IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON FEATURE REPRESENTATION AND LEARNING METHODS WITH APPLICATIONS IN LARGE-SCALE BIOLOGICAL SEQUENCE ANALYSIS

Received June 3, 2020, accepted June 16, 2020, date of publication June 29, 2020, date of current version July 23, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3005444

Efficient ResNet Model to Predict Protein-Protein Interactions With GPU Computing

SHENGYU LU, QINGQI HONG, BEIZHAN WANG, AND HONGJI WANG
School of Informatics, Xiamen University, Xiamen 361005, China

Corresponding author: Qingqi Hong (hongqq@xmu.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61502402, and in part by the Fundamental Research Funds for the Central Universities under Grant 207220180073.

ABSTRACT Protein-protein interactions (PPI) play an important role in the cell activities of organisms. The deep research about PPI can help humans understand the mechanism of life activities and apply protein functions better. Nowadays, PPI prediction algorithms based on amino acid sequences using the recurrent neural network (RNN) can overcome the disadvantages of traditional biological experimental methods and achieve high accuracy. However, these algorithms are usually time-consuming and cannot take full advantage of graphics processing units (GPU) with efficient computation performance to accelerate PPI prediction, because the RNN model considers the time series of sequences. In this paper, we propose an efficient algorithm based on the residual network (ResNet) model to predict PPI (ResPPI). Our algorithm uses the embedding method to represent amino acid sequences, combining the advantages of powerful feature extraction capabilities of the ResNet with deep layers and GPU performance. The experimental results show that the ResPPI algorithm can ensure high accuracy and reduce training time greatly. Based on the ordinary GPU device, compared with the state-of-the-art LSTM model, the speed of the ResPPI algorithm is five times faster than that of the LSTM, whereas the ResPPI algorithm can achieve similar accuracy to the LSTM. Besides, in the case of unbalanced datasets, the ResPPI algorithm can perform better.

INDEX TERMS Protein-protein interactions (PPI), residual network (ResNet), graphics processing units (GPU) computing, training speed.

I. INTRODUCTION

Protein-protein interactions (PPI) are the basis of many cellular biological processes [1] such as signal transduction, immune response, DNA replication and cell metabolism. Due to the significance of PPI, the research about PPI is becoming increasingly important. Traditional biological experimental methods [2] such as two-hybrid and mass spectrometry provide important clues and evidence to identify PPI, but those methods are expensive and labor-intensive [3].

Over the decades, with the development of computer technology, researchers have proposed using computational methods to predict PPI based on collected protein data to achieve PPI prediction accurately and rapidly. Since the generation of high-throughput technology [4], researchers have been able to obtain huge amounts of protein data, which benefits the development of computational methods to predict PPI.

The associate editor coordinating the review of this manuscript and approving it for publication was Quan Zou.

Nowadays, scholars have proposed many methods to predict PPI based on various types of information about proteins, such as protein domains, amino acids and physical properties [5]. This information and PPI labels can be collected from some open databases such as the Protein Data Bank (PDB), the Molecular Interaction Database and the Human Protein Reference Database (HPRD) [6].

Many studies have suggested approaches to predict PPI based on protein sequences and the experimental results have shown that sequence-based approaches are available [7]. Compared with other types of information about proteins, protein sequences are more convenient to collect and process, so sequence-based approaches are popular among researchers. Li *et al.* proposed a domain-based method to predict PPI using probabilities of putative interacting domain pairs derived from interacting protein pairs and non-interacting protein pairs [8]. Airola *et al.* proposed a graph kernel-based approach to extract the features of proteins automatically and achieve PPI prediction [9].

They considered all dependency paths between two pairs because those paths were important indicators.

Although much progress has been made [5], those algorithms to predict PPI based on protein sequences usually use sequence models such as the LSTM [7] and gated recurrent units (GRU) [8] to extract the semantic features and implement classification [9]. Since sequence models consider the time series, they cannot take full advantage of graphics processing units (GPU) for efficient computing [10]. Even in the case of high-quality GPU devices, sequence models are still time-consuming. Due to the fact that the CNN and machine learning models can make full use of GPU for parallel computing, some researchers suggested using those models with GPU to predict PPI. For example, Chen *et al.* proposed a method of hyperparameter estimation in the support vector machine (SVM) with GPU acceleration for PPI prediction. However, this method cannot extract the deep features of proteins and achieve satisfying results [11]. For these defects of sequence models, in this paper, we propose an efficient algorithm based on the residual network (ResNet) to predict PPI (ResPPI). The main novelty lies in that the ResNet consists of residual units, which does not consider the time series of words and has the capacity to use GPU for acceleration [10]. Besides, with the increasing number of convolution layers, the ResNet will not overfit and degenerate because of residuals [12], so it has the capacity to extract the deep features of proteins and achieve high accuracy, even in the case of unbalanced datasets. Therefore, the ResPPI algorithm can ensure high accuracy and reduce the training time greatly, which improves the performance of PPI prediction. Furthermore, it can provide a reference for the processing of other biological data such as drug-drug interactions [13] and protein-RNA interactions [14].

The organization of this paper is as follows. Section I introduces the PPI and our proposed ResPPI algorithm. Section II reviews the related research about GPU computing and the CNN for text. Section III describes the ResPPI algorithm in detail. Section IV shows the performance results of our method and the baseline methods in the experiments. Section V presents conclusions and future work.

II. BACKGROUND AND RELATED WORK

Compared with the co-occurrence methods based on protein sequences to predict PPI, machine learning methods have shown better effects [15]. Common approaches to predict PPI include the SVM, logistic regression, random forests and artificial neural networks [16]. Shen *et al.* proposed a method for PPI prediction using the information of protein sequences [17]. They combined the kernel function and conjoint triad features to describe amino acids and used the SVM method to achieve prediction. Bock *et al.* proposed an efficient algorithm to predict PPI based on protein sequences [18]. They extracted the semantic features from protein sequences, combining the physical features of amino acids in proteins such as charge properties, water solubility, and then adopted the SVM to implement classification. You *et al.* used multi-scale continuous and discontinuous local feature descriptors to encode amino acid sequences [19]. They assumed that consecutive amino acid fragments with different fragment lengths and used them to predict PPI. To select the best features, they used the minimum redundancy and maximum correlation criterion, which can reduce the dimensions and computational complexity. Finally, they used the SVM classifier to predict PPI.

