Sequence-based prediction of protein-protein interaction sites by simplified long-short term memory network

Buzhong Zhang[a,b], Jinyan Li[c], Lijun Quan[a,d], Yu Chen[a], Qiang Lü[a,d,*]

[a]*School of Computer Science and Technology, Soochow University, Suzhou 215006, China*
[b]*School of Computer and Information, and the University Key Laboratory of Intelligent Perception and Computing of Anhui Province, Anqing Normal University, Anqing 246011, China*
[c]*Advanced Analytics Institute, University of Technology Sydney, PO Box 123, Broadway, NSW 2007, Australia*
[d]*Jiangsu Provincial Key Laboratory for Information Processing Technologies, Soochow University, Suzhou 215006, China*

Abstract

Proteins often interact with each other and form protein complexes to carry out various biochemical activities. Knowledge the interaction sites of are helpful for understanding disease mechanisms and drug design. Accurate prediction of the interaction sites from protein sequences is still a challenging task and severe imbalance data also decreased the performance of computational methods. In this study, we propose to use a deep learning method for improving the imbalanced prediction of protein interaction sites. We develop a new simplified long-short term memory (SLSTM) network to implement a deep learning architecture (named DLPred). To deal with the imbalanced classification in the deep learning model, we explore three new ideas. First, our collection of the training data is to construct a set of protein sequences, instead of a set of just single residues, to retain the entire sequential completeness of each protein. Second, a new penalization factor is appended to the loss function such that the penalization to the non-interaction site loss can be effectively enhanced. Third, multi-task learning of interaction sites and residue solvent accessibility prediction are used for correcting the preference of the prediction model on the non-interaction sites. Our model is evaluated on three public datasets: Dset186, Dtestset72 and PDBtestset164. Compared with current state-of-the-art methods, DLPred is able to significantly improve the predictive accuracies and AUC values while improving the F-measure. The training dataset, test datasets, a standalone version of DLPred and online service are available at http://qianglab.scst.suda.edu.cn/dlp/.

Keywords: Protein-protein interaction sites, Imbalance classification, Deep learning, Simplified

*Corresponding author
Email address: qiang@suda.edu.cn (Qiang Lü)

long-short term memory network

1. Introduction

Protein-protein interactions are fundamental for many cellular biological processes, such as signal transduction, immune response, and cellular organization [1]. The protein-protein interaction sites (PPISs) are composed of a set of amino acid residues that form chemical bonds with a part of another molecule. Detection of interaction domains in sequences is very useful for understanding mechanisms of various biological processes, disease development and drug designs. Experimentally determined protein 3D structures indeed provide important clues to identifying interaction sites and understanding protein functions [2]. However, biological experimental methods [3] are labor-intensive and time-consuming, and the number of known 3D structures is still considerably smaller than that of protein sequences.

Over the decades, researchers have investigated the possibility of utilizing computational approaches to rapidly and accurately predict interacting residues from protein sequences. Jones and Thornton's research [4] reported that solvation potential, residue interface propensity, hydrophobicity, planarity, protrusion and accessible surface area are the most important features to differentiate an observed interface from others defined on the surface of a protein. Neuvirth [2] suggested that locations of protein-protein binding sites are imprinted in the structures of the proteins. Ofran and Rost also concluded [5] that unbound proteins could suffice for the identification of interface residues.

Hitherto many computational methods have been proposed to deal with this prediction problem, including artificial neural networks [1, 6, 7, 8], support vector machines (SVMs) [9, 7, 10], random forests [11, 12], Naïve Bayes classifier [13], L1-regularized logistic regression [14], ensembles of SVM and sample-weighted random forests [15]. In particular, Zhou and Shan [1] proposed a neural network prediction with sequence profiles of neighboring residues and solvent exposure as input. Ofran and Rost [5] proposed another neural network predictor (ISIS), which was trained on sequences profiles and structural features predicted from the sequences. Porollo and Meller [7] proposed a method named SPPIDER using an SVM, neural network and linear discriminant analysis based on 19 selected features from the sequences. Murakami and Mizuguchi [13] developed a predictor called PSIVER, which is Naïve Bayes classifier with a kernel density estimation based on position-specific scoring matrix (PSSM) and predicted solvent accessibility. Dhole, Singh et al [14]

proposed a L1-regularized logistic regression classifier named LORIS. Furthermore, Singh and D-hole [8] proposed a novel artificial neural network predictor SPRINGS. Both SPRINGS and LORIS are trained on the feature space of PSSM, averaged cumulative hydropathy and predicted relative solvent accessibility.

Although much progress has been made, there still has room for further improving the performance of PPIS prediction. And one of the challenging issues in this research is class imbalance. Recently, Some methods have dedicated effort to solve the problem. Wei et al[16] firstly concerned the problem and a cascade random forests algorithm(CRF) is proposed. The proposed CRF connects multiple random forests in a cascade-like manner, each of which is trained with a balanced training subset that includes all minority samples and a subset of majority samples. However, sampling of training data based-on residues level destroys the completeness of a sequence. Another method, SSWRF [15] combines an ensemble of SVMs and sample-weighted random forests to cope with the class imbalance issue, but its prediction accuracy is not very appealing.

In this work, we explore new ideas to address the imbalance issue and design a proper deep learning architecture such that the model has more generalization on the imbalanced data.

Firstly, A lightweight variant of long short-term memory (LSTM) [17], named Simplified Long-short Term Memory (SLSTM) network, is proposed and taken as the fundamental module in our model architecture. Our deep learning model (named DLPred) is stacked by a three-layer SLSTM linked with two layers of forward neural networks. Compared with the models using LSTM or gated recurrent units(GRU) [18], parameters of SLSTM-based model are just only 61.4% of LSTM-based model, or 81.7% of GRU-based model. The training speed of SLSTM-based model is faster than GRU-based or LSTM-based model, but the performance of DLPred model based on SLSTM is comparable to that of GRU_based model and better than that of LSTM_based model.

The training data is filtered on sequence level. Specific approaches to the construction of training data have been well investigated in the literature [19, 20] to handle the imbalance issue. The most straight-forward approaches are various techniques of adjusting training data. Traditionally the collection of training data is to form a set of individual residues. If we adjust the training data based-on residues level, Such an approach will shatter the sequential completeness of many proteins. In this work, the collection of training data is to form a set of complete protein sequences. Thus, each sequence in the training dataset still contains its complete set of binding residues and its complete set of non-binding residues. Our training dataset (TR5860) comprised of 5860 sequences

is collected from multiple data sources, where each sequence has at least 10% of the interacting residues over the whole sequence [21].

Inspired by the recent successes of cost-sensitive learning in convolutional neural networks (CNNs) [22], we append a new penalization factor in the loss function so that the penalization on the mis-classed non-interacting residues is enhanced to cope with the imbalance issue.

Finally, multi-task learning is used to correct the preference of the prediction model for the non-interacting residues. The interacting residues are closely correlated with residue solvent accessibility (RSA) in our feature space construction. Most of the interacting residues are interface residues of the protein. Only residues with more solvent accessible area have higher potential to become interface residues. We propose to concurrently predict PPISs and RSA, which is an effective approach to improve our model generalization of balanced classification.

