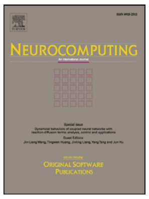
Neurocomputing 324 (2019) 10–19

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



Protein–protein interactions prediction based on ensemble deep neural networks

Long Zhang[a], Guoxian Yu[a], Dawen Xia[b,c], Jun Wang[a,*]

[a] College of Computer and Information Sciences, Southwest University, Chongqing 400715, China
[b] College of Data Science and Information Engineering, Guizhou Minzu University, Guiyang 550025, China
[c] College of National Culture and Cognitive Science, Guizhou Minzu University, Guiyang 550025, China

A R T I C L E I N F O

Article history:
Received 30 October 2017
Revised 3 February 2018
Accepted 26 February 2018
Available online 22 May 2018

Keywords:
Protein–protein interactions
Sequences of amino acids
Deep neural networks
Sequence descriptors
Ensemble DNNs

A B S T R A C T

Protein–protein interactions (PPIs) are of vital importance to most biological processes. Plenty of PPIs have been identified by wet-lab experiments in the past decades, but there are still abundant uncovered PPIs. Furthermore, wet-lab experiments are expensive and limited by the adopted experimental protocols. Although various computational models have been proposed to automatically predict PPIs and provided reliable interactions for experimental verification, the problem is still far from being solved. Novel and competent models are still anticipated. In this study, a neural network based approach called EnsDNN (*Ensemble Deep Neural Networks*) is proposed to predict PPIs based on different representations of amino acid sequences. Particularly, EnsDNN separately uses auto covariance descriptor, local descriptor, and multi-scale continuous and discontinuous local descriptor, to represent and explore the pattern of interactions between sequentially distant and spatially close amino acid residues. It then trains deep neural networks (DNNs) with different configurations based on each descriptor. Next, EnsDNN integrates these DNNs into an ensemble predictor to leverage complimentary information of these descriptors and of DNNs, and to predict potential PPIs. EnsDNN achieves superior performance with accuracy of 95.29%, sensitivity of 95.12%, and precision of 95.45% on predicting PPIs of *Saccharomyces cerevisiae*. Results on other five independent PPI datasets also demonstrate that EnsDNN gets better prediction performance than other related comparing methods.

1. Introduction

Protein–protein interactions (PPIs) play distinctly important roles in virtually all cellular processes, including signal transduction [1], immune response [2], and cell metabolism. Therefore, correctly identifying PPIs is not only helpful in understanding protein functions but also critical for structure-based drug design and disease treatment. Some high-throughput technologies have been invented for detecting PPIs, such as yeast two-hybrid, immune precipitation, X-ray crystallography, and protein chips [3]. Nevertheless, there are some limitations in these wet-lab experiments based techniques, such as time intensive, high cost, and small coverage [4,5]. These limitations have motivated the development of computational models to predict PPIs in large scale.

A number of computational approaches have been suggested to predict PPIs based on various data types [6–8]. These methods employ 3D structural information [9], Gene Ontology and annotations [10], phylogenetic profile, gene fusion [11–13] and the interacting proteins co-evolution pattern [14]. However, these approaches are not universal, their accuracy and reliability heavily depend on the prior knowledge of the collected proteins. In practice, the 3D structural of many proteins are unknown, Gene Ontology annotations of proteins are also incomplete [15–19], and PPIs of many species are completely unknown or rarely available, but abundant sequence data of these species are readily available [20,21]. Many computational methods have been explored to predict PPIs directly based on amino acid sequences, such as support vector machine (SVM) with conventional auto covariance (AC) [22,23], *k*-nearest neighbor (*k*NN) with local descriptor (LD) [24–27], SVM with conjoint triad method [28], random forest (RF) with multi-scale continuous and discontinuous local descriptor (MCD) [29,30], deep neural networks (DNNs) with amphiphilic pseudo amino acid composition descriptor (APAAC) [31,32], and so on. Extensive experiments have proved that amino acid sequences data alone can be competent to identify new PPIs. Shen et al. [28] grouped 20 standard amino acids into 7 classes according to their dipoles, volumes of the side chains, applied the conjoint triad method to extract

* Corresponding author.
E-mail address: kingjun@swu.edu.cn (J. Wang).

https://doi.org/10.1016/j.neucom.2018.02.097
0925-2312/© 2018 Elsevier B.V. All rights reserved.

feature from protein pairs, and then employed SVM to predict PPIs. This SVM based approach achieved a high accuracy of 83.9%. However, it does not take into account neighboring effect and PPIs generally exist in the non-continuous segments of amino acid sequences. Guo et al. [22] applied the auto-covariance (AC) method [23] to discover the information in the segments of discontinuous amino acid sequences and obtained an accuracy of 86.55% on PPIs of *Saccharomyces cerevisiae* (*S. cerevisiae*). You et al. [29] introduced a multi-scale continuous and discontinuous local feature descriptor to encode amino acid sequences. This descriptor hypothesizes that the segments of continuous amino acids with different segment lengths play important roles in identifying the interactions between proteins. Then they used the Minimum Redundancy Maximum Relevancy criterion [33], which can reduce feature abundance and computation complexity, to select an optimal feature subset. Finally, they employed SVM to predict PPIs. This multi-scale continuous and discontinuous local feature descriptor based solution obtained a high accuracy of 91.36%. Du et al. [32] employed the amphiphilic pseudo amino acid composition (APAAC) [31] to extract features from amino acid sequences. After that, they took the extracted features of two respective proteins as inputs of two separate deep neural networks (DNNs) to predict PPIs. This DNNs based method obtained an accuracy of 92.5% on PPIs of *S. cerevisiae*.

Different descriptors can extract different feature information of interacting protein sequences. The feature information generated by these descriptors are complementary to each other. Thus, we advocate using a number of different descriptors, which can obtain more information than a single descriptor alone, for PPIs prediction [34]. DNNs, a major recent advance in machine learning, can automatically learn a suitable representation of the raw data, discover high-level features, improve performance over traditional models, increase interpretability and provide additional understanding about the structure of the biological data [35,36]. Motivated by the characteristics of DNNs, we introduce a deep learning based approach called EnsDNN. EnsDNN firstly employs the auto covariance descriptor (AC) [23], local descriptor(LD) [24,25,27] and multi-scale continuous and discontinuous local descriptor (MCD) [29,30] to extract the interaction information of proteins from amino acid sequences, respectively. Then, EnsDNN uses these three feature sets as the inputs of 9 different and independent deep neural networks (DNNs) with different numbers of layers and neurons. After that, it takes the outputs of these 27 (9 DNNs × 3 descriptors) DNNs as inputs of a two-hidden layers neural network to produce the final ensemble prediction. EnsDNN not only combines the advantages of different descriptors, but also the diversity of 27 base DNNs, and thus produces a robust and competent ensemble predictor [37]. We perform experiments on PPIs of *S. cerevisiae* [32], EnsDNN achieves 95.29% accuracy with 95.12% sensitivity and 95.45% precision. Experimental results on other five independent datasets: *Caenorhabditis elegans* (4013 interacting pairs), *Escherichia coli* (6954 interacting pairs), *Homo sapiens* (1412 interacting pairs), *Helicobacter pylori* (1420 interacting pairs), and *Mus musculus* (313 interacting pairs) also show that EnsDNN makes more accurate prediction than other related and competitive methods [22,27,30,32,38].

2. Methodology

In this section, we elaborate on the proposed EnsDNN approach for predicting PPIs based on amino acid sequences. EnsDNN is consisted with the following three steps: (1) Encode the protein pairs interaction information into numeric vectors via the AC descriptor [23], LD descriptor [24,25,27] and MCD descriptor [29,30], respectively. (2) Train nine individual DNNs based on each of the three types of vectors. (3) Ensemble the twenty-seven individual DNNs via a two-hidden layers neural network. The flowchart of EnsDNN is shown in Fig. 1.

2.1. Feature vector extraction

Whether the extracted features are reliable or not can significantly affect the performance of PPIs prediction. Thus, how to effectively describe and encode the essential information of interacting protein pairs is a main challenge. To avoid the bias of using single descriptor alone, we use three representative and widely used feature descriptors (AC [23], LD [24,25,27], MCD [29,30]).

