot)$ denotes an activation function, such as $ReLU(\cdot)$. X_0^{N*N} is the original feature matrix, N is the number of proteins in the graph, here we use one-hot encoding to encode each unique protein, so the original feature matrix is the identity matrix with shape of $N * N$. X_1^{N*f} is the output feature matrix, f is the feature length of each node after GCNs operation. So after the GCNs operation, each node's hidden representation is composed of its original feature information and first-order neighbor's information, which contains the position information of the protein in the PPIs networks graph.

Design of DNN modules

To combine amino acid sequence information and position information, we concatenate the above two matrices from the amino acid sequence information and position information and get the final representation matrix, where each row is a feature vector of an unique protein. We take the final representation matrix as inputs of DNN modules. There are two separate DNN modules, each module processes one protein of the input pairs. These two DNN modules have the same structure, consisting of 4 fully connected neural network layers, 4 normalization neural network layers and 4 dropout neural network layers to extract high-level features that hidden in embedding vectors.

As an important achievement of deep learning in recent years, batch normalization has been widely proven to be effective and important. During model training, batch normalization uses the mean and standard deviation on small batches to continuously adjust the intermediate output of the neural network, so that the value of the intermediate output of the entire neural network in each layer is more stable. The use of batch normalization can make the convergence faster, the total training time is shorter and the effect is improved.

Dropout is another important trick widely used in deep learning, it means that during the training process of deep neural networks, the neural network unit is temporarily dropped from the network with a certain probability. For machine learning models, if the model has too many parameters and too few training samples, the trained model is prone to be overfitting. Dropout can effectively alleviate the problem of overfitting and improve the generalization ability of the model.

After extracting the features of the two proteins, we concatenate last hidden vectors of the both DNN modules. Then the concatenated vector is processed by a joint module composed of 2 fully connected neural network layers, and a softmax layer is used to predict interaction probability.

Abbreviations

PPIs: Protein–protein interactions; GCNs: Graph convolutional networks; DNN: Deep neural network; SGD: Stochastic gradient descent; auPR: Area under precision–recall curve.

Acknowledgements

We thank the reviewers' valuable comments for improving the quality of this work.

About this supplement

This article has been published as part of BMC Bioinformatics Volume 21 Supplement 16, 2020: Selected articles from the Biological Ontologies and Knowledge bases workshop 2019. The full contents of the supplement are available online at <https://bmcbioinformatics.biomedcentral.com/articles/supplements/volume-21-supplement-16>.

Authors' contributions

LL and XZ contributed to the algorithm design and theoretical analysis. YM, HP, YY, XH, YF, LW and JP contributed equally to the quality control and document reviewing. All authors read and approved the final manuscript.

Funding

Publication costs are funded by The National Natural Science Foundation of China (Grant No. U1836214) and Tianjin Nature fund, 19JCYBJC16300, Research on Data Platform Technology Based on Automotive Electronic Identification System. The funding bodies did not play any role in the design of the study, the collection, analysis, and interpretation of data, or in writing the manuscript.

Availability of data and materials

The datasets analysed during the current study are available in the zenodo repository, <https://doi.org/10.5281/zenodo.3960077>.

Ethics approval and consent to participate

Not applicable.

Consent to publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

¹ College of Intelligence and Computing, Tianjin University, No.135 Yaguan Road, Tianjin 300350, China. ² Automotive Data Center, CATARC, No.69 Xianfeng Road, Tianjin 300300, China. ³ School of Electronics and Information, Northwestern Polytechnical University, No.127 West Youyi Road, Xi'an 710072, China. ⁴ School of Computer Science, Northwestern Polytechnical University, No.127 West Youyi Road, Xi'an 710072, China.