In this study, we incorporate sequence-derived features such as the PSSM, physical properties, hydropathy index, etc. in the DLPred model. DLPred is evaluated on three public PPISs test datasets Dset186, Dtestset72 and PDBtestset164. Experimental results show that our model has improved F-measures, predictive accuracies and AUC values. We achieved 38.9%, 69.1% and 80.1% in F-measure, accuracy and AUC respectively on Dset186; we achieved 42.6%, 69% and 81.1% in F-measure, accuracy and AUC respectively on Dtestset72; and we achieved 38.8%, 68.4% and 78.9% in F-measure, accuracy and AUC respectively on PDBtestset164. Compared with other predictors, DLPred is simple but more generalizable and improved the performance of balanced classification.

2. Materials and methodology

In this section, the proposed method of protein-protein interaction sites prediction is explained in detail.

2.1. Training and test datasets of protein sequences

The imbalance degree [21] of a class distribution can be denoted by the ratio of the sample size of the small class to that of the prevalent class. However, there is a modest class imbalance like 1:10 [21] that can cause imbalance problems. Studies [23, 21, 19] indicate that a relatively balanced distribution usually attains a better result. Meanwhile, considering the completeness of a sequence, we select training data based on sequence level and the sequence whose proportion of interacting residues is less than 10% is discarded.

Protein sequences used for training the DLPred model are obtained from three CullPDB datasets[24](containing 9494 sequences, 12665 sequences, and 13707 sequences, respectively) and cons-PPISP's [6] training dataset (1256 sequences). The sequences whose similarity is more than 40% with those in testing datasets are removed from the candidate training dataset by CD-hit [25] software.

Then a training dataset, TR5860 is developed and it has 5860 filtered sequences. TR5860 has 332327 interacting residues, and 1154052 residues in total. Of these sequences, 4210 sequences of our training set are collected from three CullPDB datasets under the parameters of similarity less than 25%, resolution better than 3.0Å, R factor of 1.0 and sequence length between 50 and 600. Next, The other 781 sequences in the training set are re-selected from the 4210 sequences because the minimum 40% ratio of interacting residues can improve the class balance for better prediction results as suggested by Galar et al. [23], Sun et al. [21] and, He and Garcia [19]. Finally, 959 (denoted TR959) are collected from the training dataset of cons-PPISP [6]. The sequence length of all of these proteins is at least 50. An independent validating dataset denoted by VD141 which has 141 protein sequences, is randomly selected from TR959. Other 5719 sequences in TR5860 are used to train the model.

Three public datasets, Dset186, Dtestset72 and PDBtestset164 are used to test the generalization performance of our deep learning model. Dset186 and Dtestset72 are benchmark datasets previously explored by [13]. Dset186 contains 186 protein sequences extracted from 105 heterodimeric protein complexes with a sequence identity<25% and a resolution of $\leq$3.0 Å. Dset186 has a total of 36216 residues (including 5551 interacting residues). This dataset has been widely used by[13, 15, 26] to train learning methods. However, we use it here to test the generalization performance of our method. Dtestset72 contains 72 protein sequences from 36 protein complexes in the protein-protein docking benchmark set version 3.0 [27]. These sequences having $\geq$25% sequence identity over a 90% overlap with any of the sequences in Dset186 are removed. Dtestset72 contains 17975 residues in total with 3799 interacting residues. PDBtestset164 consists of 164 non-redundant protein sequences released by [14] with the same filtering requirement as for Dset186. There are 6111 interacting residues and a total of 33678 residues in PDBtestset164.

In these training and test datasets, a residue is defined as an interacting residue if its loss of the absolute solvent accessibility is at least 1.0 $Å^2$ on complex formation. In our experiments, all PPISs were identified by the software PSAIA [28]. Residue accessible areas were computed by DSSP software [29]. It is defined as the ratio obtained by dividing the solvent accessible surface

area by the maximum solvent accessibility [30], where Gly-X-Gly extended tripeptides are used in the calculation. If the ratio of a residue accessible area is greater than or equal to the threshold of 25%, the residue is classified as exposed (positive class); otherwise the residue is classified as buried (negative class).

2.2. Features of a residue derived from protein sequence

Each residue in a protein sequence is represented by a vector of eight groups of sequence-derived features: a 20-dimensional Position-specific scoring matrix(PSSM), 7-dimensional physical properties, a 1-dimensional hydropathy index, 3-dimensional physicochemical characteristics, a 1-dimensional PKx, 18-dimensional 3D-1D scores, a 1-dimensional conservation score, and 22-dimensional protein sequence coding.

Position-specific scoring matrix. The PSSM describes the evolutionary conservation of the residue positions. The PSSM was obtained by performing multiple sequence alignments on a large protein database (NCBI NR database) using PSI-BLAST [31] with an expectation value (E-value) threshold of 0.001, for three iterations against the BLAST non-redundant protein sequence database. The PSSM profile is in the form of a $20 \times L$ matrix, where L is the length and each amino acid in the sequence is described by 20 features.

Physical properties. The seven dimensional physical properties [32] are as follows: a steric parameter (graph-shape index), polarizability, volume (normalized van der Waals volume), hydrophobicity, isoelectric point, helix probability and sheet probability.

Hydropathy index. Hydrophobicity scales (hydropathy index), composed of experimentally determined transfer free energies for each amino acid, are essential for understanding the energetics of protein-bilayer interactions [33]. The specific values are from [34].

Physicochemical characteristics. Protein physicochemical characteristics [35] include the number of atoms, electrostatic charges and potential hydrogen bonds.

PKx. PKx is the negative of the logarithm of the dissociation constant for any other group in the molecule [36].

3D-1D scores. The side-chain environment was first proposed by Bowie et al.[37] and used in the 3D-profile structural prediction method. The 3D structure profile connects the 3D structure and 1D sequence by specifying the 3D-1D scores. Fan et al.[38] utilized it for the prediction of protein solvent accessibility. Details of the classification of side-chain environments and 3D-1D scores are described in their article[37].

The described features are normalized using a logistic regression function $y = 1/(1 + e^{-x})$.

Conservation score. Residue conservation is derived from the amino acid frequency distribution in the corresponding column of a multiple-sequence alignment of homologous proteins. A 1-dimensional conservation score is computed by the method developed by [39].

Protein sequence coding. As proposed by Zhang [40], a sparse one-hot vector which is used to represent a residue, is mapped into a denser 22-dimensional vector.

In order to train the model on a GPU in batches, proteins shorter than 600 AA are padded with all-zero features and sequences with more than 600 length are broken into two sequences. The outputs corresponding to the padded inputs are labeled as "None".

2.3. Simplified LSTM

2.3.1. LSTM

The recurrent neural network (RNN) is an extremely powerful model for sequence modeling tasks [41] by learning both local features and long range dependencies from sequential data. However, training the RNN can become problematic especially when the exploding or vanishing gradient problem occurs [42]. Gated recurrent neural network, such as LSTM [17] or GRU [18] is a very successful attempt to resolve the problem. Gated recurrent neural networks use gate components to control the information flow to alleviate the vanishing and exploding gradient problems. The vector formulas of vanilla LSTM given by [43] can be written as follows:

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + U_i h_{t-1} + V_i \bigodot c_{t-1} + b_i) \\
f_t &= \sigma(W_f x_t + U_f h_{t-1} + V_f \bigodot c_{t-1} + b_f) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + V_o \bigodot c_{t-1} + b_o) \\
c_t &= f_t \bigodot c_{t-1} + i_t \bigodot \tanh(W_c x_c + U_c h_{t-1} + b_c) \\
h_t &= o_t \bigodot \tanh(c_t)
\end{aligned}
\tag{1}
$$

where x_t and h_t are the input and output vectors at time t respectively; σ is the sigmoid function; i, f, o, and c are respectively the input gate, forget gate, output gate and cell activation vectors, all of which are the same size as the hidden vector h. Furthermore, V_i, V_f and V_o are peephole connecting weights. $\odot$ is denoted as element-wise multiplication of two vectors.