Deep learning methods with powerful feature extraction capabilities have been applied in many fields such as computer vision [20], nature language process (NLP) [21] and bioinformatics [22]. Some scholars have proposed using deep learning methods to improve the performance of PPI prediction [23]. Du *et al.* proposed a method called DeepPPI to predict PPI [24]. They used the amphiphilic pseudo amino acid composition (APAAC) to extract features from the protein sequences, and then inputted the features of two proteins into two independent deep neural networks (DNN) to predict PPI. Zhang *et al.* proposed the ensemble neural networks to predict PPI based on different representations of amino acid sequences [25]. They used the auto-covariance descriptor and several kinds of local descriptors to represent and explore the pattern of interactions between amino acid residues, and trained the DNN based on each descriptor and integrated them into an ensemble predictor. In particular, some state-of-the-art methods to predict PPI is based on the recurrent neural network (RNN). Yadav *et al.* proposed a novel algorithm based on the deep bidirectional long short-term memory (Bi-LSTM) that exploited word sequences and dependency path related information to predict PPI [26]. They also proposed an algorithm based on the attentive deep RNN, which combined multiple levels of representations using word sequences and dependency path related information to predict PPI [27]. Ahmed proposed a novel tree RNN with the attention mechanism to predict PPI [28]. They used the long short-term memory (LSTM) to model the dependency tree structure of sentences and achieved a significant improvement than other methods without explicit feature extraction. Other researchers suggested using the models based on the convolutional neural network (CNN) to predict PPI [29]. Hua *et al.* proposed the shortest dependency path based on the CNN (sdpCNN) model to predict PPI [30]. They took the shortest dependency path and word embedding as input, and made full use of structure information and avoided manual feature selection by the CNN. Peng *et al.* proposed a method named McDepCNN for PPI prediction [31]. They used a multichannel dependency-based CNN to extract the features of protein sequences and implement classification. Hashemifar *et al.* proposed a novel framework named DPPI to model and achieve PPI prediction from protein sequences [1]. They applied a Siamese-like CNN to capture information about the composition of amino acids and used a novel random project to investigate the combination of protein motifs.

A. GPU COMPUTING

GPU consists of thousands of small and efficient cores [32], which are designed for dealing with multiple tasks simultaneously. Compared with CPU, GPU has more processing units and performed better effects on accelerated computing, because CPU only has the limited number of cores, designed for handling serial tasks. Although the deep learning methods have powerful feature extraction capabilities and greatly improve the performance effects through multiple layers and massive parameters, we also have the challenge of high computational time. Many researchers suggest using GPU to accelerate the calculation and reduce the time costs [33] when dealing with massive biological data. Sun *et al.* presented a scarified backpropagation technique for neural network learning based on GPU to achieve acceleration [34]. Loc *et al.* proposed a mobile deep learning framework named DeepMon with GPU computing to execute the DNN for continuous video [35]. Song *et al.* presented a distributed and dynamically tuned framework with GPU computing for CNN based big data processing and achieve acceleration [36]. Wu *et al.* presented a CPU-GPU implementation of a graph clustering heuristic named Shingling [37]. They used CPU and GPU for different stages of computation, using GPUs for the time-consuming steps to accelerate the calculation. Pang *et al.* presented a novel GPU version of Mrbayes for processing large protein data [38]. They used the Kahan summation to improve accuracy and reduce runtime. Dubey *et al.* presented a parallel approach with GPU computing for protein structure prediction [32] using the 2D triangular hydrophobic-polar lattice model. The experimental results showed that the approach significantly improved the performance of prediction and reduced the computation greatly.

B. THE CNN FOR TEXT

The dominant approach for NLP tasks is the RNN, in particular the LSTM [39] and sequence to sequence (Seq2Seq) [40]. The RNN processes sentences in token order, which can learn local features and long-distance dependencies [41]. Compared with the CNN and DNN, the RNN performs better in many NLP tasks such as text recognition [42], machine translation [43] and reading comprehension [44]. However, since the RNN considers the sequence dependency, parallel computing performance of the RNN is poor. Therefore, sequence models usually face the problem of time costs. Some scholars proposed using the CNN to handle NLP tasks such as text classification, because the CNN can execute parallel computing efficiently and reduce the time costs. Santos *et al.* presented a novel CNN architecture named CharSCNN that used from character level to sentence level information to implement sentiment analysis of short texts [45]. They adopted two convolutional layers to extract relevant features from words and sentences. Nal *et al.* presented a convolutional architecture dubbed the DCNN for the semantic modeling of sentences [46]. They handled input sentences of varying lengths and used a feature graph over sentences to capture long-distance dependencies. Alexis *et al.* presented a new architecture (VD-CNN) for text classification [42]. They operated at the lowest atomic representation of text and used only small convolutions and pooling operations to learn the high-level representations of sentences.

III. METHOD

We adopt the embedding method to represent two amino acid sequences as vectors respectively and integrate them together, and then input them into the ResNet to extract the deep features of two proteins. Based on the connected vector, we use the softmax function for classification. Figure 1 shows the framework of the ResPPI algorithm.

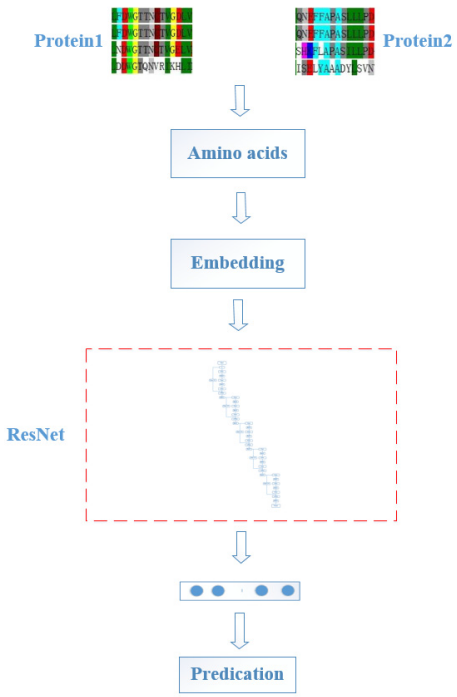


FIGURE 1. The framework of the ResPPI algorithm.

A. AMINO ACID SEQUENCES REPRESENTATION

We define an amino acid sequence as $P = \{a_1, a_2 \ldots a_n\}$, where n represents the length of the amino acid sequence, and $a_i \in P$ represents the amino acid i of the protein. We define that there are m kinds of amino acids in the protein, so a_i has m kinds of different representations. Sequence representation is the basis for protein data analysis and PPI prediction [47]. In current PPI prediction methods, some researchers suggested representing the amino acid sequence as a matrix. For example, Hashemifar *et al.* proposed adopting the position specificity scoring matrix (PSSM) to represent the sequence [1]. For each amino acid of the sequence, they calculated the mutation probability of other amino acids. Therefore, each amino acid could be represented as a $1*(m-1)$ vector, and the amino acid sequence was represented as a

matrix $n^*(m-1)$. However, this method is complicated to calculate, and cannot represent the relation between amino acids. We refer to the embedding representation of words in NLP [48]. We segment the protein sequence into individual words based on amino acids and encode each word as a vector, so the protein sequence can be represented as a matrix. The input and output of the architecture of the embedding layer is the initial matrix of protein sequence and the feature matrix with 500*128. The initial matrix of protein sequence consists of several vectors of amino acids which are random because the training processing of the neural network is updating the parameters and obtaining the feature vector representation of protein sequence. This method can overcome the sparsity problem and perform better in word representation in NLP tasks than the one-hot encoding method. The parameters of the embedding layer and the ResNet are trained simultaneously. We train the model and optimize the loss function, and obtain the vector representation of each amino acid.