2.1.1. Auto covariance descriptor (AC)

PPIs can be divided into four interaction modes: electrostatic, hydrophobic, steric and hydrogen bond [23]. Seven physicochemical properties of amino acids are selected to reflect these interaction modes whenever possible. These properties include hydrophobicity (H_1) [39], hydrophilicity (H_2) [40], volumes of side chains of amino acids (VSC) [41], polarity (P_1) [42], polarizability (P_2) [43], solvent-accessible surface area [44] and net charge index of side chains [45]. Feature normalization can improve the accuracy and efficiency of mining algorithms on the data [46,47]. Given that, we firstly normalized data with zero mean and unit standard deviation (SD) as follows:

$$P'_{ij} = \frac{P_{i,j} - \tilde{P}_j}{D_j} \tag{1}$$

where $P_{i,j}$ is the jth physicochemical property value for the ith amino acid, $\tilde{P}_j$ is the mean of the jth physicochemical property over 20 amino acids and D_j is the corresponding standard deviation of the jth physicochemical property. Then each protein sequence is translated into seven vectors with each amino acid represented by the normalized values.

AC is a statistical tool proposed by Wold et al. [23], it was employed to transform amino acid sequences into uniform matrices. AC can account for the average interactions between residues, a certain *lag* apart throughout the entire sequence. To represent a protein sequence X with length L, the AC variables are computed as:

$$AC(lag, j) = \frac{1}{L - lag}\sum_{i=1}^{L-lag}\left(X_{ij} - \frac{1}{L}\sum_{i=1}^{L}X_{i,j}\right)\times\left(X_{(i+lag),j} - \frac{1}{L}\sum_{i=1}^{L}X_{i,j}\right) \tag{2}$$

lag is the distance between residues, X_{ij} is the jth physicochemical property of the ith amino acid of X. In this way, the number of AC variables can be calculated as $D = lg \times p$, where p is the number of descriptors, which is set as 7 according to seven properties of amino acids and lg is the maximum $lag (lag = 1, 2, \ldots, lg)$, which is set as 30 [22]. After each protein sequence is represented by a vector of AC variables, a protein pair is characterized by concatenating the vectors of the two respective proteins. Finally, we can use a 420-dimensional vector to encode interacting (non-interacting) protein pairs.

2.1.2. Multi-scale continuous and discontinuous local descriptor (MCD)

MCD is suggested by You et al. [29,30] to extract the interaction information of proteins. MCD firstly divides the entire amino acid sequences into a number of equal length segments and then uses a binary coding scheme to construct varying length segments. For example, "ACCLLACCAAALCCALLCACCCALCA" contains 26 residues as shown in Fig. 2; MCD first divides this sequence into 5 equal length segments (denoted by S_1, S_2, S_3, S_4 and S_5). Then it encodes the sequence using a 5-bit binary form with '1' and '0'. In each 5-bit binary, these combinations are written as 00001, 00010, 00011,

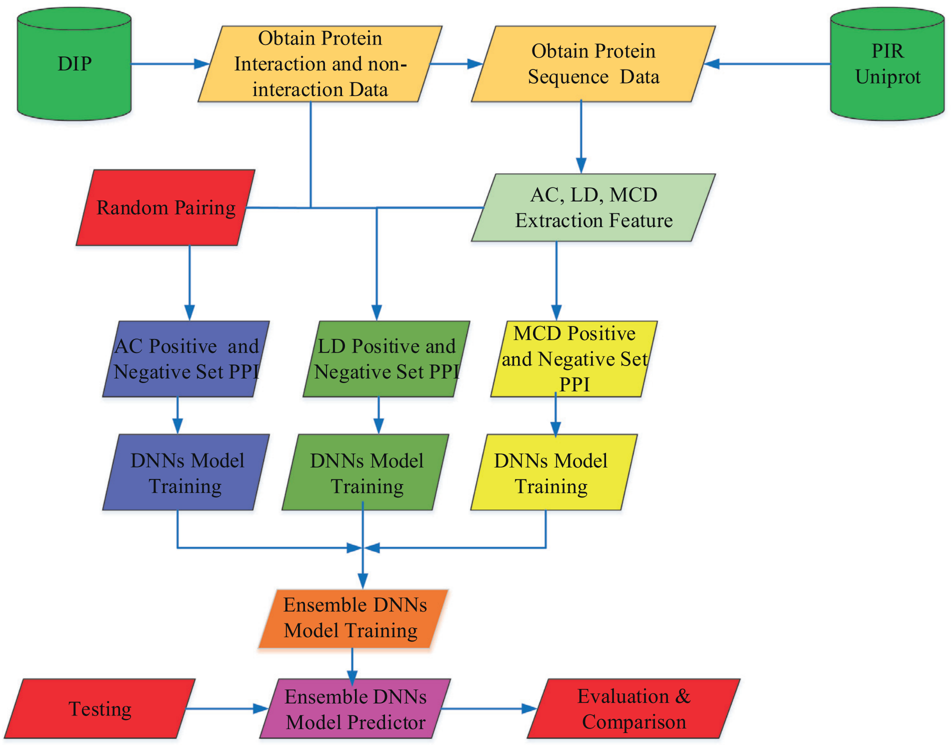

DIP
Obtain Protein Interaction and non-interaction Data
Obtain Protein Sequence Data
PIR Uniprot
Random Pairing
AC, LD, MCD Extraction Feature
AC Positive and Negative Set PPI
LD Positive and Negative Set PPI
MCD Positive and Negative Set PPI
DNNs Model Training
DNNs Model Training
DNNs Model Training
Ensemble DNNs Model Training
Testing
Ensemble DNNs Model Predictor
Evaluation & Comparison

Fig. 1. Flowchart of EnsDNN for predicting protein–protein interactions.

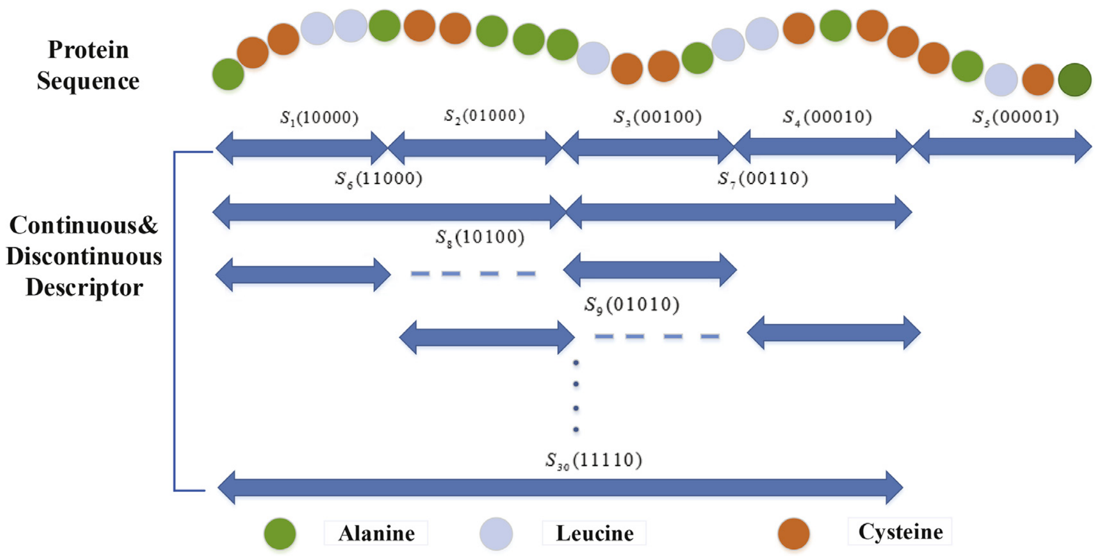


Fig. 2. An illustrative diagram for constructing continuous and discontinuous descriptor regions for a hypothetical protein sequence using 5-bit binary form. Each protein sequence is divided into 30 ($2^5 - 2$) sub-sequences ($S_1 - S_{30}$) of varying length to represent multi-overlapping continuous or discontinuous segments.

00100, 00101, 00110, 00111, $\cdots$, 11100, 11101, and 11110. A protein sequence can be represented by $2^5 - 2$ sub-protein sequence segments. Here 0 or 1 denote one of the five equal length segments $S_1 - S_5$ is excluded or included in constructing the continuous and discontinuous regions, respectively. For instance, 00110 represents a continuous region constructed by S_3 and S_4, 01010 denotes a discontinuous region constructed by S_2 and S_4. These regions are illustrated in Fig. 2.