2.3.2. SLSTM

The core contribution of LSTM is that the gradient can flow for long durations [44]. The LSTM network uses memory cell to learn short-term and long-term dependencies. A system of gating units controls the information flow. The LSTM has been found to be extremely successful in sequential data processing. However, parameters of LSTM are very large. To train the LSTM-based model sufficiently, more training data and computing resources should be provided. However, in practice, it is difficult to obtain more training data in the PPISs prediction problem. So simplifying the architecture of LSTM and improving computing performance can reduce the number of parameters and the computational cost. Additionally, a lightweight variant of LSTM is helpful for building deep learning model to predict PPISs. We need to find which components may not be needed for good results.

Recently, many variants of LSTM are proposed,such as GRU, QRNN[49] and SRU [48]. And Greff et al [45] systematically studied the utility of various computational components. As Greff's study, two variants are attractive: removing peephole connections is not significantly decreasing performance; and coupling the input and forget gates only slightly impair the performance.

Since memory cell and controlling gates are important components in RNN, many researches try to redefined the RNN's architecture. Mikolov et al. [46] proposed a structurally constrained recurrent network

$$s_t = \alpha \cdot s_{t-1} + (1 - \alpha) \cdot W_s x_t. \tag{2}$$

Balduzzi and Ghifary [47] developed the structurally constrained recurrent network into a new strongly-typed recurrent neural network.

On the above-mentioned works, to process strongly-sequential data such as protein sequences and to speed up the method, we propose a novel simplified LSTM (SLSTM) network. The architecture

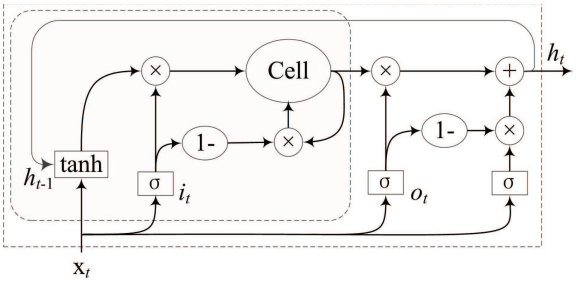

Figure 1: SLSTM architecture.

of SLSTM is illustrated in Fig.1 and equation is described as follows:

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + b_i) \\
o_t &= \sigma(W_o x_t + b_o) \\
X_t &= \sigma(W_x x_t + b_x) \\
z_t &= tanh(W_z x_t + U_z h_{t-1} + b_z) \\
c_t &= (1 - i_t) \bigodot c_{t-1} + i_t \bigodot z_t \\
h_t &= o_t \bigodot c_t + (1 - o_t) \bigodot X_t
\end{aligned}
\tag{3}
$$

We note that in an SLSTM, the previous state h_{t-1} in the input and output gate is removed, and the forget and input gate is coupled. Similar to GRU and SRU , the current state output h_t is computed through the combination of the cell output c_t and the weighting input X_t. A single output gate o_t is used to control the cell state and the weighting input. Our h_t computation is in accordance with the strongly-typed recurrent neural network, and therefore the non-linear activity function in the output equation can be removed.

The non-linear activity function and the previous state h_{t-1} are also preserved at the memory cell input. The speedup variants of LSTM such as QRNN and SRU are excluded for all of the previous state dependencies h_{t-1}. Nevertheless, removing all of the previous state dependencies h_{t-1} is inconvenient for the current state to remember the previous information. Experiments of SRU [48] on SQuAD have demonstrated that the performance of bidirectional SRU is inferior to that of bidirectional LSTM under the same setting of three layers and 128 dimensions. Since protein sequences are strongly context-dependent, we need to retain h_{t-1}.

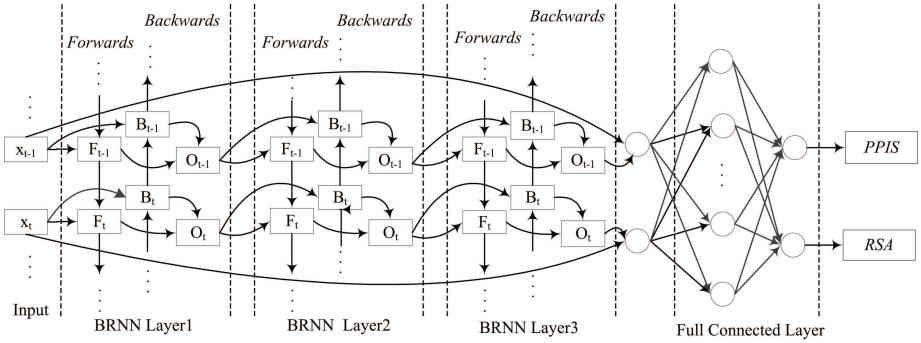

Figure 2: DLPred architecture. There are three stacked BRNNs using BSLSTM layers in DLPred. Two fully connected layers are linked with the last BSLSTM layer. In the output layer, RSA and PPIS are concurrently predicted by DLPred.

2.4. Implementation of DLPred

2.4.1. Architecture of DLPred

Protein structures and functions are affected largely by long-range interactions between residues. We employ Bidirectional Recurrent Neural Networks(BRNNs) [50] to capture long-distance dependencies between amino acids. As shown in Figure 2, Our proposed model is very concise. DLPred uses three stacked bidirectional simplified LSTM (BSLSTM) layers, followed by two fully connected layers. When the forward output F_t is merged with the backward output B_t, merging computation in the first BRNN layer is concatenated, and computations in other BRNN layers are summed. Similar to residual network [51], the input information are short-cut connected to the first fully connected layer, which can improve information flow throughout the network. O_t^3 is the last BRNN layer output and $\mathbf{x}$ is the model input. I_t, is the concatenation of them, and is fed to the full connected layer:

$$\begin{aligned} O_t^1 &= Concat(F_t, B_t) \\ O_t^{2,3} &= Sum(F_t, B_t) \\ I_t &= Concat(O_t^3, \mathbf{x}) \end{aligned} \tag{4}$$

2.4.2. Algorithm-level approaches for combating imbalance classification

Only surface residues have the potential to become interacting residues, and thus interacting residues have more solvent accessibility. Two different but correlated types of results are predicted by performing multi-task learning in two shared multi-perception layers. The output from our

proposed model consists of the predicted PPIS labels $\hat{y_i}$ and RSA labels $\hat{s_i}$. The joint loss function is formulated as follows:

$$L(\{y_i\},\{s_i\}) = -\frac{1}{N}\sum_{i=1}^{N} y_i * log(\hat{y_i}) - \frac{1}{N}\sum_{i=1}^{N} s_i * log(\hat{s_i}) \tag{5}$$

The first part of the equation (5) is the loss function for PPIS prediction and the second part is the loss function of RSA prediction. $\hat{y_i}$ and $\hat{s_i}$ are predicted probabilities of PPIS labels and RSA labels respectively. y_i and s_i are ground-truth labels of PPIS and RSA respectively. N is the number of residues.