For an amino acid sequence P, we assume that there are m different representations of amino acids. We let E represent the matrix of word embedding and H represents the matrix of the encoded sequence which is the input of the embedding and uses the one-hot encoding. So the feature matrix F can define as follows:

$$F = H * E \tag{1}$$

where, H is a matrix of n^*m, and E is a matrix of m^*d. We let d represent the dimension of the vector, so F is a matrix of n^*d. We can see the schematic figure of the sequence representation in Figure 2.

$$\begin{bmatrix} 5 & 1 & \cdots & 2 & 3 \\ 2 & 7 & \cdots & 8 & 4 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 3 & 2 & \cdots & 0 & 0 \\ 4 & 1 & \cdots & 7 & 6 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 1 & \cdots & 0 & 0 \end{bmatrix} = \begin{bmatrix} 5 & 3 & \cdots & 1 & 2 \\ 2 & 4 & \cdots & 7 & 8 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 3 & 0 & \cdots & 2 & 0 \\ 4 & 6 & \cdots & 1 & 7 \end{bmatrix}$$

H [n*m] E [m*d] F [n*d]

FIGURE 2. The schematic figure of amino acid sequence representation.

B. ResNet

The ResPPI algorithm is based on the ResNet, which introduces shortcut connections that bypass the signal from one layer to the next layer [12]. The connections pass through the gradient flows of the DNN and have the capability to ease the training of very deep layers and prevent gradients from disappearing [49]. Inspired by the successes of the ResNet in many challenging tasks such as image recognition and text classification, we propose applying the ReNet to predict PPI.

1) RESIDUAL UNITS

The ResNet consists of many residual units, and each unit includes identity mapping and residuals [50]. Figure 3 shows

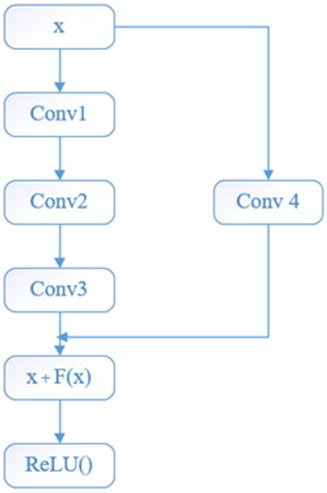


FIGURE 3. The framework of the residual unit.

the framework of the residual unit. It defines as follows:

$$x_{t+1} = h(x_t) + F(x_t) \tag{2}$$

where, x_{t+1} and x_t represent the input and output of the t-th residual unit respectively. $h(x_t)$ is the identity mapping and $F(x_t)$ is the non-linear function.

x_{t+1} and x_t usually have different dimensions in the ResNet, so we will add a convolution layer of 1*1 in the shortcut to reduce the dimensions. So the $h(x_t)$ defines as:

$$h(x_t) = W_t * x_t \tag{3}$$

where, W_t represents the weight matrix.

2) 2D CONVOLUTIONS

Given an encoded amino acid sequence P with the size of n^*d. n and d represent the length of P and the dimensions of the vectors respectively. We utilize the 2D convolutions [45] in the ResNet to extract the features of proteins. From Figure 3, we design three convolution layers in each residual unit, each convolution layer followed by the batch normalization and activation function ReLU [12]. We use pooling to fill in 0 and restore the dimension of x to 64 after the convolution operations in each convolution layer. We denote the size of 2D convolutional filters and the number of convolution kernels of each layer in the residual unit in Table 1.

TABLE 1. The parameters of each residual unit.

Convolution layers	Convolutional filters	Convolution kernels
Conv1	3*3	32
Conv2	3*3	32
Conv3	3*3	64
Conv4	1*1	64

3) ResNet DESIGN

Our proposed ResPPI algorithm can handle protein pairs with different lengths through preprocessing. If the length of a protein sequence is longer than 500, we only select the fragment

with a length of 500 in the sequence. Otherwise, we fill in 0 to make the length of a protein sequence to 500. Based on the distribution of long amino acid sequences, we segment the sequences into several sentences whose length is 500, encoding them as vectors respectively and integrating them together, and then input the vectors to the ResNet. We design five residual units in the ResNet and several residual units introduce shortcut connections. Figure 4 shows the framework of the ResNet.

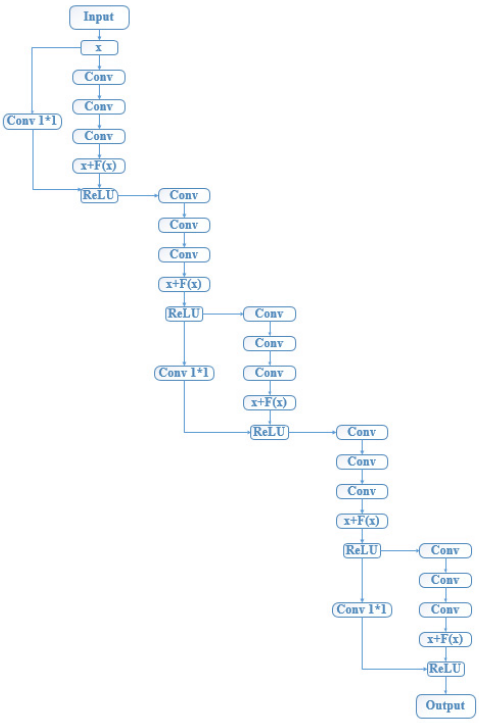


FIGURE 4. The framework of the ResNet.

C. PPI PREDICTION

We obtain the vector h at the full connected layer of the ResNet and h contains the information about two proteins. Then we input h into softmax function to implement binary classification and predict PPI. The equations are as follows:

$$\hat{p}(y \mid s) = softmax\left(W^s h + b^s\right) \tag{4}$$

$$\hat{y} = \text{argmax}\hat{p}(y \mid s) \tag{5}$$

where, $\hat{y}$ and $y \in Y$ are predicted labels and original labels respectively, and s represents the inputted amino acid sequence. W^s and b^s denote weight matrix and bias respectively.

The training process is to minimize the cross-entropy loss function[21] as follows:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} t_i \log(y_i) + \lambda \|\theta\|_F^2 \tag{6}$$

where, $t \in R^m$ is a one-hot vector representing true or false. $y_i \in R^m$ is the probability of predicted category, m is the number of categories. λ is the regularization parameter, used to avoid the overfitting and accelerate the training progress. We use the 5-fold cross-validation method to tune the hyper-parameter λ util the optimal model is obtained.