To reduce the complexity inherent in the representation of the 20 standard amino acids, MCD firstly divides them into 7 groups based on the dipoles and volumes of the side chains (See Table 1) [27,28]. MCD extracts the interaction features of local protein sequences based on these 7 groups. For each sub-sequence, three descriptors, composition (C), transition (T) and distribution (D), are applied to describe its trait. C represents the proportion of each amino acid group; T represents the frequency with which amino acids in one group are followed by amino acids of another group; D measures the proportion of the chain length within which the first 25%, 50%, 75% and 100% of the amino acids of a particular group are located [48].

Then each sub-sequence is replaced by the index depending on its group. For instance, protein sequence “ACCLLACCAAALCCALLCACCCALCA” is replaced by “12233122111322133212221321” based on divided amino acid groups as shown in Fig. 3. There are

Table 1
Division of amino acids into seven groups based on the dipoles and volumes of the side chains.

Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	Group 7
A, G, V	C	F, I, L, P	M, S, T, Y	H, N, Q, W	K, R	D, E

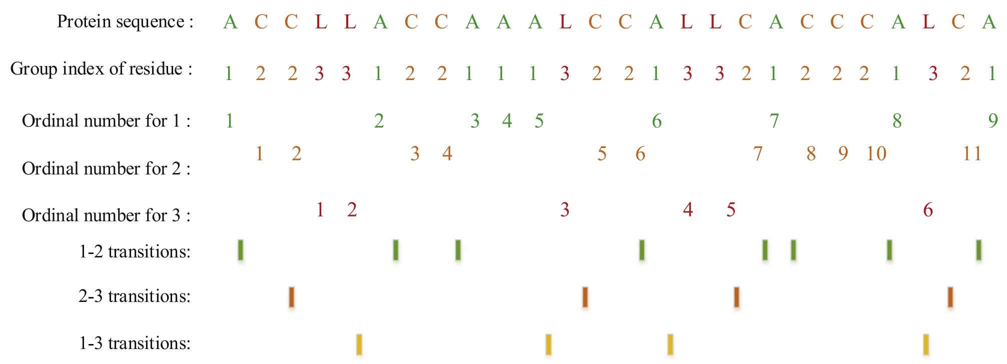

Protein sequence : A C C L L A C C A A A L C C A L L C A C C C A L C A
Group index of residue : 1 2 2 3 3 1 2 2 1 1 1 3 2 2 1 3 3 2 1 2 2 2 1 3 2 1
Ordinal number for 1 : 1 2 3 4 5 6 7 8 9
Ordinal number for 2 : 1 2 3 4 5 6 7 8 9 10 11
Ordinal number for 3 : 1 2 3 4 5 6
1-2 transitions:
2-3 transitions:
1-3 transitions:

Fig. 3. Sequence of a hypothetic protein indicating the construction of composition, transition and distribution description descriptors of a protein region.

nine '1', eleven '2' and six '3' in this sequence. The composition for these three symbols is $9\times100\%/(9+11+6)=34.61\%$, $11\times100\%/(9+11+6)=42.31\%$, and $6\times100\%/(9+11+6)=23.08\%$, respectively. There are 8 transitions from '1' to '2' or from '2' to '1' in this sequence, and the percentage frequency of these transitions is $(8/25)\times100\%=32\%$. Similarly, we can calculate the transitions from '1' to '3' or '3' to '1', and transitions from '2' to '3' or '3' to '2' with $(4/25)\times100\%=16\%$ and $(4/25)\times100\%=16\%$, respectively.

Next, MCD calculates the distribution of these three symbols. For example, there are 9 residues encoded as '1' in Fig. 3, the position of the first residue '1', the second residue '1' ($25\%\times9\approx2$), the fifth residue '1' ($50\%\times9\approx5$), the seventh residue '1' ($75\%\times9\approx7$), and the ninth residue '1' ($100\%\times9\approx9$) in the encoded sequence are 1, 6, 11, 19 and 26, respectively. Thus D descriptor for '1' is: $(1/26\times100\%=3.84\%)$, $(2/26\times100\%=7.69\%)$, $(5/26\times100\%=19.23\%)$, $(7/26\times100\%=26.92\%)$ and $(9/26\times100\%=34.62\%)$, respectively. Similarly, the D descriptor for '2' and '3' are 7.69%, 26.92%, 53.85%, 76.92%, 96.15% and 15.38%, 19.23%, 46.15%, 65.39%, 92.31%, respectively.

For each continuous and discontinuous region, the three descriptors (C, T, D) are calculated and concatenated, and a total of 63 descriptors are generated: 7 for C, 21 ($7\times6/2$) for T and 35 (7×5) for D. Then all descriptors from 30 (2^5-2) segments are concatenated into a 1890-dimensional vector. Finally, MCD concatenates the numeric feature vectors of two individual proteins. Thus, a 3780-dimensional vector is constructed to characterize each protein pair and then used as input for DNNs.

2.1.3. Local descriptor (LD)

LD extracts feature in a more simple way than MCD. LD also groups 20 standard amino acids into 7 groups based on the dipoles and volumes of the side chains, as shown in Table 1, but it divides the entire protein sequence into 10 regions as shown in Fig. 4 and extracts each segment information based on composition(C), transition(T), distribution(D) introduced in the previous subsection of MCD [24,25,27]. Each region can generate 63 attributes, 7 for C, 21 for T, and 35 for D. Thus, an entire protein sequence can be encoded by a 630-dimensional vector. Finally, it concatenates the two vectors of two individual proteins and obtains a 1260-dimensional vector to represent the information of pairwise proteins.

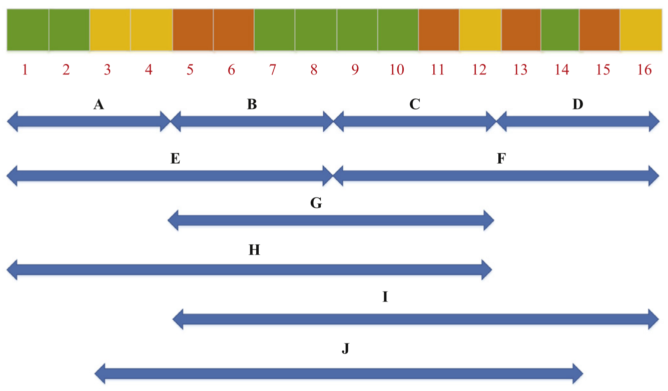


Fig. 4. Schematic diagram for constructing ten descriptor regions (A)–(J) for a hypothetical protein sequence. Adapted from Tong et al. [25] and Davies et al. [49]. The regions (A)–(D) and (E)–(F) are respectively generated by dividing the whole sequence into four equal regions and two equal regions. The region G, H, I, and J stand for the central 50%, the first 75%, the final 75%, and the central 75% of the entire sequence, respectively.

2.2. Ensemble deep neural networks

2.2.1. Construct separate deep neural networks

Artificial neural network, inspired by the biological neural network that constitutes animal brains, consists of layers of interconnected nodes (neurons) [35]. Each connection between neurons can transmit a signal to another neuron. The depth of a neural network corresponds to the number of hidden layers, and the width is the maximum number of neurons of its layers [35]. Neural network with a large number (four or more) of hidden layers is called deep neural networks (DNNs) [35,50]. In general, a neural network feeds the data into the input layer, and then transforms these data in a nonlinear way through multiple hidden layers, and outputs the prediction by the output layer (See the left part of Fig. 5 for example). Neurons of a hidden layer or output layer are connected to all neurons of the previous layer. Each neuron computes a weighted sum of its inputs and applies a nonlinear activation function to calculate its outputs $f(\mathbf{x})$ (as shown in the right part of Fig. 5). The most popular activation function is the rectified linear unit (ReLU is a $max(0,\ x)$ function that thresholds negative signals to 0 and passes through positive signals), which can accelerate model