Similar to addressing cost sensitive losses in a convolutional neural network [22], a constant matrix named the penalization factor is attached to the softmax function without any modifications to the loss function. The prediction output of PPIS via the softmax function is as follows:

$$\hat{y_i} = \frac{exp(\xi_i \ o_i)}{\sum\limits_{k} exp(\xi_k \ o_k)} \tag{6}$$

where ξ is the penalization factor, which is empirically valued by label distribution and $\xi = [1.821, 1.0, 1.821]$. ξ is computed as :

$$\xi = \alpha \times |D_{maj}|/|D_{min}| \tag{7}$$

α is the coefficient. $|D_{maj}|$ is the number of non-interacting residues and $|D_{min}|$ is the number of interacting residues.

3. Experimental results and analysis

3.1. Experimental setup

In this study, 200, 400 and 400 units are used in the first, second and third BRNN layers, respectively. The output dimensionality of each BRNN layer is 400. Sixty-four hidden nodes are used in the first fully connected layer and the following fully connected layer is the classification layer with the softmax function. A weight constraint of dropout (p = 0.5) used to avoid overfitting is applied to the output of each hidden layer. To obtain a better overall performance model, the F-measure is used as a measurement in each iterations of updating the model.

In our experiments, an Adam optimizing function is used for training the entire network with the default setting parameters. The default learning rate is initially set at 0.0008 with a decreasing ratio 0.2, whereas the F-measure on the validation dataset does not increase after more than 20 epochs. The learning rate threshold is set to 0.0002. When the model iterated about 200 epochs, it converged and the predictive performance is stabilized. The training procedure is illustrated in Fig.3 and summerized as Algorithm 1.

Our model is implemented in Keras, which is a publicly available deep-learning software. The weights in the DLPred are initialized using default values, and the entire network is trained on a single NVIDIA GeForce GTX 1080 Ti GPU with 12GB memory.

Algorithm 1 The trainging algorithm of DLPred

```
Input: TR5860( 5719 sequences for training, 141 squences for validating).
initialize ξ, batches, early_ stoping, non_ increasing_ threshold, learning_ rate;
while termination criterion is not met do
  for k = 1; k ≤ ⌈training_dataset_size/batches⌉; k++ do
    train DLPred using training_dataset(batches*k);
  end for
  F-masure ← validate DLPred using validating_dataset;
  if F-masure ≥ Previous_F-masure then
    Previous_F-masure = F-masure ;
    non_increasing_count=0;
  else
    non_increasing_count+=1;
    g_non_increasing_count+=1;
  end if
  if non_increasing_count ≥ non_increasing_threshold then
    if learning_rate > 0.0002 then
      learning_rate=learning_rate*0.8 ;
    else
      learning_rate=0.0002 ;
    end if
    non_increasing_count=0;
  end if
  if g_non_increasing_count ≥ early_stoping then
    stop training model ;
  end if
end while
Ouput: DLPred model.
```

To comparatively evaluate the prediction performance of DLPred, the six routine measurements Recall, Precision, Specificity, Accuracy, Matthew's Correlation Coefficient (MCC) and the

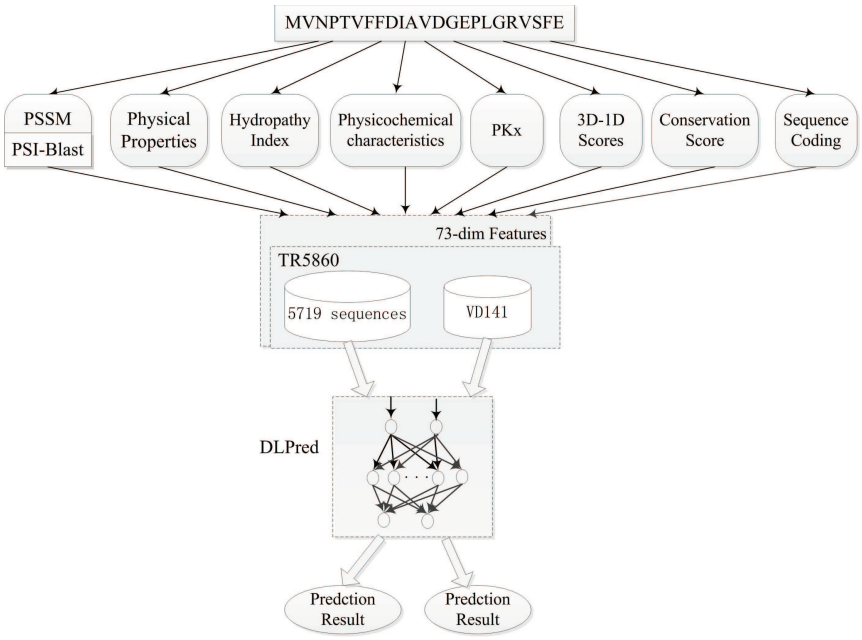

Figure 3: Follow chart of training DLPred.

F-measure are used to evaluate the performance. Their definitions are given as follows:

$$
\begin{aligned}
&Recall = TP/(TP + FN)\\
&Precision = TP/(TP + FP)\\
&Specificity = TN/(TN + FP)\\
&Accuracy = (TP + TN)/(TP + TN + FP + FN)\\
&MCC = (TP \times TN - FP \times FN)/\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}\\
&F - measure = 2 \times Recall \times Precision/(Recall + Precision)
\end{aligned}
\tag{8}
$$

Where, true positives (TP) are residues correctly predicted as interacting residues, false positives (FP) are residues incorrectly predicted as interacting residues, true negatives (TN) are residues correctly predicted as non-interacting residues, and false negatives (FN) are residues incorrectly predicted as non-interacting residues. Among these measurements, the MCC and F-measure can measure the overall performance of a prediction method. The MCC is a correlation coefficient between the observed and predicted binary classifications while the F-measure corresponds to the

harmonic mean of precision and recall.

Besides the above measurements, receiver operating characteristic (ROC) curves are also plotted for comparing different methods. The area under the ROC curve (AUC) can give a threshold-independent evaluation of the overall performance of classifiers.

3.2. Analysis of input features

As subsection 2.2 described, eight groups of sequence-derived features are used to denote one residue. We analyze the effectiveness of input features on the performance of prediction in this subsection. The eight groups of input features are fed to the DLPred model respectively. The performances of F-measure and accuracy on independently validating dataset VD141 are compared and shown in Figure 4, 5 respectively. In the training period, we analyze the influence according to the changes of F-measure and accuracy under the input features. It can be concluded that PSSM is the most important feature. When the index of F-measure is being improved, the index of accuracy is being improved too. When the input features are PKx or Hydropathy Index or Physicochemical Characteristics, the predicting accuracy is low. For the predicting accuracy of interacting residues is high, these features are helpful for dealing with imbalanced classification.

3.3. Selection attempts to decide an effective model architecture

To determine an effective architecture of the deep learning model, we attempted different types of BRNNs, including bidirectional GRU (BGRU), bidirectional SRU (BSRU), bidirectional LSTM (BLSTM) and BSLSTM with the same hyper-parameters and training dataset. The F-measure is used to evaluate the performance on an independent validating dataset VD141.