This is the pseudocode of our algorithm ResPPI as follows:

Algorithm ResPPI

Input: Amino acid sequence a, b and their length n
The dimensions of embedding: d
Procedure:
For each sequence (a, b) do:
Segment the sequence into several words based on amino acids
Use the embedding method to represent the sequence as P, $P \epsilon R^{n*d}$
Integrate the vectors from two sequences into a vector
For each residual unit do:
Perform operations in the convolutional layer Conv1
Perform operations in the convolutional layer Conv2
Perform operations in the convolutional layer Conv3
If (shortcut==true):
Perform operations in the convolutional layer Conv4
Compute the identity mapping with Equation(3)
Add the identity mapping and residuals
Implement classification with Equation(4) and Equation(5)
Output: Predicted result

IV. EXPERIMENT

To quantitatively evaluate the performance of our proposed algorithm ResPPI under different situations, we execute experiments and use six evaluation metrics, compared with several baseline methods.

A. DATASETS

Our model is evaluated on two public datasets for PPI, namely Benchmark and PDB. They both contain positive samples and negative samples.

1) BENCHMARK DATASET

We obtained the popular Pan's PPI dataset in order to use it in this study [51]. The positive samples of this dataset are from the HPRD database.[1] The duplicated interactions are removed from the original samples and there are 36630 pairs as positive samples. For negative samples, they are generated by pairing proteins at different subcellular locations based on the Swiss-Prot database version 57.3. In addition, non-human proteins, proteins from ambiguous or uncertain subcellular locations, and proteins with fragments are removed. Finally, there are 36480 negative samples in Pan's PPI dataset. We remove protein pairs with unusual amino acids to obtain the Benchmark dataset, which has 36545 positive samples and 36323 negative samples.

[1] http://www.hprd.org/.

2) PDB DATASET

The PDB dataset is generated by processing the protein data downloaded from the PDB database.[2] We download 11,680 compound proteins and 2042 monomer proteins respectively. Compound proteins consist of multiple peptide chains, and we select the compound proteins that contain two peptide chains. Then we use Python[3] tool to parse the compound protein files and obtain the interactions of protein sequences as positive samples. For negative samples, we generate them by pairing monomer proteins randomly. In addition, we remove the duplicated interactions and those interactions whose similarities are more than 30%. Finally, we obtain the PDB dataset with 11680 positive samples and 9059 negative samples.

B. EVALUATION METRICS

We use four evaluation metrics which are generally implemented in previous research to evaluate the accuracy of PPI prediction [18]. Besides, we add two evaluation metrics to measure the training time of all methods. These metrics include precision, recall, F1, accuracy, T_e and T_s. They define as follows:

$$precision = \frac{TP}{TP + FP} \tag{7}$$

$$recall = \frac{TP}{TP + FN} \tag{8}$$

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{9}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

$$T_s = \frac{T_e}{N} \tag{11}$$

where, *TP, TN, FP, FN* represent the true positive, true negative, false positive and false negative respectively. T_e and T_s respectively denote the training time of all samples and each sample in each epoch. N is the number of samples.

C. BASELINE METHODS

PPI prediction methods based on deep learning methods and the SVM model performed superior compared to the feature-based models [25]. In order to make an effective comparison of our proposed ResPPI algorithm, we design six baseline methods based on sequence models and other advanced models.

1) RNN

The RNN model first introduces the time series in sequences, considering the positional information between words [52]. We also segment the amino acid sequences into several words based on amino acids and use the embedding method to represent the sequences as vectors. We input the vectors into the neural network to extract the semantic features of proteins and implement binary classification.

[2]http://www.rcsb.org/.

[3]https://www.python.org/.

2) LSTM

The LSTM is the state-of-the-art model to predict PPI based on amino acid sequences and it can be applied to deal with protein data under many situations [53]. The LSTM model uses several components to control the information flow and can overcome the problems of gradient explosion and gradient disappearance, which often appear in deep learning methods. Similar to the RNN, we remove the decoder of the neural network and input the vectors of amino acid sequences into the LSTM model to extract the features of proteins. For the outputted vector from the final hidden layer, we use the softmax function to implement classification.

3) GRU

The GRU is also a sequence model, which is derived from the RNN model as well as the LSTM. Compared with the LSTM model, in the memory unit, it combines the original forget gate and input gate into an update gate to control the information flow [8]. Similar to the RNN and the LSTM, we input the vectors of protein sequences into the GRU model to predict PPI.

4) DCNN

The DCNN can extract the deep features of sequences through convolution operations and non-linear function operations [42]. The DCNN model is often used to predict PPI, and several variants have been derived based on different representations methods of protein sequences such as covariance matrix, PSSM matrix and embedding methods [24]. We use the embedding method to represent the amino acid sequences, and then input them into the DCNN model to extract features and implement classification. There are three convolutional layers in the DCNN model and the size of convolutional filters is 3*3.

5) SVM

The SVM is a popular machine learning method to predict PPI [54], and it is widely used to solve the problem of limited and unbalanced datasets[55]. We use the term frequency-inverse document frequency (TF-IDF) statistical method of word frequency to calculate the probability distribution of each amino acid and obtain the feature matrix of proteins. The TF-IDF method consists of two parts: compute the normalized term-frequency (TF) that the number of times an amino acid appears in a sequence; compute the logarithm of the number of the sequence in the corpus. Then we input the feature matrix to the SVM model to find the optimal hyperplane and implement binary classification.

D. SETTING

We perform experiments using Python tool and Tensorflow[4] 2.0 platform. The main equipment in the experiment includes: Intel (R) Core (TM) i7-8700, 8GB RAM, GPU (GeForce GTX 1060, 5GB RAM). The number of epochs of the ResPPI

[4]https://www.tensorflow.org/.

algorithm is 15 and other parameters such as the size of convolutional filters are shown in Table 1. The learning rate and the size of the batch are set to 0.001 and 32 respectively. For the embedding of amino acids, we set the dimension to 128. For the SVM classifier, the dimension of the feature vector is set to 1000. The penalty coefficient of the objective function is 1.0 and the learning rate is 0.001.

E. RESULT ANALYSIS

To evaluate the performance of our model, we randomly select 18000 and 20000 samples from the Benchmark and PDB datasets respectively, dividing them into five equal subsets. Three subsets are used as a training set; One subset is used as a validation set; The other subset is used as a testing set. In the training phase, we input the vectors from the embedding layer to train the ResNet and optimize the parameters. In the testing phase, we use the trained ResNet model to implement classification and predict the interactions based on testing data, and evaluate the performance of the ResPPI algorithm. The number of epochs of all baseline models is also 15. We can see the average results of precision, recall, F1 and accuracy with the standard deviation of all methods in Table 2 and Table 3. Each task is performed several times and we calculate the mean and standard deviation of all results. Besides, we compare the training time of the ResPPI algorithm and sequence models, the results of T_e and T_s of those methods shown in Figure 5 and Figure 6.

TABLE 2. The results of four metrics in the Benchmark dataset.