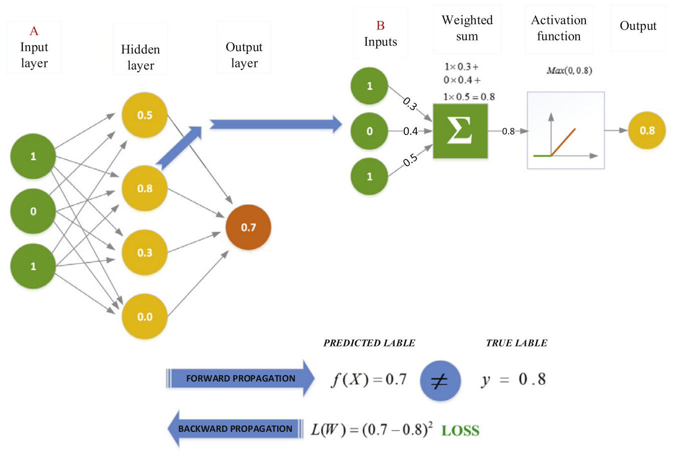


Fig. 5. Neural network training procedure. The left part is the architecture of one-hidden layer, the right part is the procedure of computing the output of one-hidden layer.

training. In this work, the loss is calculated by the cross entropy function, which can speed up the training and obtain better prediction results [51]. The network is defined as:

$$\mathbf{H}_{i1} = \sigma_1(\mathbf{W}_{i1}\mathbf{X}_{i1} + \mathbf{b}_{i1})(i = 1, \dots, n) \tag{3}$$

$$\mathbf{H}_{i(j+1)} = \sigma_1(\mathbf{W}_{ij}\mathbf{H}_{ij} + \mathbf{b}_{ij})(j = 1, \dots, h) \tag{4}$$

$$L = -\frac{1}{n}\sum_{i=1}^{n}[\mathbf{y}_i ln(\sigma_2(\mathbf{W}_{ih}\mathbf{H}_{ih} + \mathbf{b}_{ih}) + (1 - \mathbf{y}_i) ln(1 - \sigma_2(\mathbf{W}_{ih}\mathbf{H}_{ih} + \mathbf{b}_{ih}))] \tag{5}$$

where n is the number of PPIs for batch training. σ_1 is the activation function of ReLU, σ_2 is the activation function of output layer with sigmoid, $\mathbf{X}$ is the batch training inputs, $\mathbf{H}$ is the outputs of hidden layer, and $\mathbf{y}$ is the corresponding desired outputs. h is the depth of DNNs, and $\mathbf{W}$ is the weight matrix between the input layer and output layer, and $\mathbf{b}$ is the bias.

In this study, we individually construct 27 DNNs using TensorFlow platform, as depicted in Fig. 1. Three types of feature vectors extracted by AC [23], LD [24,25,27] or MCD [29,30] are separately used as the inputs for 9 independent neural networks with different configurations. These neural networks have different learning rates, batch sizes, and dropout rates. For each neural network, we use the mini-batch gradient descent [52] and Adam algorithm [53] to reduce the sensitivity to the specific choice of learning rate and to speed up training. In order to avoid overfitting, the dropout technique (the most common regularization) and the $L1$-norm, $L2$-norm are used at the same time. The activation of some neurons is randomly set to zero ("dropped out") during training in each forward pass, and intuitively results in an ensemble of different networks, whose predictions are averaged, as shown in Fig. 6. The dotted line means this neuron is dropout; in other words, this neuron will not be activated and calculated. We also use the batch normalization approach to reduce the dependency of training with the parameter initialization and overfitting, and to speed up training.

2.2.2. Ensemble strategy

Aggregating the outputs of multiple predictors can generally improve the performance of a single predictor [37]. However, to obtain a good aggregation effect, the individual predictors must be as accurate and diverse as possible. There are three commonly strategies to maintain the diversity: (1) using different algorithms to learn from the data, such as artificial neural network, support vector machine, decision tree and so on; (2) changing the internal structure of a given algorithm, for instance, the depth and width of the neural network, the number of trees in the decision tree classifier; (3) learning from different feature views [37]. In this study, we use strategy (2) and (3) to maintain the diversity. In practice, DNNs is very sensitive to small changes and thus different configurations of base DNNs can result in diverse predictors. To fuse these predictors, EnsDNN uses a two-hidden layers neural network to synergize their outputs and to predict PPIs. To further study the performance of EnsDNN under different numbers of based DNNs, we fuse different number of base DNNs and find that the performance is best and stable when the ensemble size is around 27, so we fuse 27 base DNNs. The flowchart of EnsDNN is shown in Fig. 1.

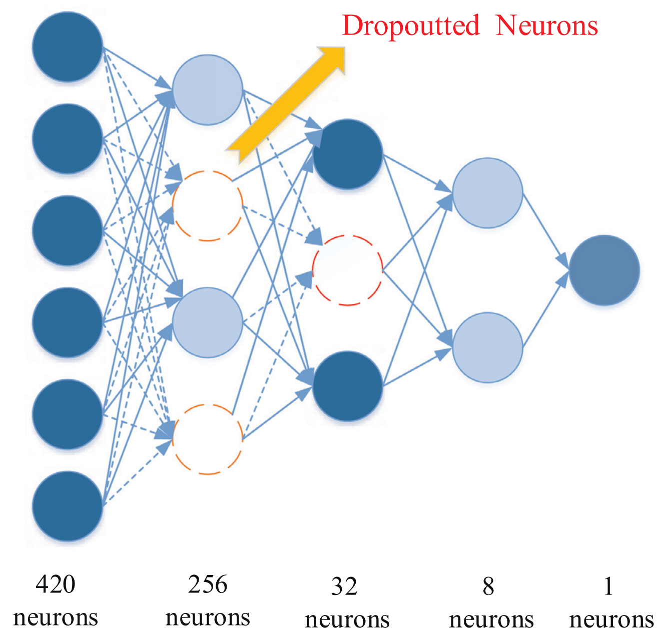


Fig. 6. The structure of a deep neural network with AC descriptor and the dropout technique.

3. Experimental results and analysis

3.1. Experimental setup

To quantitatively evaluate the performance of EnsDNN, we collected PPIs of *S. cerevisiae* used by Du et al. [32] from the Database of Interacting Proteins (DIP) [54]. This dataset contains 17257 positive protein pairs after removing protein pairs that contain a protein with fewer than 50 amino acids and 40% sequence identity [22]. Selecting negative examples is very crucial for training a predictor for PPIs [49]. The negative set was obtained by pairing proteins whose subcellular localizations are different. This selection should meet the following requirements [22,28]: (1) the non-interacting pairs cannot appear in the positive data set, and (2) the contribution of proteins in the negative set should be as harmonious as possible. Finally, 48,594 negative pairs were generated based on these requirements. The final PPI dataset of *S. cerevisiae* consisted of 34,514 protein pairs, where half were from the positive data set and the other were from the negative data set (randomly selected 17,257 negative pairs from all negative pairs). Five independent PPI datasets, including *Caenorhabditis elegans* (4013 interacting pairs), *Escherichia coli* (6954 interacting pairs), *Homo sapiens* (1412 interacting pairs), *Mus musculus* (313 interacting pairs) and *Helicobacter pylori* (1420 interacting pairs) [38], which were used to evaluate previously proposed methods, were also

Table 2
Recommended parameters of DNNs in the experiments.

Name	Range	Recommendation		
		AC	LD	MCD
Learning rate	1,0.1,0.001,0.0001,0.0007	0.0007	0.001	0.0003
Weight initialization	Uniform, normal, lecun_uniform, glorot_normal, glorot_uniform	glorot_normal	glorot_normal	glorot_normal
Per-parameter Adaptive learning rate	SGD, RMSprop, Adagrad, Adadelta, Adam, Adamax, Nadam	Adam	Adam	Adam
Activation function	relu, tanh, sigmoid, softmax, softplus,	relu	relu	relu
Dropout rate	0.5, 0.4, 0.7	0.5, 0.4	0.5, 0.7	0.5, 0.7
Depth	2, 3, 4, 5, 6, 7, 8 ,9	2, 3, 4	3, 4, 5	4, 5, 6, 7, 9
Width	2^2, 2^3, 2^4, 2^5, 2^6, 2^7, 2^8, 2^9, 2^{10}, 2^{11}	2^2, 2^3, 2^4, 2^5, 2^6, 2^7, 2^8	2^2, 2^3, 2^4, 2^5, 2^6, 2^7, 2^8, 2^9,	2^2, 2^3, 2^4, 2^5, 2^6, 2^7, 2^8, 2^9, 2^{10}, 2^{11}

Table 3
Parameter setup for comparing methods.