The comparison results are shown in Table 1. The model using BLSTM has 5,595,526 parameters and the performance of F-measure is 51.7%. The iteration time by this model is about 160 seconds. It can be seen that this model is under-fitting and the speed of training is the slowest of all of the models. The model using BSRU has 2,071,526 parameters and it has the fastest speed, but the generalization performance is poor. The best performance is obtained by the model using BGRU or BSLSTM. However, the model using BSLSTM has less parameters and its training speed is faster.

To understand more details, a comparison of the iteration procedure between DLPred models using BSLSTM and BLSTM, BGRU , or BSRU are illustrated in Figures 6. Figure 6 indicates that the performance of the BSLSTM-based model is comparable to the BGRU-based model. However,

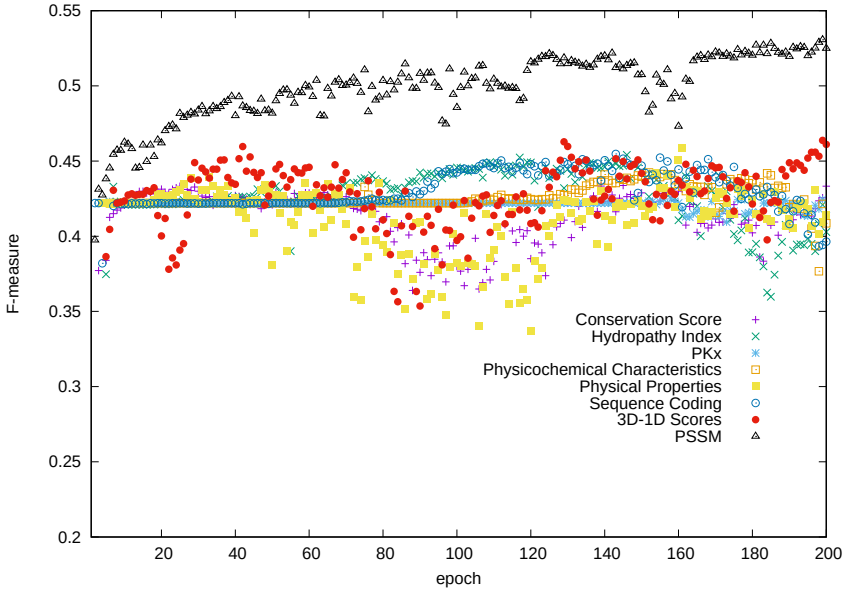

Figure 4: Performance of F-measure on independent validating dataset under different group of input features.

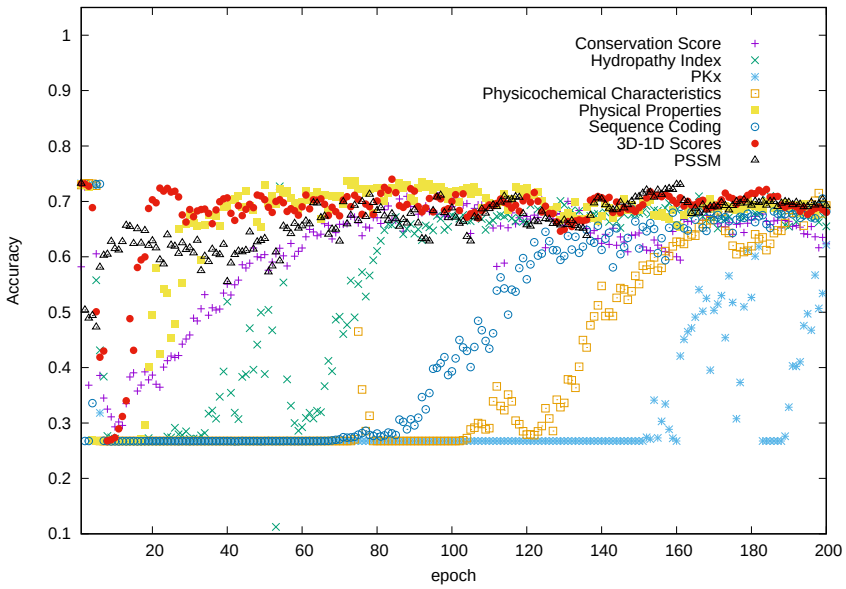

Figure 5: Performance of accuracy on independent validating dataset under different group of input features.

the BSLSTM-based model is more stable, having fewer parameters and running faster. For all of the concerned with the model parameters, training speed and prediction performance, we decide that the model using the BSLSTM is the best.

Table 1: Model performance comparison where DLPred is implemented by different BRNNs.

Model	Time(s)/Epoch	Parameters	F-measure(%)	Accuracy(%)
DLPred using BLSTM	160	5,595,526	51.7	71.5
DLPred using BGRU	111	4,204,131	54.6	71.8
DLPred using BSLSTM	80	3,435,526	54.7	71.3
DLPred using BSRU	55	2,071,526	51.6	71.3

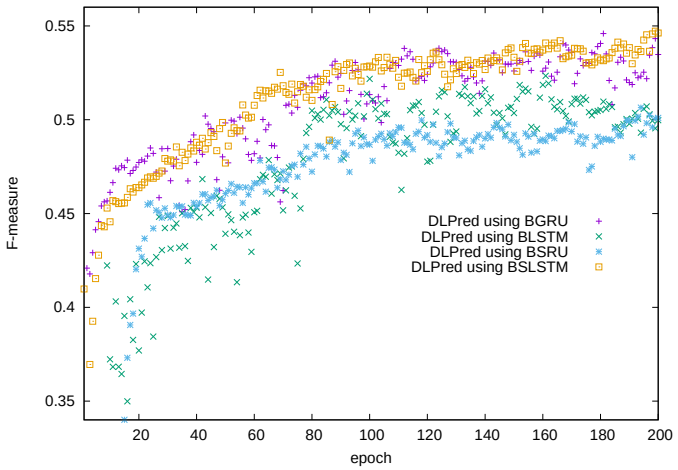

Figure 6: Comparison of the iteration procedure between DLPred models using BSLSTM and BLSTM, BGRU , or BSRU. The F-measure is used to monitor the iteration procedure.

3.4. Options for reducing imbalance effects on prediction performance

To identify vital elements for reducing the negative effect of the imbalance issue on the classification performance, we conduct an ablation study by removing or re-placing multi-task learning, training datasets or the cross-entry loss function.

The iteration time of DLPred with multi-task learning (MTL) and without multi-task learning is plotted in Fig 7. It is suggested that multi-task learning of interacting sites prediction and solvent accessibility prediction is effective for rectifying the imbalance classification.