Received: 17 November 2020 Accepted: 18 November 2020

Published: 16 December 2020

References

1. Ben-Hur A, Noble WS. Kernel methods for predicting protein–protein interactions. *Bioinformatics Suppl.* 2005;21:38–46.
2. You Z-H, Zhu L, Zheng C-H, Yu H-J, Deng S-P, Ji Z. Prediction of protein–protein interactions from amino acid sequences using a novel multi-scale continuous and discontinuous feature set. *BMC Bioinformatics.* 2014;15:9.
3. Ryan DP, Matthews JM. Protein–protein interactions in human disease. *Curr Opin Struct Biol.* 2005;15(4):441–6.
4. Browne F, Zheng H, Wang H, Azuaje F. From experimental approaches to computational techniques: a review on the prediction of protein–protein interactions. *Adv Artif Intell.* 2010;2010:924529.
5. Hashemifar S, Neyshabur B, Khan AA, Xu J. Predicting protein–protein interactions through sequence-based deep learning. *Bioinformatics.* 2018;34(17):802–10.
6. Guo Y, Yu L, Wen Z, Li M. Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences. *Nucleic Acids Res.* 2008;36(9):3025–30.
7. Du X, Sun S, Hu C, Yao Y, Yan Y, Zhang Y. Deepppi: boosting prediction of protein–protein interactions with deep neural networks. *J Chem Inf Model.* 2017;57(6):1499–510.
8. You Z-H, Chan KC, Hu P. Predicting protein–protein interactions from primary protein sequences using a novel multi-scale local feature representation scheme and the random forest. *PLoS One.* 2015;10(5):e0125811.
9. Szklarczyk D, Gable AL, Lyon D, Junge A, Wyder S, Huerta-Cepas J, Simonovic M, Doncheva NT, Morris JH, Bork P, et al. String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.* 2019;47(D1):607–13.

10. Ding Y, Tang J, Guo F. Identification of protein–protein interactions via a novel matrix-based sequence representation model with amino acid contact information. *Int J Mol Sci.* 2016;17(10):1623.
11. Yao Y, Du X, Diao Y, Zhu H. An integration of deep learning with feature embedding for protein–protein interaction prediction. *PeerJ.* 2019;7:7126.
12. Li H, Gong X-J, Yu H, Zhou C. Deep neural network based predictions of protein interactions using primary sequences. *Molecules.* 2018;23(8):1923.
13. Huang Y-A, You Z-H, Chen X, Chan K, Luo X. Sequence-based prediction of protein–protein interactions using weighted sparse representation model combined with global encoding. *BMC Bioinformatics.* 2016;17(1):184.
14. Hamp T, Rost B. Evolutionary profiles improve protein–protein interaction prediction from sequence. *Bioinformatics.* 2015;31(12):1945–50.
15. Davis J, Goadrich M. The relationship between precision–recall and ROC curves. In: Proceedings of the 23rd international conference on machine learning; 2006. p. 233–240.
16. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems; 2012. p. 1097–1105.
17. Bernal J, Kushibar K, Asfaw DS, Valverde S, Oliver A, Martí R, Lladó X. Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. *Artif Intell Med.* 2019;95:64–81.
18. Graves A, Mohamed A-R, Hinton G. Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE; 2013. p. 6645–6649.
19. Ma L, Lu Z, Shang L, Li H. Multimodal convolutional neural networks for matching image and sentence. In: Proceedings of the IEEE international conference on computer vision. 2015; p. 2623–2631.
20. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)
21. Yang Y, Hao J, Zheng Y, Yu C. Large-scale home energy management using entropy-based collective multiagent deep reinforcement learning framework. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19; 2019. p. 630–636
22. Fu H, Tang H, Hao J, Lei Z, Chen Y, Fan C. Deep multi-agent reinforcement learning with discrete-continuous hybrid action spaces. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19; 2019. p. 2329–2335
23. Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE. A survey of deep neural network architectures and their applications. *Neurocomputing.* 2017;234:11–26.
24. Peng J, Hui W, Li Q, Chen B, Wei Z, Hao J, Shang X. A learning-based framework for miRNA-disease association prediction using neural networks. *Bioinformatics.* 2019;35(21):4364–71.
25. Hao J, Huang D, Cai Y, Leung H-F. The dynamics of reinforcement social learning in networked cooperative multi-agent systems. *Eng Appl Artif Intell.* 2017;58:111–22.
26. Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS. A comprehensive survey on graph neural networks. arXiv preprint [arXiv:1901.00596](https://arxiv.org/abs/1901.00596) (2019)
27. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR; 2017.
28. Ma Y, Hao J, Yang Y, Li H, Jin J, Chen G. Spectral-based graph convolutional network for directed graphs. arXiv preprint [arXiv:1907.08990](https://arxiv.org/abs/1907.08990) (2019)
29. Zitnik M, Agrawal M, Leskovec J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics.* 2018;34(13):457–66.
30. Fout A, Byrd J, Shariat B, Ben-Hur A. Protein interface prediction using graph convolutional networks. In: Advances in neural information processing systems; 2017. p. 6530–6539

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