Method	Parameters			
SVM+AC [22]	C	γ		Kernel
	32768.0	0.074325444687670064		Poly
kNN+LD [27]	n_neighbors	Weights	Algorithm	p
	3	Distance	Auto	1
SVM+LD [38]	C	γ		Kernel
	3.1748021	0.07432544468767006		rbf
RF+MCD [30]	n_estimators	max_features	criterion	Bootstrap
	5000	Auto	Gini	True

collected to evaluate the performance of EnsDNN. These datasets are available at http://ailab.ahu.edu.cn:8087/DeepPPI/index.html.

EnsDNN was implemented using TensorFlow platform (https://www.tensorflow.org/). We separately implemented 27 DNNs as the base predictors and then constructed a two-hidden layers neural network to fuse the predicted outputs of these predictors. The flowchart of EnsDNN is shown in Fig. 1. The hyper-parameters of 27 base DNNs are differently specified to enlarge the diversity of these base predictors, and thus to produce accurate and robust consensus predictor. The best hyper-parameter configuration is data and application dependent, and models with different configurations can be evaluated on the test dataset. We summarized the recommended parameter setup of EnsDNN in Table 2. As to the parameter setup of the comparing methods, we used the grid search approach to obtain the optimal parameters. The optimal parameters are shown in 3. For Du et al. work [32], we obtained the information via http://ailab.ahu.edu.cn:8087/DeepPPI/index.html. All the experiments were carried out on a desktop with Intel (R) Core (TM) i5-4590, 16 GB RAM and Win 7.

To comparatively evaluate the prediction performance of EnsDNN, five-fold cross-validation is adopted. In other words, each PPI dataset is randomly divided into five equal subsets, and each subset is used as a testing set in turn, meanwhile the other four subsets are used as training set. We use the following metrics to measure the performance of these methods: overall prediction accuracy *(ACC)*, recall *(RE)*, specificity *(SPE)*, precision *(PE)*, matthews correlation coefficient *(MCC)*, F_1 score values, and area under the receiver operating characteristic curve *(AUC)*. The first six metrics are defined as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$RE = \frac{TP}{TP + FN} \tag{7}$$

$$SPE = \frac{TN}{TN + FP} \tag{8}$$

$$PE = \frac{TP}{TP + FP} \tag{9}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{11}$$

where *TP* (true positive) is the number of true PPIs that are correctly predicted, *TN* (true negative) is the number of true non-interacting pairs that are correctly predicted, *FP* (false positive) is the number of the wrongly predicted interacting pairs, and *FN* (false negative) is the number of the true interacting pairs that are failed to be predicted. MCC is a measure of the quality of binary classifications, which is a correlation coefficient between the observed and predicted results. MCC equal to 0 means completely random prediction, -1 means completely wrong prediction and 1 means perfect prediction. F_1 score is a harmonic average of precision and recall. Receiver operating characteristic curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. This curve is the plot of the true positive rate versus the false positive rate under different thresholds. AUC is the area under the curve and the value of AUC can be used to compare predictors. A larger AUC indicates a better predictor.

3.2. Results on PPIs of *S. cerevisiae*

To evaluate the robustness of EnsDNN, five-fold cross-validation is employed to reduce the impact of data dependency and to improve the reliability of the results. Table 4 reports the results of EnsDNN on five individual folds (fold 1–5) and the overall results of five folds. From Table 4, we can see that the overall accuracy of EnsDNN is ranging from 94.86 to 95.31%. In order to comprehensively evaluate the performance of EnsDNN, the results with respect to other six evaluation metrics (including *RE, SPE, PE, MCC,* F_1 and *AUC*) are also included. EnsDNN achieves competent prediction performance with an average sensitivity of 95.12%, specificity of 95.48%, precision of 95.45%, MCC of 90.59%, F_1 of 95.29% and AUC of 97.00%.

We compare EnsDNN against the approaches proposed by Guo et al. [22], Yang et al. [27], Zhou et al. [38], You et al. [30], and Du et al. [32] and reveal their results in Table 4. These comparing approaches were introduced in the Section 1, they separately used AC [22,23], LD [24,25,27], MCD [29,30] or APAAC [31] to encode amino acid sequences, and then made prediction using SVM, k-nearest neighbor (kNN), random forest (RF) or DNNs to predict PPIs. From Table 4, we can observe that EnsDNN generally outperforms these state-of-the-art PPIs predictors. That is because EnsDNN resorts to different classifiers and feature representations. EnsDNN has higher *RE, PE, SPE, MCC,* F_1 than other comparing methods. Based on these results, we can conclude that EnsDNN is more effective than other comparing methods in predicting PPIs. These observations can be

Table 4
Results of five-fold cross validation on PPIs of *S. cerevisiae.*

Method		ACC	PE	RE	SPE	MCC	**F$_1$**	AUC
EnsDNN	Fold 1	95.31%	95.31%	95.26%	95.35%	90.61%	95.29%	96.89%
	Fold 2	94.86%	94.24%	95.33%	94.40%	89.72%	94.78%	97.19%
	Fold 3	96.00%	96.50%	95.50%	96.51%	92.01%	96.00%	97.19%
	Fold 4	95.19%	96.11%	94.35%	96.06%	90.40%	95.22%	97.15%
	Fold 5	95.10%	95.10%	95.15%	95.06%	90.21%	95.12%	96.60%
		95.29% ± 0.43%	**95.45% ± 0.89%**	**95.12% ± 0.45%**	**95.48% ± 0.82%**	**90.59% ± 0.86%**	**95.29% ± 0.44%**	**97.00% ± 0.26%**
EnsDNN-Con		90.68% ± 0.55%	91.19% ± 2.22%	90.14% ± 2.44%	91.19% ± 2.67%	81.43% ± 1.06%	90.62% ± 0.55%	96.45% ± 0.37%
EnsDNN-Sep		91.19% ± 0.57%	90.41% ± 1.81%	92.23% ± 1.02%	90.17% ± 2.14%	82.44% ± 1.10%	91.29% ± 0.48%	96.59% ± 0.19%
DNNs+APAAC [32]		92.58% ± 0.38%	94.21% ± 0.45%	90.95% ± 0.41%	94.41% ± 0.45%	85.41% ± 0.76%	92.55% ± 0.39%	**97.55% ± 0.16%**
RF+MCD [30]		89.15% ± 0.33%	90.00% ± 0.57%	88.10% ± 0.17%	90.21% ± 0.61%	78.33% ± 0.67%	89.04% ± 0.31%	94.78% ± 0.21%
SVM+LD [38]		88.76% ± 0.37%	89.44% ± 0.27%	87.89% ± 0.45%	89.62% ± 0.30%	77.53% ± 0.53%	88.66% ± 0.28%	94.69% ± 0.31%
*k*NN+LD [27]		84.81% ± 0.37%	87.53% ± 0.14%	81.18% ± 0.84%	88.44% ± 0.18%	69.80% ± 0.71%	84.23% ± 0.47%	90.03% ± 0.31%
SVM+AC [22]		87.88% ± 0.56%	88.16% ± 0.90%	87.53% ± 0.59%	88.24% ± 1.02%	75.77% ± 1.12%	87.84% ± 0.53%	93.69% ± 0.33%

attributed to that the adopted three feature descriptors of amino acid sequences can explore more patterns of PPIs and the base DNNs trained on these feature descriptors are complementary to each other.

To show the effectiveness and non-redundancy of our ensemble strategy, we designed two variants of EnsDNN. One of them is EnsDNN-Con, the other is EnsDNN-Sep. EnsDNN-Con firstly concatenates three feature representations (AC [22,23], LD [22,23] and MCD [29,30]), and then takes the concatenated features as input of a dropout network [55]. The hidden layers for this network fixed as 2048-1024-256-32, which means 4 hidden layers with 2048, 1024, 256, 32 neurons, respectively. EnsDNN-Sep has three DNNs, it separately takes AC [22,23], LD [22,23] and MCD [29,30] feature representation as inputs. The hidden layers for these three DNNs are fixed as 256-128-4, 512-128-4, and 1024-512-128-4, respectively. After that, it concatenates the outputs of these three DNNs to generate 12 neurons as the input of a dropout DNNs. The average results of five-fold cross validation are also reported in Table 4. From Table 4, we can observe that the performance of EnsDNN is better than EnsDNN-Con and EnsDNN-Sep. EnsDNN-Con has relatively lower performance among these three methods. EnsDNN-Con only uses single DNNs with dropout, it cannot adequately extract the complementarity of different descriptors. The performance of EnsDNN-Sep is better than EnsDNN-Con, except PE and SPE. That is because EnsDNN-Sep uses three DNNs with dropout, it can extract more complementarity of different descriptors than EnsDNN-Con. EnsDNN trains 27 base DNNs with dropout for ensemble and achieves the best performance. These results prove that although the dropout is also an ensemble method, using it alone cannot effectively and adequately extract the complementarity of different descriptors. In contrast, using the explicit ensemble strategy can more adequately employ the complementarity and diversity of base DNNs, and thus improve the performance of EnsDNN.