To understand the effect of our constructed training dataset and cost sensitive learning to cope with the imbalance issue, we try to four groups of experiments:

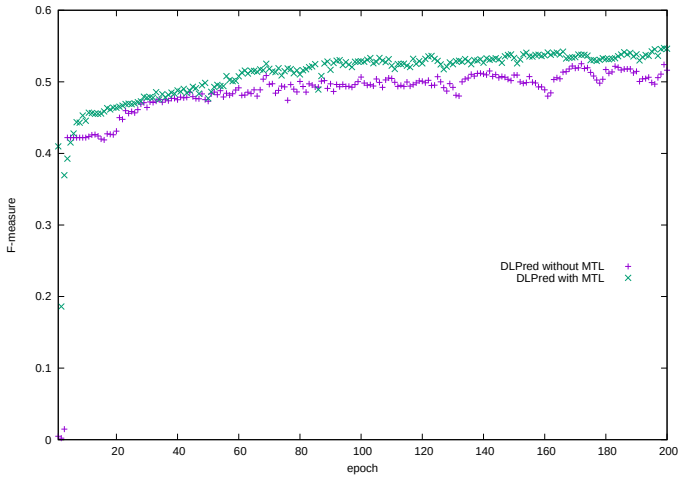

Figure 7: Iteration time of DLPred with multi-task learning (MTL) and without multi-task learning. The F-measure is used to monitor the iteration procedure.

- DLPred model is trained by an un-filtered dataset containing 4369 sequences and a normal cross-entry loss function (named as DLPred using UTD and NCE).
- DLPred model is trained by a filtered dataset(TR5860) and a normal cross-entry loss function (named as DLPred using FTD and NCE).
- DLPred model is trained by an un-filtered dataset containing 4369 sequences and an adjusted cross-entry loss function with the penalization factor (named as DLPred using UTD and ACE).
- DLPred model is trained by a filtered dataset(TR5860) and an adjusted cross-entry loss function with the penalization factor (named as DLPred using FTD and ACE).

The comparison is shown in Figure 8. It can been seen that Results of DLPred using UTD and NCE is unsatisfactory. then we try to train DLPred using a filtered training dataset and the normal cross-entry loss function (DLPred using FTD and NCE),The performance is improved. Or, we try to train DLPred using an un-filtered training dataset and an adjusted cross-entry loss function with the new penalization factor (DLPred using NTD and ACE), The performance is also improved. When the model is trained on the filtered training dataset(TR5860) and the adjusted cross-entry loss function with the new penalization factor (DLPred using FTD and ACE), the performance is effectively improved.

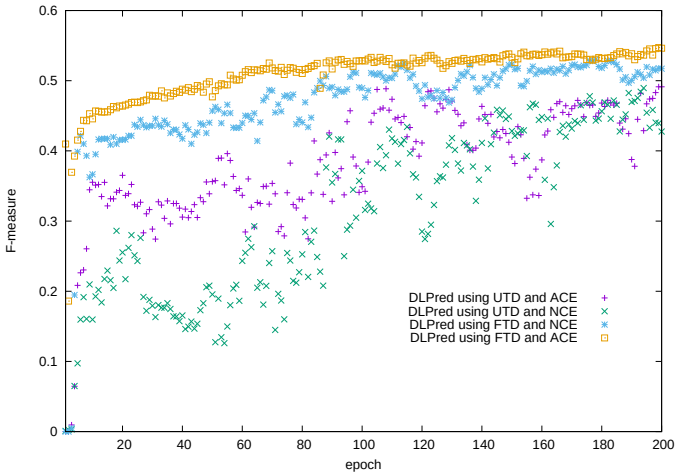

Figure 8: The affects of training dataset and cost-sensitive learning on DLPred are compared, where DLPred is implemeted by DLPred using UTD and NCE, DLPred using FTD and NCE, DLPred using UTD and ACE, DLPred using FTD and ACE.

3.5. Performance Comparisons with existing PPISs predictors

To evaluate the performance of our proposed model, we compared its performance with those of SPPIDER [7], PSIVER [13], SPRINGS [8], LORIS [14] and SSWRF [15] on three independent test datasets Dset186, PDBtestset164 and Dtestset72. In previous research, each sequence in the test dataset was tested independently and average indexes on all sequences were reported. In our experiment, both the average indexes and overall performance of all sequences in the test dataset are provided.

3.5.1. Comparison on PDBtestset164 dataset

Comparison of averaged performance on the PDBtestset164 are presented in Table 2. DLPred achieved an MCC of 18.1%, F-measure of 38.8% and accuracy of 68.4%. Compared with SSWRF, our MCC, F-measure and accuracy are improved by 2.9%, 2.3% and 6.3% respectively. Figure 9 plots the ROC curves of DLPred and SSWRF on the PDBtestset164. The AUC value of DLPred and SSWRF is 0.789 and 0.635 respectively, which shows an improvement of 15.4%.

The performance of F-measure on each sequence of Pdbtestset164 dataset is compared in Figure 10. The sequence is arranged in length from short to long. "Positive rate" is the rate of interacting residues in the sequence. When the positive rate of long sequence is less than 10%, the performance

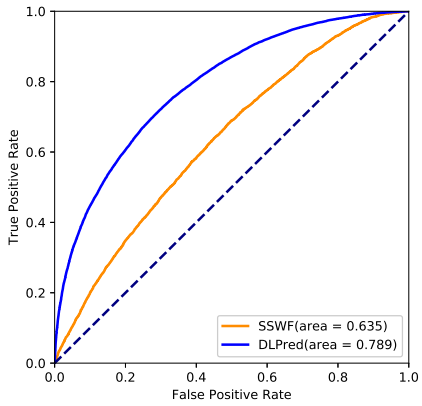

Figure 9: ROC curve comparison between DLPred and SSWRF on the Pdbtestset164 dataset.

of DLPred is inferior. The performance on each residue or secondary structure in Pdbtestset164 dataset is also illustrated in Figure 11, 12 respectively . "Mean" denotes the averaged results and "ACC" denotes accuracy.

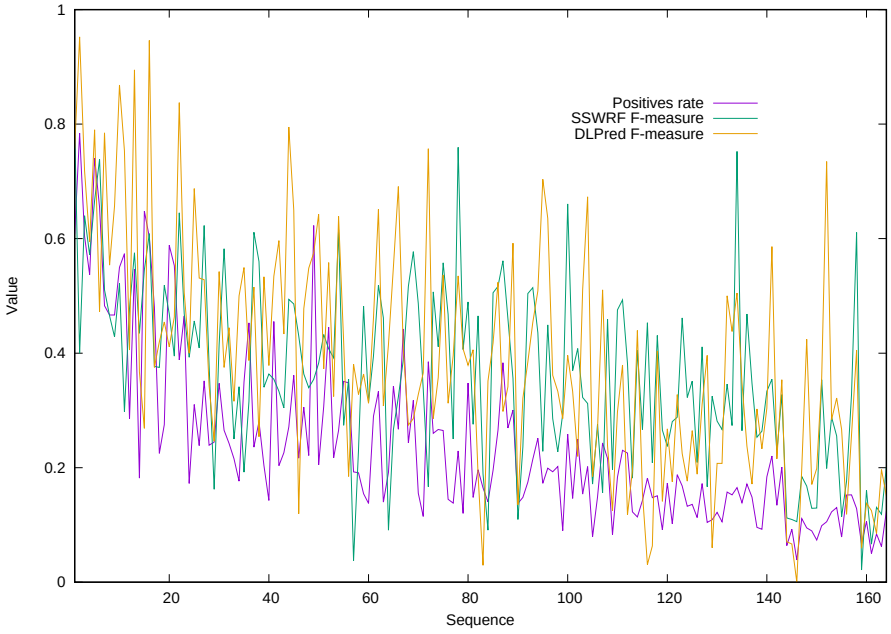

Figure 10: F-measure comparison between DLPred and SSWRF on the sequences of Pdbtestset164 dataset.