Methods	Precision	Recall	F1	Accuracy
ResPPI	**0.9805**	**0.9576**	**0.9689**	**0.9669**
	±0.0169	**±0.0082**	**±0.0507**	**±0.069**
RNN	0.9634	0.9562	0.9598	0.9595
	±0.0104	±0.0175	±0.0044	±0.0039
LSTM	0.9799	0.9398	0.9594	0.9609
	±0.0150	±0.0147	±0.0037	±0.0039
GRU	0.9684	0.9532	0.9607	0.9614
	±0.0028	±0.0139	±0.0063	±0.0052
DCNN	0.9555	0.9165	0.9356	0.9384
	±0.0126	±0.0186	±0.0048	±0.0032
SVM	0.7102	0.7301	0.7200	0.7148
	±0.0232	±0.0477	±0.0316	±0.0244

In order to enable fair comparisons of all algorithms, the results of precision, recall, F1 and accuracy of all algorithms in the Benchmark and PDB dataset are combined. Therefore, the number of variances is equal to 5, and a free degree equals 2. The Friedman test values of precision, recall, F1 and accuracy are 29.390, 39.486, 40.215 and 37.371 respectively. Compared with the overall Friedman test values, it is observed that they are noticeably higher than the critical value, for the significance level $\alpha = 0.05$ (19.296). Hence, the original hypothesis can be safely rejected and there are significant differences among the results of different algorithms

TABLE 3. The results of four metrics in the PDB dataset.

Methods	Precision	Recall	F1	Accuracy
ResPPI	**0.9028**	**0.8852**	**0.8939**	**0.8799**
	±0.0374	**±0.0256**	**±0.0103**	**±0.0088**
RNN	0.8489	0.8824	0.8653	0.8543
	±0.0177	±0.0193	±0.0064	±0.0061
LSTM	0.9014	0.8571	0.8787	0.8717
	±0.0191	±0.0158	±0.0034	±0.0043
GRU	0.8564	0.8832	0.8785	0.8696
	±0.0154	±0.0237	±0.0092	±0.0080
DCNN	0.8803	0.8162	0.8470	0.8449
	±0.0212	±0.0274	±0.0187	±0.0088
SVM	0.7248	0.7853	0.7538	0.7402
	±0.0311	±0.0122	±0.0153	±0.0105

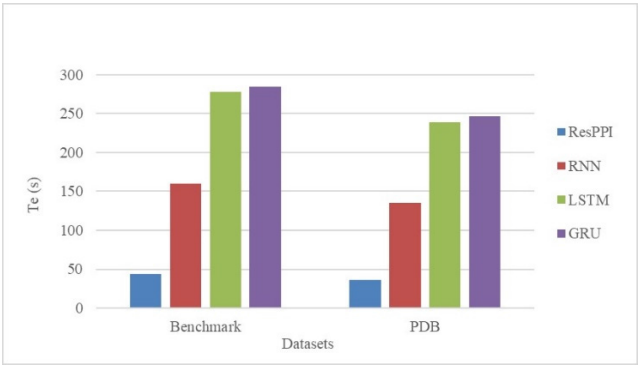


FIGURE 5. The results of T_e in the Benchmark and PDB datasets.

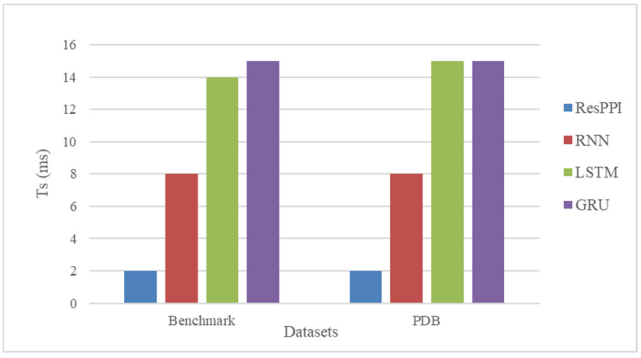


FIGURE 6. The results of T_s in the Benchmark and PDB datasets.

Table 2 and Table 3 show that the results of precision, recall, F1 and accuracy of our proposed ResPPI algorithm are all higher than those of other baseline methods, so the ResPPI performs best to predict PPI, compared with other five methods. In baseline methods, sequence models such as the RNN, LSTM and GRU perform better than the DCNN and the SVM models, because sequence models consider the

positional information of amino acids and can discover more latent features of proteins than the DCNN and machine learning methods. However, sequence models are time-consuming because they cannot take full advantage of GPU. As shown in Figure 5 and Figure 6, we can see that the training time of the RNN, LSTM, GRU models are several times than the ResPPI, so the ResPPI algorithm can achieve rapid PPI prediction. Although the training time of the DCNN and SVM are less than the ResPPI because the DCNN only contains fewer parameters, the accuracy of the DCNN and SVM are less than the ResPPI. We increase the number of convolution layers of the DCNN model to test its performance. However, the results of accuracy cannot improve and the training time becomes longer, so the DCNN model with three layers is the optimized structure to predict PPI. Therefore, our proposed ResPPI algorithm can achieve high accuracy and predict PPI rapidly.

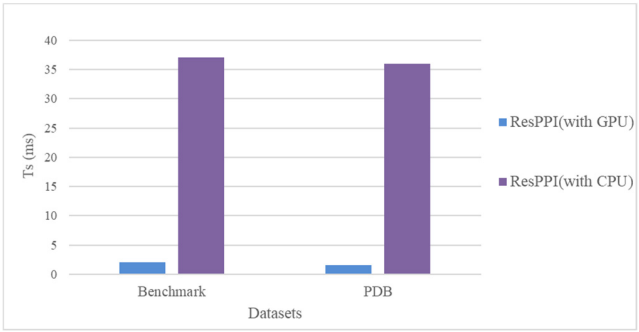


FIGURE 7. The performance of the ResPPI algorithm based on GPU and CPU.

From Figure 7, we can see the results of T_e of the ResPPI algorithm based on GPU and CPU. Compared with CPU, GPU has the capability to make use of parallel computing to accelerate the ResPPI algorithm greatly.

TABLE 4. The results of accuracy of different methods in Benchmark dataset.

Reference	Model	Accuracy
This paper	ResPPI	**0.9669**
Zhang et al.	CS-SVM	0.9400
You et al.	ELM	0.8480

We also compare the ResPPI algorithm against the methods proposed by Zhang *et al.* and You *et al.* based on the Benchmark dataset and reveal the results of accuracy in Table 4. Zhang *et al.* proposed a novel method based on compressed sensing theory to predict PPI from amino acid sequences [56]. They reduce the dimensions of original protein features based on the sparsity of spatial distribution, and then used the SVM classifier to implement classification. You *et al.* proposed a method to predict PPI based on fusion features[51]. They used different methods to represent amino acid sequences and integrate them together, and then trained the DNN to achieve prediction.

Based on the distribution of protein data and the training speed of the ResPPI algorithm, we need to design the appropriate number of residual units and convolution layers to optimize the performance of the ResPPI. In the Benchmark dataset, we adjust the number of residual units and other parameters, and the best results of the ResPPI algorithm with different structures are shown in Table 5. We also segment the dataset into five equal subsets, and select three subsets as a training set, one subset as a validation set and the other subset as a testing set. As the number of residual units increases, the training time of the ResPPI algorithm would become longer. When we design five residual units and 15 convolution layers in the ResNet, the performance of the ResPPI is the best.