To further investigate the contribution of using different feature descriptors, we separately trained 27 DNNs based on AC [22,23], LD [24,25,27], and MCD [29,30], and then integrated these DNNs in the same way as EnsDNN. We report the results in Fig. 7. From the figure, we can observe that the LD [24,25,27] descriptor is better than AC [22,23] descriptor, because LD has more features than AC, it can encode more interactions information. MCD [29,30] descriptor is only 0.23% better than LD descriptor on ACC. That is because they are similar. From Fig. 7, we can find that EnsDNN+AC+LD+MCD performs significantly better than its cousins that use single type of feature sets extracted by AC, LD, or MCD. From the results in this figure, we can conclude that these three descriptors are complementary to each other.

To study the performance of EnsDNN under different numbers of base DNNs, we fixed the number of base DNNs as 3, 9, 12,..., 45, respectively. The results of EnsDNN under each fixed number

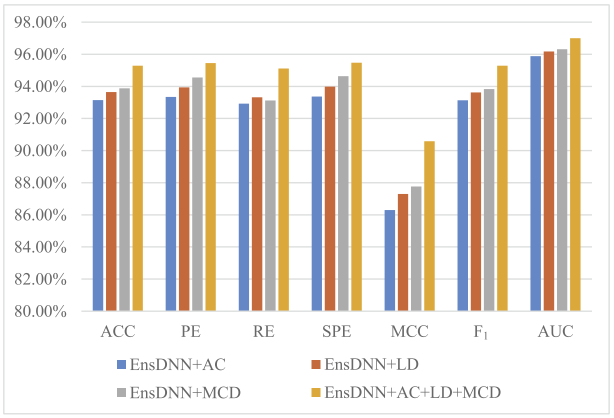


Fig. 7. Comparison of the prediction performance between EnsDNN+AC, EnsDNN+LD, EnsDNN+MCD and EnsDNN+AC+LD+MCD.

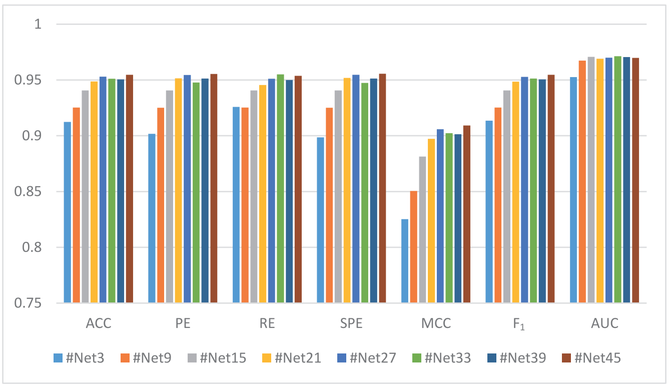


Fig. 8. Prediction performance under different numbers of base DNNs.

of DNNs are plotted in Fig. 8. We can see the performance of EnsDNN steadily increases as the number of base DNNs increasing and keeps relatively stable when the number of base DNNs is ≥ 27. To balance the efficiency and effectiveness, we fix the number of base DNNs as 27 for experiments.

Meanwhile, to further investigate the contribution of using an ensemble predictor to fuse the outputs of 27 DNNs. We integrated the outputs of 27 base DNNs by using averaging, the results are reported in Fig. 9. From Fig. 9, we can observe that EnsDNN has better performance than averaging across all evaluation metrics. The reason is that the averaging method equally treats the outputs of 27 base DNNs, it may be misled by some low quality base DNNs. In contrast, by using two additional layers to fuse the outputs of these base DNNs, EnsDNN can assign different weights to them.

Table 5
Comparison of training times of different comparing methods.

Method	EnsDNN	DNNs+APAAC [32]	RF+MCD [30]	SVM+LD [38]	*k*NN+LD [27]	SVM+AC [22]
Time (s)	23400	519	1341	18820	488	6925

Table 6
Prediction results on five independent PPI datasets, PPIs of *S. cerevisiae* are used as the training set.

Species	Test pairs	ACC		
		EnsDNN (%)	DNNs+APAAC [32] (%)	SVM+LD [38] (%)
C. elegans	4013	93.22	**94.84**	75.73
E. coli	6984	**95.10**	92.19	71.24
H. sapiens	1412	**95.00**	93.77	76.27
H. pylori	1420	89.14	**93.66**	75.87
M. musculus	313	**94.06**	91.37	76.68

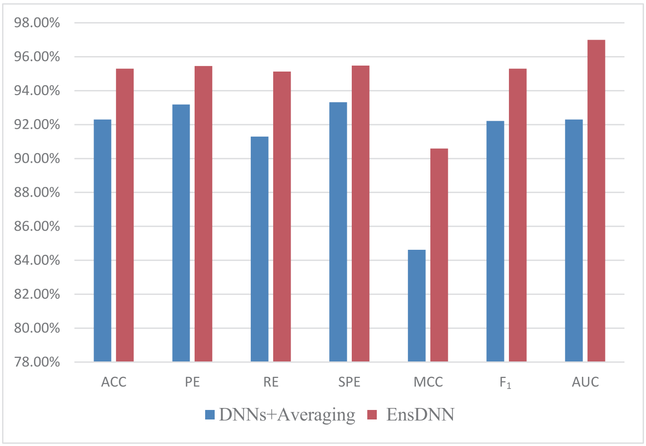


Fig. 9. Comparison of the prediction performance between averaging 27 base DNNs and fusing 27 base DNNs by two-hidden layers neural network.

From these results, we can conclude that the additional two layers for fusing DNNs can improve the performance of prediction.

We also report the training times of different comparing methods in Table 5. From Table 5, we can observe that EnsDNN takes the most time for training, but it has the highest prediction performance across all evaluation metrics, except AUC. That is because EnsDNN has to train 27 base DNNs and fuse them. As well as that, it trains DNNs based on MCD [29,30] feature representation, which encodes amino acid sequences by a 3780-dimensional numeric vector and asks for DNNs with 4 or 5 hidden layers. The more hidden layers, the more training parameters and training time are. *k*NN has the smallest training time but its performance is the lowest. Since SVM has a high time complexity, SVM+LD also takes a long time for training.

3.3. Results on independent datasets

To further evaluate the practical prediction ability of EnsDNN. We firstly trained EnsDNN using PPIs of *S. cerevisiae* dataset and then applied it on five independent datasets, including *C. elegans* (4013 interacting pairs), *E. coli* (6954 interacting pairs), *Homo sapiens* (1412 interacting pairs), *H. pylori* (1420 interacting pairs), and *M. musculus* (313 interacting pairs). The prediction results are shown in Table 6. From Table 6, we can see that the accuracy of EnsDNN on *C. elegans, E. coli, H. sapiens, H. pylori*, and *M. musculus* are 93.22%, 95.10%, 95.00%, 89.14%, and 94.06%, respectively. EnsDNN obtains larger accuracy than DeepPPI [32] and SVM+LD [38] on *E. coil, H. sapiens*, and *M. musculus*. The accuracy of EnsDNN on other two datasets is slightly lower than that of DeepPPI, but still much higher than that of SVM+LD. We can see the accuracy is over 93% on *C. elegans, E. coli, H. sapiens* and *M. musculus*, while the accuracy on *H. pylori* is only 89.14%. The reason is that *S. cerevisiae* has closer relationship with *C. elegans, H. sapiens* and *M. musculus* than *H. pylori*. The prediction results show that EnsDNN is capable of predicting PPIs of the other species based on PPIs of another species, and it holds good generalization ability.