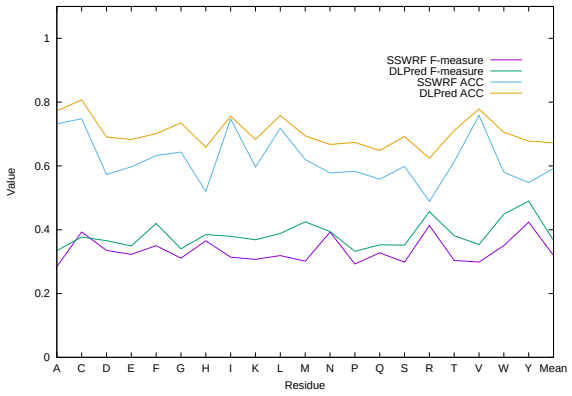

Figure 11: F-measure and Accuracy comparison between DLPred and SSWRF on residues of Pdbtestset164 dataset.

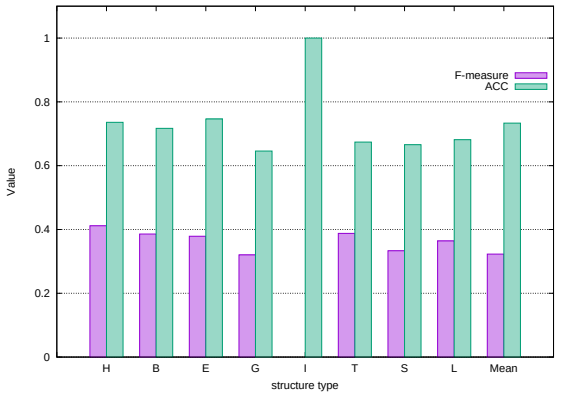

Figure 12: Performance of DLPred on secondary structures of Pdbtestset164 dataset.

We also compared DLPred with SSWRF on the performance of all sequences. DLPred achieved better performance here as well. The P-value of significance test between DLPred and SSWRF on the PDBtestset164 is 1.16E-5 (<0.005).

Table 2: Performance comparison between DLPred and existing methods on the independent dataset PDBtestset164.

Method	MCC(%)	F-measure(%)	Recall	Specificity	Precision	Accuracy(%)
PSIVER	7.8	29.5	0.464	0.634	0.253	59.6
SPPIDER	1.5	12.9	0.162	**0.851**	0.231	**71.6**
SPRINGS	10.8	31.1	0.407	0.648	0.268	60.6
LORIS	11.1	32.3	0.538	0.609	0.263	58.8
CRF	15.5	37.0	**0.543**	0.645	0.323	61.3
SSWRF	15.2	36.5	0.527	0.656	0.323	62.1
DLPred	**18.1**	**38.8**	0.503	0.701	**0.341**	68.4
SSWRF[ab]	13.9	33.5	**0.514**	0.661	0.248	63.4
DLPred[a]	**21.4**	**38.1**	0.491	**0.76**	**0.312**	**71.1**

[a] Overall performance of all sequences. Others are the average results of each sequence.
[b] Data are generated by our experiments and the others are from their papers.

3.5.2. Comparison on the Dtestset72 dataset

The performance comparisons on Dtestset72 are shown in Table 3. We obtain an F-measure of 42.6%, slightly better than that of SSWRF (42.3%). The accuracy of DLPred is 69.0%, showing an improvement of 4.4%. However, the MCC value of DLPred is worse than that of SSWRF. All of the proteins in our experiment are split in a single sequence. However, some sequences that have been split, such as 1JMOL, cannot compute the metrics of MCC . The performance on the overall dataset is also compared in Table 3 and the results demonstrate that our method is more effective than the other methods. From the ROC curves plotted in Figure 13, the AUC values of DLPred and SSWRF are 0.811 and 0.729 respectively. Our method brings an improvement of 8.2%.

3.5.3. Comparison on the Dset186 dataset

To further validate our model, performances on Dset186 are also compared. We note that the performances by DLPred are independently validated while the performances of the other methods are leave-one-out cross-validation results. As Table 4 illustrates, DLPred achieved 22.7%, 38.9% and 69.1% for average MCC, F-measure and accuracy respectively. Although our MCC is weaker than that of SSWRF and the F-measure is slightly weaker than that of CRF, the accuracy of our method are better.

Table 3: Performance comparison between DLPred and existing methods on the independent dataset Dtestset72.

Method	MCC(%)	F-measure(%)	Recall	Specificity	Precision	Accuracy(%)
SPPIDER	9.1	24.5	0.35	**0.762**	0.21	**70.9**
PSIVER	13.5	27.8	0.465	0.693	0.25	66.1
SPRINGS	17.0	31.8	0.59	0.63	0.241	62.4
LORIS	17.7	32.4	0.631	0.61	0.238	61.4
CRF	20.9	34.0	**0.64**	0.64	0.256	64.0
SSWRF[b]	**25.0**	42.3	0.635	0.659	0.351	65.6
DLPred	22.7	**42.6**	0.544	0.702	**0.387**	69.0
SSWRF[ab]	27.3	45.5	**0.633**	0.689	0.355	67.8
DLPred[a]	**29.8**	**46.5**	0.553	**0.779**	**0.401**	**73.1**

[a] Performance on the overall dataset.
[b] Data are generated by our experiments and others are from their papers.

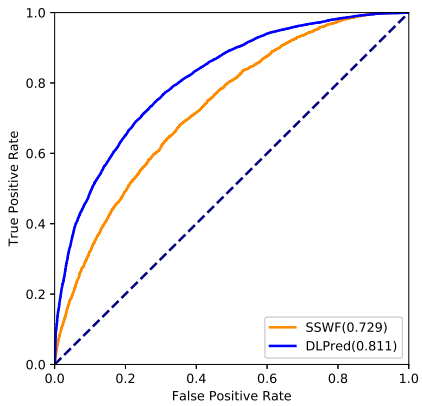

Figure 13: ROC curves of DLPred and SSWRF on Dtestset72.

Table 4: Performance comparison between DLPred with the existing methods on Dset186.

Method	MCC(%)	F-measure(%)	Recall	Specificity	Precision	Accuracy(%)
LORIS	22.1	38.4	**0.698**	0.586	0.287	60.4
PSIVER	15.1	35.3	0.416	**0.743**	0.306	67.3
CRF	**23.5**	**39.0**	0.612	0.674	0.318	66.2
SSWRF	23.4	38.6	0.581	0.697	**0.322**	67.9
DLPred	22.7	38.9	0.568	0.70	0.32	**69.1**
DLPred[a]	23.8	37.7	0.556	0.747	0.285	71.8

[a] Performance on the overall dataset.
Performances of our model are independently validated, while performances of other methods are leave-one-out cross-validation results.

3.5.4. Validating on homodimeric and heteromeric proteins

To evaluate our DLpred model as a more general predictor, we apply DLPred predictor to the independent heteromeric dataset dset_48 and five homodimeric sequences. The performances of DLPred, SSRF and a random forest interface predictor(RF_hetero)[12] for heteromeric complexes on dset_48 (a subset filtered from Dtestset72) are compared. The detailed test performances on dset_48 testset by RF_hetero, SSWRF and DLPred are presented in Table 5. Our method achieves 31.04%, 47.61% and 73.68% for MCC, F-measure and accuracy respectively, which are all better than those of the state-of-the-art. DLPred also outperformed the other two methods in ROC plots (see Fig 14). The AUC score achieved by DLPred is 81.81%, which is 7.29% higher than that of SSWRF and 16.59% higher than that of RF_hetero.