TABLE 5. The results of the ResPPI algorithm in Benchmark dataset.

N	Precision	Recall	F1	Accuracy	T_s
3	0.9628	0.9392	0.9509	0.9468	**30s**
4	0.9726	0.9385	0.9552	0.9595	36s
5	0.9788	**0.9521**	**0.9653**	**0.9663**	44s
6	**0.9852**	0.9341	0.9590	0.9617	49s
7	0.9595	0.9278	0.9434	0.9402	54s

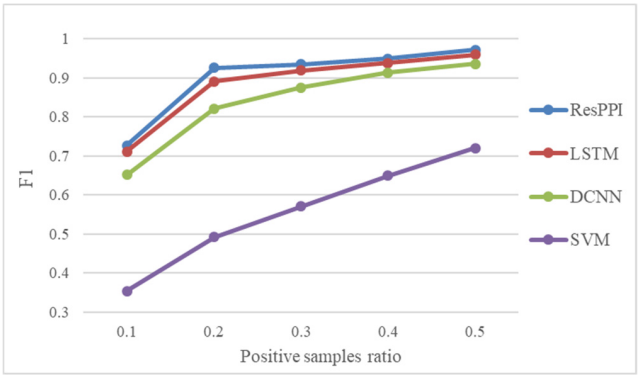


FIGURE 8. The F1 results under different positive samples ratio.

When dealing with biological data, we often have to address the problem of unbalanced datasets. We adjust the positive samples ratio in the Benchmark dataset to test the performance of the ResPPI algorithm, compared with the LSTM, DCNN and SVM methods. Among the sequence models, we choose the LSTM model as the representative because it has the best generalization. Figure 8 shows the F1 results of the ResPPI and three other methods under different positive samples ratio. We can see that our proposed ResPPI algorithm always performs best in different situations. Even when the distribution of positive samples and negative samples is seriously unbalanced, for example, the positive samples ratio is 0.1, the ResNet model still has powerful feature capacities to achieve great effects. Therefore, the ResPPI algorithm has strong generalization

and can provide a reference for the processing of other biological data.

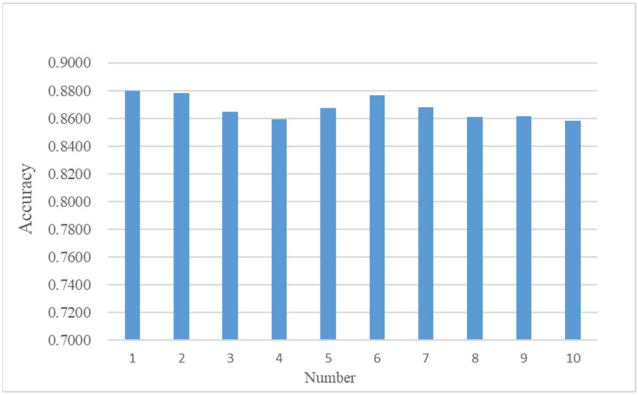


FIGURE 9. The variance of accuracy in the PDB dataset.

In the PDB dataset, we repeat pairing proteins randomly and generating negative samples ten times. And then we use the obtained PDB datasets to evaluate the sensitivity of results. The variance of accuracy is shown in Figure 9, we can see that the results of accuracy based on variance negative samples remain stable between 0.86 and 0.88.

V. CONCLUSION

In this paper, we propose an efficient algorithm to predict PPI based on amino acid sequences. Our algorithm adopts the embedding method to automatically encoder the sequences, using the ResNet with powerful extraction capacities to extract the deep features of proteins and implement classification. Under the ordinary GPU device, our proposed ResPPI algorithm can achieve high accuracy and rapid prediction. The main contribution is that our model need not consider the time series of amino acid sequences like sequence models and can make use of GPU performance to accelerate the PPI prediction. The experimental results show that the ResPPI algorithm is quite successful in PPI prediction, including the accuracy and training speed, compared with other methods in the literature. Our algorithm can provide motivation for new researchers to be done in the field. For future work, we will consider adding the physical properties of amino acids in the ResNet model to improve the accuracy of PPI and optimize the structure of ResNet to accelerate the PPI prediction further.

REFERENCES

[1] S. Hashemifar, B. Neyshabur, A. A. Khan, and J. Xu, "Predicting protein–protein interactions through sequence-based deep learning," *Bioinformatics*, vol. 34, no. 17, pp. i802–i810, Sep. 2018, doi: 10.1093/bioinformatics/bty573.

[2] X. Peng, J. Wang, W. Peng, F.-X. Wu, and Y. Pan, "Protein–protein interactions: Detection, reliability assessment and applications," *Briefings Bioinf.*, vol. 18, no. 5, Jul. 2016, Art. no. bbw066, doi: 10.1093/bib/bbw066.

[3] Y. E. Göktepe and H. Kodaz, "Prediction of protein-protein interactions using an effective sequence based combined method," *Neurocomputing*, vol. 303, pp. 68–74, Aug. 2018, doi: 10.1016/j.neucom.2018.03.062.

[4] T. Sun, B. Zhou, L. Lai, and J. Pei, "Sequence-based prediction of protein protein interaction using a deep-learning algorithm," *BMC Bioinf.*, vol. 18, no. 1, pp. 1–8, Dec. 2017, doi: 10.1186/s12859-017-1700-2.

[5] S. Khurana, R. Rawi, K. Kunji, G.-Y. Chuang, H. Bensmail, and R. Mall, "DeepSol: A deep learning framework for sequence-based protein solubility prediction," *Bioinformatics*, vol. 34, no. 15, pp. 2605–2613, Aug. 2018, doi: 10.1093/bioinformatics/bty166.

[6] X. Du, S. Sun, C. Hu, X. Li, and J. Xia, "Prediction of protein–protein interaction sites by means of ensemble learning and weighted feature descriptor," *J. Biol. Res.-Thessaloniki*, vol. 23, no. S1, pp. 23–28, May 2016, doi: 10.1186/s40709-016-0046-7.

[7] B. Zhang, J. Li, L. Quan, Y. Chen, and Q. Lü, "Sequence-based prediction of protein-protein interaction sites by simplified long short-term memory network," *Neurocomputing*, vol. 357, pp. 86–100, Sep. 2019, doi: 10.1016/j.neucom.2019.05.013.

[8] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," Dec. 2014, *arXiv:1412.3555*. Accessed: Feb. 13, 2020. [Online]. Available: http://arxiv.org/abs/1412.3555

[9] X. Liu, B. Liu, Z. Huang, T. Shi, Y. Chen, and J. Zhang, "SPPS: A sequence-based method for predicting probability of protein-protein interaction partners," *PLoS ONE*, vol. 7, no. 1, Jan. 2012, Art. no. e30938, doi: 10.1371/journal.pone.0030938.

[10] D. Manatunga, D. H. Kim, and D. S. Mukhopadhyay, "SP-CNN: A scalable and programmable CNN-based accelerator," *IEEE Micro*, vol. 35, no. 5, pp. 42–50, Sep./Oct. 2015.