4. Conclusion

In this paper, we proposed an effective and accurate sequence-based approach called EnsDNN to predict PPIs. EnsDNN firstly extract the feature information of protein sequences by AC descriptor [23], LD descriptor [24,25,27], and MCD descriptor [29,30], respectively. These descriptors can complementarily capture the interactions information from the continuous and discontinuous amino acid segments. Three types of interactions feature sets were obtained based on these three descriptors, respectively. Then, nine independent DNNs with different configurations were trained on each feature set, resulting in 27 DNNs. Finally, we adopted a two-hidden layers neural network to integrate these DNNs. Multiple metrics were applied to evaluate the performance of EnsDNN. Meanwhile, PPIs of five independent species were further used to assess the practical prediction ability of EnsDNN. The experimental results showed that EnsDNN generally outperforms current machine learning based PPIs predictors and fusing these three descriptors can complement each other very well.

There are three factors contribute to the competitive performance of EnsDNN. The first factor is that we utilized three descriptors to extract feature information of amino acid sequences, which can capture more interactions information than a single descriptor alone. The second factor is that we used nine DNNs with different configurations for each descriptor to generate diverse base DNNs, and the diversity among base DNNs improved the robustness and accuracy of EnsDNN. The third factor is that we integrated the predicted outputs of base DNNs by a two-hidden layers neural network to nonlinearly fuse these base DNNs.

Acknowledgment

This work is supported by Natural Science Foundation of China (61741217, 61402378 and 61762020), Natural Science Foundation of CQ CSTC (cstc2016jcyjA0351), Science and Technology Foundation of Guizhou (QKHJC20161076), Science and Technology Top-notch Talents Support Project of Colleges and Universities in Guizhou (QJHKY2016065), Fundamental Research Funds for the Central Universities of China (2362015XK07).

References

[1] X.M. Zhao, R.S. Wang, L. Chen, K. Aihara, Uncovering signal transduction networks from high-throughput data by integer linear programming, Nucl. Acids Res. 36 (9) (2008) e48, doi:10.1093/nar/gkn145.

[2] N.E. Williams, Immunoprecipitation procedures, Methods Cell Biol. 62 (2000) 449, doi:10.1016/S0091-679X(08)61549-6.

[3] H. Zhu, M. Bilgin, R. Bangham, D. Hall, A. Casamayor, P. Bertone, N. Lan, R. Jansen, S. Bidlingmaier, T. Houfek, Global analysis of protein activities using proteome chips, Science 293 (5537) (2001) 2101–2105, doi:10.1126/science.1062191.

[4] P. Uetz, L. Giot, G. Cagney, T.A. Mansfield, R.S. Judson, J.R. Knight, D. Lockshon, V. Narayan, M. Srinivasan, P. Pochart, et al., A comprehensive analysis of protein–protein interactions in *Saccharomycescerevisiae*, Nature 403 (6770) (2000) 623–627, doi:10.1038/35001009.

[5] T.T. Aumentado-Armstrong, B. Istrate, R.A. Murgita, Algorithmic approaches to protein–protein interaction site prediction, Algorithms Mol. Biol. 10 (1) (2015) 7, doi:10.1186/s13015-015-0033-9.

[6] X.-M. Zhao, X. Li, L. Chen, K. Aihara, Protein classification with imbalanced data, Prot. Struct. Funct. Bioinf. 70 (4) (2008) 1125–1132, doi:10.1002/prot.21870.

[7] W.W. Lam, K.C. Chan, Discovering functional interdependence relationship in PPI networks for protein complex identification, IEEE Trans. Biomed. Eng. 59 (4) (2012) 899–908, doi:10.1109/TBME.2010.2093524.

[8] B.A. Shoemaker, A.R. Panchenko, Deciphering protein–protein interactions. Part II. Computational methods to predict protein and domain interaction partners, PLoS Comput. Biol. 3 (4) (2007) e43, doi:10.1371/journal.pcbi.0030043.

[9] G.R. Smith, M.J.E. Sternberg, Prediction of protein–protein interactions by docking methods, Curr. Opin. Struct. Biol. 12 (1) (2002) 28–35, doi:10.1016/S0959-440X(02)00285-3.

[10] H. Lee, M. Deng, F. Sun, T. Chen, An integrated approach to the prediction of domain–domain interactions, BMC Bioinf. 7 (1) (2006) 1–15, doi:10.1186/1471-2105-7-269.

[11] E.M. Marcotte, M. Pellegrini, H.L. Ng, D.W. Rice, T.O. Yeates, D. Eisenberg, Detecting protein function and protein–protein interactions from genome sequences, Science 285 (5428) (1999) 751–753, doi:10.1126/science.285.5428.751.

[12] A.J. Enright, I. Iliopoulos, N.C. Kyrpides, C.A. Ouzounis, Protein interaction maps for complete genomes based on gene fusion events, Nature 402 (6757) (1999) 86–90, doi:10.1038/47056.

[13] D.-S. Huang, C.-H. Zheng, Independent component analysis-based penalized discriminant method for tumor classification using gene expression data, Bioinformatics 22 (15) (2006) 1855–1862, doi:10.1093/bioinformatics/btl190.

[14] R. Jothi, P.F. Cherukuri, A. Tasneem, T.M. Przytycka, Co-evolutionary analysis of domains in interacting proteins reveals insights into domain–domain interactions mediating protein–protein interactions, J. Mol. Biol. 362 (4) (2006) 861, doi:10.1016/j.jmb.2006.07.072.

[15] G. Yu, G. Fu, J. Wang, Y. Zhao, NewGOA: predicting new GO annotations of proteins by bi-random walks on a hybrid graph, IEEE/ACM Trans. Comput. Biol. Bioinf. 99 (1) (2017) 1–14, doi:10.1109/TCBB.2017.2715842.

[16] G. Fu, J. Wang, B. Yang, G. Yu, NegGOA: negative GO annotations selection using ontology structure, Bioinformatics 32 (19) (2016) 2996, doi:10.1093/bioinformatics/btw366.

[17] P.D. Thomas, V. Wood, C.J. Mungall, S.E. Lewis, J.A. Blake, On the use of gene ontology annotations to assess functional similarity among orthologs and paralogs: a short report, PLoS Comput. Biol. 8 (2) (2012) e1002386, doi:10.1371/journal.pcbi.1002386.

[18] S.-P. Deng, L. Zhu, D.-S. Huang, Predicting hub genes associated with cervical cancer through gene co-expression networks, IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB) 13 (1) (2016) 27–35, doi:10.1109/TCBB.2015.2476790.

[19] C.-H. Zheng, D.-S. Huang, L. Zhang, X.-Z. Kong, Tumor clustering using nonnegative matrix factorization with gene selection, IEEE Trans. Inf. Technol. Biomed. 13 (4) (2009) 599–607, doi:10.1109/TITB.2009.2018115.

[20] S.-P. Deng, D.-S. Huang, SFAPS: an R package for structure/function analysis of protein sequences based on informational spectrum method, Methods 69 (3) (2014) 207–212, doi:10.1016/j.ymeth.2014.08.004.

[21] D.-S. Huang, L. Zhang, K. Han, S. Deng, K. Yang, H. Zhang, Prediction of protein–protein interactions based on protein–protein correlation using least squares regression, Curr. Prot. Pept. Sci. 15 (6) (2014) 553–560, doi:10.2174/1389203715666140724084019.

[22] Y. Guo, L. Yu, Z. Wen, M. Li, Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences, Nucl. Acids Res. 36 (9) (2008) 3025–3030, doi:10.1093/nar/gkn159.

[23] S. Wold, J. Jonsson, M. Sjöström, M. Sandberg, S. Rännar, DNA and peptide sequences and chemical processes multivariately modelled by principal component analysis and partial least-squares projections to latent structures, Anal. Chim. Acta 277 (2) (1993) 239–253, doi:10.1016/0003-2670(93)80437-P.

[24] M.N. Davies, A. Secker, A.A. Freitas, E. Clark, J. Timmis, D.R. Flower, Optimizing amino acid groupings for GPCR classification, Bioinformatics 24 (18) (2008) 1980–1986, doi:10.1093/bioinformatics/btn382.

[25] J.C. Tong, M.T. Tammi, Prediction of protein allergenicity using local description of amino acid sequence, Front. Biosci. A J. Virt. Lib. 13 (16) (2008) 6072, doi:10.2741/3138.