[11] N. Strodthoff, P. Wagner, M. Wenzel, and W. Samek, "UDSMProt: Universal deep sequence models for protein classification," *Bioinformatics*, vol. 36, no. 8, pp. 2401–2409, Apr. 2020, doi: 10.1093/bioinformatics/btaa003.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Dec. 2015, *arXiv:1512.03385*. Accessed: Feb. 6, 2020. [Online]. Available: http://arxiv.org/abs/1512.03385

[13] H. A. Burkhardt, D. Subramanian, J. Mower, and T. Cohen, "Predicting adverse drug-drug interactions with neural embedding of semantic predications," *Bioinformatics*, vol. 10, pp. 1–10, Sep. 2019, doi: 10.1101/752022.

[14] Z.-P. Liu and H. Miao, "Prediction of protein-RNA interactions using sequence and structure descriptors," *Neurocomputing*, vol. 206, pp. 28–34, Sep. 2016, doi: 10.1016/j.neucom.2015.11.105.

[15] B. Liu, "BioSeq-analysis: A platform for DNA, RNA and protein sequence analysis based on machine learning approaches," *Briefings Bioinf.*, vol. 20, no. 4, pp. 1280–1294, Jul. 2019, doi: 10.1093/bib/bbx165.

[16] M. Kulmanov, M. A. Khan, and R. Hoehndorf, "DeepGO: Predicting protein functions from sequence and interactions using a deep ontology-aware classifier," *Bioinformatics*, vol. 34, no. 4, pp. 660–668, Feb. 2018, doi: 10.1093/bioinformatics/btx624.

[17] J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, and H. Jiang, "Predicting protein-protein interactions based only on sequences information," *Proc. Nat. Acad. Sci. USA*, vol. 104, no. 11, pp. 4337–4341, Mar. 2007, doi: 10.1073/pnas.0607879104.

[18] Y. E. Göktepe and H. Kodaz, "Prediction of protein-protein interactions using an effective sequence based combined method," *Neurocomputing*, vol. 303, pp. 68–74, Aug. 2018, doi: 10.1016/j.neucom.2018.03.062.

[19] Z.-H. You, X. Li, and K. C. Chan, "An improved sequence-based prediction protocol for protein-protein interactions using amino acids substitution matrix and rotation forest ensemble classifiers," *Neurocomputing*, vol. 228, pp. 277–282, Mar. 2017, doi: 10.1016/j.neucom.2016.10.042.

[20] S. Lu, B. Wang, H. Wang, L. Chen, M. Linjian, and X. Zhang, "A real-time object detection algorithm for video," *Comput. Electr. Eng.*, vol. 77, pp. 398–408, Jul. 2019, doi: 10.1016/j.compeleceng.2019.05.009.

[21] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling," Nov. 2016, *arXiv:1611.06639*. Accessed: Feb. 06, 2020. [Online]. Available: http://arxiv.org/abs/1611.06639

[22] J. Jiménez, S. Doerr, G. Martínez-Rosell, A. S. Rose, and G. De Fabritiis, "DeepSite: Protein-binding site predictor using 3D-convolutional neural networks," *Bioinformatics*, vol. 33, no. 19, pp. 3036–3042, Oct. 2017, doi: 10.1093/bioinformatics/btx350.

[23] M. Torrisi, G. Pollastri, and Q. Le, "Deep learning methods in protein structure prediction," *Comput. Structural Biotechnol. J.*, vol. 18, pp. 1301–1310, Jan. 2020, doi: 10.1016/j.csbj.2019.12.011.

[24] X. Du, S. Sun, C. Hu, Y. Yao, Y. Yan, and Y. Zhang, "DeepPPI: Boosting prediction of protein–protein interactions with deep neural networks," *J. Chem. Inf. Model.*, vol. 57, no. 6, pp. 1499–1510, Jun. 2017, doi: 10.1021/acs.jcim.7b00028.

[25] L. Zhang, G. Yu, D. Xia, and J. Wang, "Protein–protein interactions prediction based on ensemble deep neural networks," *Neurocomputing*, vol. 324, pp. 10–19, Jan. 2019, doi: 10.1016/j.neucom.2018.02.097.

[26] S. Yadav, A. Kumar, A. Ekbal, S. Saha, and P. Bhattacharyya, "Feature assisted bi-directional LSTM model for protein-protein interaction identification from biomedical texts," Jul. 2018, *arXiv:1807.02162*. Accessed: Mar. 30, 2020. [Online]. Available: http://arxiv.org/abs/1807.02162

[27] S. Yadav, A. Ekbal, S. Saha, A. Kumar, and P. Bhattacharyya, "Feature assisted stacked attentive shortest dependency path based bi-LSTM model for protein–protein interaction," *Knowl.-Based Syst.*, vol. 166, pp. 18–29, Feb. 2019, doi: 10.1016/j.knosys.2018.11.020.

[28] M. Ahmed, J. Islam, M. R. Samee, and R. E. Mercer, "Identifying protein-protein interaction using tree LSTM and structured attention," Jul. 2018, *arXiv:1808.03227*. Accessed: Apr. 12, 2020. [Online]. Available: http://arxiv.org/abs/1808.03227

[29] G. Pagés, B. Charmettant, and S. Grudinin, "Protein model quality assessment using 3D oriented convolutional neural networks," *Bioinformatics*, vol. 35, no. 18, pp. 3313–3319, Sep. 2019, doi: 10.1093/bioinformatics/btz122.

[30] L. Hua and C. Quan, "A shortest dependency path based convolutional neural network for protein-protein relation extraction," *BioMed Res. Int.*, vol. 2016, pp. 1–9, Jul. 2016, doi: 10.1155/2016/8479587.

[31] Y. Peng and Z. Lu, "Deep learning for extracting protein-protein interactions from biomedical literature," in *Proc. BioNLP*, Vancouver, BC, Canada, 2017, pp. 29–38, doi: 10.18653/v1/W17-2304.

[32] S. P. Dubey, N. G. Kini, M. S. Kumar, and S. Balaji, "*Ab initio* protein structure prediction using GPU computing," *Perspect. Sci.*, vol. 8, pp. 645–647, Sep. 2016, doi: 10.1016/j.pisc.2016.06.046.

[33] S. Christley, B. Lee, X. Dai, and Q. Nie, "Integrative multicellular biological modeling: A case study of 3D epidermal development using GPU algorithms," *BMC Syst. Biol.*, vol. 4, no. 1, p. 107, Dec. 2010, doi: 10.1186/1752-0509-4-107.

[34] X. Sun, X. Ren, S. Ma, and H. Wang, "meProp: Sparsified back propagation for accelerated deep learning with reduced overfitting," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, Aug. 2017, pp. 3299–3308.

[35] L. N. Huynh, Y. Lee, and R. K. Balan, "DeepMon: Mobile GPU-based deep learning framework for continuous vision applications," in *Proc. 15th Annu. Int. Conf. Mobile Syst., Appl., Services*, Niagara Falls, NY, USA, Jun. 2017, pp. 82–95, doi: 10.1145/3081333.3081360.

[36] M. Song, Y. Hu, Y. Xu, C. Li, H. Chen, J. Yuan, and T. Li, "Bridging the semantic gaps of GPU acceleration for scale-out CNN-based big data processing: Think big, see small," in *Proc. Int. Conf. Parallel Archit. Compilation*, Haifa, Israel, Sep. 2016, pp. 315–326, doi: 10.1145/2967938.2967944.

[37] C. Wu and A. Kalyanaraman, "GPU-accelerated protein family identification for metagenomics," in *Proc. IEEE Int. Symp. Parallel Distrib. Process., Workshops Phd Forum*, Cambridge, MA, USA, May 2013, pp. 559–568, doi: 10.1109/IPDPSW.2013.185.

[38] S. Pang, R. J. Stones, M.-M. Ren, X.-G. Liu, G. Wang, H.-J. Xia, H.-Y. Wu, Y. Liu, and Q. Xie, "GPU MrBayes V3.1: MrBayes on graphics processing units for protein sequence data: Table 1," *Mol. Biol. Evol.*, vol. 32, no. 9, pp. 2496–2497, Sep. 2015, doi: 10.1093/molbev/msv129.

[39] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," Aug. 2015, *arXiv:1508.01991*. Accessed: Feb. 12, 2020. [Online]. Available: http://arxiv.org/abs/1508.01991

[40] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," Dec. 2014, *arXiv:1409.3215*. Accessed: Feb. 6, 2020. [Online]. Available: http://arxiv.org/abs/1409.3215

[41] A. Graves, "Generating sequences with recurrent neural networks," Jun. 2013, *arXiv:1308.0850*. Accessed: Feb. 6, 2020. [Online]. Available: http://arxiv.org/abs/1308.0850

[42] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," Jun. 2016, *arXiv:1606.01781*. Accessed: Feb. 6, 2020. [Online]. Available: http://arxiv.org/abs/1606.01781

[43] J. Zhang and C. Zong, "Deep neural networks in machine translation: An overview," *IEEE Intell. Syst.*, vol. 30, no. 5, pp. 16–25, Sep. 2015, doi: 10.1109/MIS.2015.69.

[44] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, Apr. 2016, doi: 10.1016/j.neucom.2015.09.116.

[45] C. Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proc. COLING*, Ireland, Aug. 2014, pp. 69–78.

[46] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," Apr. 2014, *arXiv:1404.2188*. Accessed: Apr. 12, 2020. [Online]. Available: http://arxiv.org/abs/1404.2188

[47] Z.-H. You, L. Zhu, C.-H. Zheng, H.-J. Yu, S.-P. Deng, and Z. Ji, "Prediction of protein-protein interactions from amino acid sequences using a novel multi-scale continuous and discontinuous feature set," *BMC Bioinf.*, vol. 15, no. 15, p. S9, 2014, doi: 10.1186/1471-2105-15-S15-S9.

[48] A. Nikfarjam, A. Sarker, K. O'Connor, R. Ginn, and G. Gonzalez, "Pharmacovigilance from social media: Mining adverse drug reaction mentions using sequence labeling with word embedding cluster features," *J. Amer. Med. Inform. Assoc.*, vol. 22, no. 3, pp. 671–681, Mar. 2015, doi: 10.1093/jamia/ocu041.

[49] Z. Wu, C. Shen, and A. van den Hengel, "Wider or deeper: Revisiting the ResNet model for visual recognition," *Pattern Recognit.*, vol. 90, pp. 119–133, Jun. 2019, doi: 10.1016/j.patcog.2019.01.006.

[50] K. Hara, H. Kataoka, and Y. Satoh, "Learning spatio-temporal features with 3D residual networks for action recognition," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Venice, Italy, Oct. 2017, pp. 3154–3160, doi: 10.1109/ICCVW.2017.373.

[51] Z.-H. You, S. Li, X. Gao, X. Luo, and Z. Ji, "Large-scale protein-protein interactions detection by integrating big biosensing data with computational model," *BioMed Res. Int.*, vol. 2014, pp. 1–9, Aug. 2014, doi: 10.1155/2014/598129.

[52] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," Oct. 2015, *arXiv:1506.00019*. Accessed: Feb. 13, 2020. [Online]. Available: http://arxiv.org/abs/1506.00019

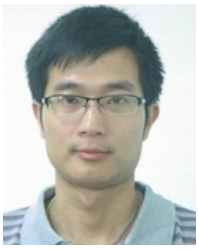
[53] J. Hanson, Y. Yang, K. Paliwal, and Y. Zhou, "Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks," *Bioinformatics*, vol. 33, no. 5, Mar. 2017, Art. no. btw678, doi: 10.1093/bioinformatics/btw678.

[54] A. Koike and T. Takagi, "Prediction of protein-protein interaction sites using support vector machines," *Protein Eng. Design Selection*, vol. 17, no. 2, pp. 165–173, Jan. 2004, doi: 10.1093/protein/gzh020.

[55] T. Ebina, H. Toh, and Y. Kuroda, "DROP: An SVM domain linker predictor trained with optimal features selected by random forest," *Bioinformatics*, vol. 27, no. 4, pp. 487–494, Feb. 2011, doi: 10.1093/bioinformatics/btq700.

[56] Y.-N. Zhang, X.-Y. Pan, Y. Huang, and H.-B. Shen, "Adaptive compressive learning for prediction of protein–protein interactions from primary sequence," *J. Theor. Biol.*, vol. 283, no. 1, pp. 44–52, Aug. 2011, doi: 10.1016/j.jtbi.2011.05.023.

SHENGYU LU was born in Ganzhou, Jiangxi, China, in 1995. He received the B.S. degree from the Software School, Nanchang University, China, in 2017. He is currently pursuing the M.S. degree with the School of Informatics, Xiamen University, China. He was a Visiting Researcher at Shanghai Jiao Tong University. His research interests include computer vision, bioinformatics, and machine learning.



QINGQI HONG received the Ph.D. degree in computer science from the University of Hull, U.K. He is currently an Associate Professor with the School of Informatics, Xiamen University, China. His current research interests include medical imaging processing, 3D visualization, 3D modeling, computer-aided diagnosis and surgery, deep learning, and GPU computing.

BEIZHAN WANG was born in Xianyang, Shaanxi, China, in 1965. He received the B.S., M.S., and Ph.D. degrees from the School of Computer Science, Northwestern Polytechnical University, China. Since 2004, he has been a Professor with the School of Informatics, Xiamen University, China. His research interests include computer vision and data mining.

HONGJI WANG received the Ph.D. degree in computer software and theory from the Chinese Academy of Sciences, China. He was a Visiting Researcher at Purdue University. Since 2008, he has been an Associate Professor with the School of Informatics, Xiamen University, China. His research interests include data mining and information security.