[26] J. Cui, L.Y. Han, H. Li, C.Y. Ung, Z.Q. Tang, C.J. Zheng, Z.W. Cao, Y.Z. Chen, Computer prediction of allergen proteins from sequence-derived protein structural and physicochemical properties, Mol. Immunol. 44 (4) (2007) 514–520, doi:10.1016/j.molimm.2006.02.010.

[27] L. Yang, J.F. Xia, J. Gui, Prediction of protein-protein interactions from protein sequence using local descriptors, Prot. Pept. Lett. 17 (9) (2010) 1085, doi:10.2174/092986610791760306.

[28] J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, H. Jiang, Predicting protein–protein interactions based only on sequences information, Proc. Natl. Acad. Sci. 104 (11) (2007) 4337–4441, doi:10.1073/pnas.0607879104.

[29] Z.H. You, L. Zhu, C.H. Zheng, H.J. Yu, S.P. Deng, Z. Ji, Prediction of protein–protein interactions from amino acid sequences using a novel multi-scale continuous and discontinuous feature set, BMC Bioinf. 15 (S15) (2014) S9, doi:10.1186/1471-2105-15-S15-S9.

[30] Z.H. You, K.C. Chan, P. Hu, Predicting protein–protein interactions from primary protein sequences using a novel multi-scale local feature representation scheme and the random forest, PLoS ONE 10 (5) (2015) e0125811, doi:10.1371/journal.pone.0125811.

[31] K.C. Chou, Using amphiphilic pseudo amino acid composition to predict enzyme subfamily classes, Bioinformatics 21 (1) (2005) 10, doi:10.1093/bioinformatics/bth466.

[32] X. Du, S. Sun, C. Hu, Y. Yao, Y. Yan, Y. Zhang, DeepPPI: boosting prediction of protein-protein interactions with deep neural networks, J. Chem. Inf. Model. (2017), doi:10.1021/acs.jcim.7b00028.

[33] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226, doi:10.1109/TPAMI.2005.159.

[34] S.-P. Deng, L. Zhu, D.-S. Huang, Mining the bladder cancer-associated genes by an integrated strategy for the construction and analysis of differential co-expression networks, BMC Genom. 16 (3) (2015) S4, doi:10.1186/1471-2164-16-S3-S4.

[35] C. Angermueller, T. Pärnamaa, L. Parts, O. Stegle, Deep learning for computational biology, Mol. Syst. Biol. 12 (7) (2016) 878, doi:10.15252/msb.20156651.

[36] D. Huang, Systematic Theory of Neural Networks for Pattern Recognition, Publishing House of Electronic Industry, China, 1996. Google Scholar.

[37] P.M. Granitto, P.F. Verdes, H.A. Ceccatto, Neural network ensembles: evaluation of aggregation algorithms, Artif. Intell. 163 (2) (2005) 139–162, doi:10.1016/j.artint.2004.09.006.

[38] Y.Z. Zhou, Y. Gao, Y.Y. Zheng, Prediction of protein-protein interactions using local description of amino acid sequence, Commun. Comput. Inf. Sci. 202 (2011) 254–262, doi:10.1007/978-3-642-22456-0_37.

[39] C. Tanford, Contribution of hydrophobic interactions to the stability of the globular conformation of proteins, J. Am. Chem. Soc. 84 (22) (1962) 4240–4247, doi:10.1021/ja00881a009.

[40] T.P. Hopp, K.R. Woods, Prediction of protein antigenic determinants from amino acid sequences, Proc. Natl. Acad. Sci. 78 (6) (1981) 3824, doi:10.1073/pnas.78.6.3824.

[41] W. Krigbaum, A. Komoriya, Local interactions as a structure determinant for protein molecules: II, Biochim. Biophys. Acta (BBA)Prot. Struct. 576 (1) (1979) 204–228, doi:10.1016/0005-2795(79)90498-7.

[42] R. Grantham, Amino acid difference formula to help explain protein evolution, Science 185 (4154) (1974) 862–864, doi:10.1126/science.185.4154.862.

[43] M. Charton, B.I. Charton, The structural dependence of amino acid hydrophobicity parameters, J. Theor. Biol. 99 (4) (1982) 629–644, doi:10.1016/0022-5193(82)90191-6.

[44] G.D. Rose, A.R. Geselowitz, G.J. Lesser, R.H. Lee, M.H. Zehfus, Hydrophobicity of amino acid residues in globular proteins, Science 229 (4716) (1985) 834–838, doi:10.1126/science.4023714.

[45] P. Zhou, F.F. Tian, B. Li, S.R. Wu, Z.L. Li, Genetic algorithm-based virtual screening of combinative mode for peptide/protein, Acta Chim. Sin. 64 (7) (2006) 691–697.

[46] L. Al Shalabi, Z. Shaaban, B. Kasasbeh, Data mining: a preprocessing engine, J. Comput. Sci. 2 (9) (2006) 735–739.

[47] H.-J. Yu, D.-S. Huang, Normalized feature vectors: a novel alignment-free sequence comparison method based on the numbers of adjacent amino acids, IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB) 10 (2) (2013) 457–467, doi:10.1109/TCBB.2013.10.

[48] I. Dubchak, I. Muchnik, S.R. Holbrook, S.H. Kim, Prediction of protein folding class using global description of amino acid sequence, Proc. Natl. Acad. Sci. 92 (19) (1995) 8700, doi:10.1073/pnas.92.19.8700.

[49] M.J. Davis, W. Simon, S. Chang, M.A. Ragan, Protein–protein interaction as a predictor of subcellular location, BMC Syst. Biol. 3 (1) (2009) 28, doi:10.1186/1752-0509-3-28.

[50] D.-s. Huang, Radial basis probabilistic neural networks: model and application, Int. J. Pattern Recognit. Artif. Intell. 13 (07) (1999) 1083–1101, doi:10.1142/S0218001499000604.

[51] N. Michael, Neural networks and deep learning, (http://neuralnetworksanddeeplearning.com/index.html). Accessed Oct 30, 2017.

[52] S. Ruder, An overview of gradient descent optimization algorithms, arXiv:1609.04747 (2016) pp. 1.

[53] D. Kingma, J.A. Adam, Adam: A method for stochastic optimization, arXiv:1412.6980 (2014) pp. 1.

[54] I. Xenarios, L. SalwÃnski, X.J. Duan, P. Higney, S.M. Kim, D. Eisenberg, DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions, Nucl. Acids Res. 30 (1) (2002) 303, doi:10.1093/nar/30.1.303.

[55] Y. Li, F.-X. Wu, A. Ngom, A review on machine learning principles for multi-

view biological data integration, Brief. Bioinf. (2016) bbw113, doi:10.1093/bib/bbw113.

Long Zhang is a Graduate in the College of Computer and Information Science, Southwest University, Chongqing, China. He received B.Sc. degree in Computer Science from Taiyuan University of Science and Technology, Shanxi, China. His current research interests include data mining and bioinformatics.

Guoxian Yu is an Associate Professor in the College of Computer and Information Science, Southwest University, Chongqing, China. He received B.Sc. degree in Software Engineering from Xi'an University of Technology, Xi'an, China in 2007, and Ph.D. in Computer Science from South China University of Technology, Guangzhou, China in 2013. He was a Postdoc Research Fellow at the Department of Computer Science, Hong Kong Baptist University, Hong Kong from 2014 to 2015. His current research interests include data mining and bioinformatics. He has served as PC member for KDD, ICDM, SDM and other prestigious conferences. He is a recipient of Best Poster Award of SDM12 Doctral Forum and Best Student Paper Award of 10th IEEE International Conference on Machine Learning and Cybernetics (ICMLC).

Dawen Xia is an Associate Professor with the College of Data Science and Information Engineering, and College of National Culture and Cognitive Science, Guizhou Minzu University, Guiyang, China. He received the Ph.D. degree from Southwest University, Chongqing, China. His research interests include big data analytics, multi-agent systems, parallel and distributed computing, and spatio-temporal data mining.

Jun Wang is an Associate Professor in the College of Computer and Information Science, Southwest University, Chongqing, China. She received B.Sc. degree in Computer Science, M.Eng. degree in Computer Science and Ph.D. in Artificial Intelligence from Harbin Institute of Technology, Harbin, China in 2004, 2006 and 2010, respectively. Her current research interests include machine learning, data mining and their applications in bioinformatics.