Table 5: Performance comparison of RF_hetero, SSWRF, and DLPred on heteromeric dset_48.

Method	MCC(%)	F-measure(%)	Recall	Specificity	Precision	Accuracy(%)	AUC(%)
RF_hetero	13.36	23.27	0.538	0.679	0.149	66.58	65.21
SSWRF	27.75	46.01	**0.631**	0.696	0.362	68.22	74.51
DLPred	**31.04**	**47.61**	0.554	**0.787**	**0.418**	**73.68**	**81.81**

All data are generated by our experiments.

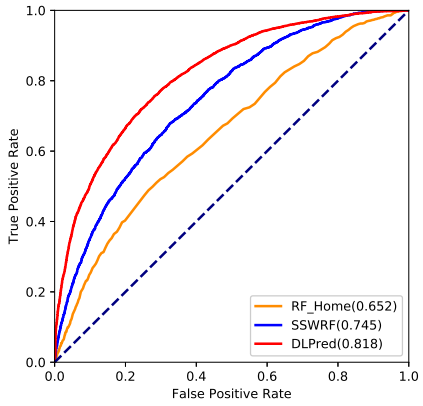

Figure 14: ROC curves of RF_hetero, SSWRF and DLPred on the independent heteromeric dataset dset_48.

Five sequences from homodimeric complex proteins(1H9R, 1K66, 1QXR, 1TC1 and 3SQF) are used as homodimeric test dataset. The performances of Accuracy and F-measure are illustrated in

Figure 15. The F-measure of all sequences is more than 44% and the predicting accuracy is not less than 60%. It is suggested that our deep learning model is a well-performing interface predictor.

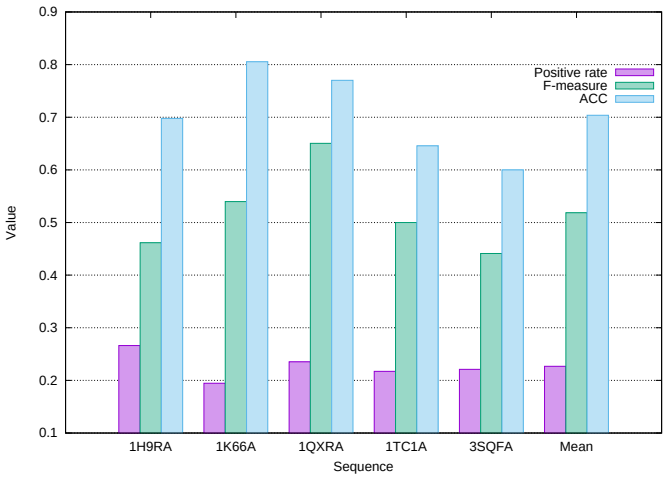

Figure 15: Performance of DLPred on five homomeric sequences.

4. Conclusion

We have presented a novel deep learning method for improving the prediction performance of protein interacting residues. This is an imbalanced classification problem. We proposed to use a simplified Long-short Term Memory (SLSTM) network to design the deep learning model. Three ideas are used to deal with the imbalance issue: collection of protein sequences having a high ratio of interacting residues for the training dataset, a new penalization factor introduced in the loss function, and multi-task learning of PPIS prediction and residue solvent accessibility prediction. Substantial computational results confirm that our deep learning model outperforms the state-of-the-art method for the accurate prediction of protein interacting residues from protein sequences.

ACKNOWLEDGMENTS

This research was supported in part by National Natural Science Foundation of China (No. 61170125 and 31801108), the Special Program for Applied Research on Super Computation of the NSFC-Guangdong Joint Fund (the second phase) and the Natural Science Research Project of Anhui Provincial Department of Education (No. KJ2018A0383).

References

[1] H. Zhou, Y. Shan, Prediction of protein interaction sites from sequence profile and residue neighbor list, Proteins-structure Function & Bioinformatics 44 (3) (2001) 336.

[2] H. Neuvirth, R. Raz, G. Schreiber, Promate: a structure based prediction program to identify the location of protein-protein binding sites, Journal of Molecular Biology 338 (1) (2004) 181–199.

[3] G. Drewes, T. Bouwmeester, Global approaches to protein-protein interactions, Current Opinion in Cell Biology 15 (2) (2003) 199–205.

[4] S. Jones, J. M. Thornton, Analysis of protein-protein interaction sites using surface patches, Journal of Molecular Biology 272 (1) (1997) 121–132.

[5] Y. Ofran, B. Rost, Isis: interaction sites identified from sequence, Bioinformatics 23 (2) (2007) e13–e16.

[6] H. Chen, H.-X. Zhou, Prediction of interface residues in protein-protein complexes by a consensus neural network method: test against nmr data., Proteins:structure Function & Bioinformatics 61 (1) (2005) 21–35.

[7] A. Porollo, J. Meller, Prediction-based fingerprints of protein-protein interactions, Proteins Structure Function & Bioinformatics 66 (3) (2007) 630–645.

[8] G. Singh, K. Dhole, P. P. Pai, S. Mondal, Springs: Prediction of protein-protein interaction sites using artificial neural networks, Journal of Proteomics & Computational Biology 1 (1).

[9] D. Lei, J. Guan, Q. Dong, S. Zhou, Prediction of protein-protein interaction sites using an ensemble method, Bmc Bioinformatics 10 (1) (2009) 426–426.

[10] P. Chen, L. Wong, J. Li, Detection of outlier residues for improving interface prediction in protein heterocomplexes, IEEE/ACM Transactions on Computational Biology & Bioinformatics 9 (4) (2012) 1155–1165.

[11] M. Šikić, S. Tomić, K. Vlahovicek, Prediction of protein-protein interaction sites in sequences and 3d structures by random forests, Plos Computational Biology 5 (1).
URL `https://doi.org/10.1371/journal.pcbi.1000278`

[12] Q. Hou, P. F. G. De Geest, W. F. Vranken, J. Heringa, K. A. Feenstra, Seeing the trees through the forest: sequence-based homo- and heteromeric protein-protein interaction sites prediction using random forest, Bioinformatics 33 (10) (2017) 1479–1487. doi:10.1093/bioinformatics/btx005.
URL http://dx.doi.org/10.1093/bioinformatics/btx005

[13] Y. Murakami, K. Mizuguchi, Applying the naïve bayes classifier with kernel density estimation to the prediction of proteincprotein interaction sites, Bioinformatics 26 (15) (2010) 1841.

[14] D. Kaustubh, S. Gurdeep, P. P. Priyadarshini, M. Sukanta, Sequence-based prediction of protein-protein interaction sites with l1-logreg classifier, Journal of Theoretical Biology 348 (10) (2014) 47–54.

[15] Z. S. Wei, K. Han, J. Y. Yang, H. B. Shen, D. J. Yu, Protein-protein interaction sites prediction by ensembling svm and sample-weighted random forests, Neurocomputing 193 (C) (2016) 201–212.

[16] Z. S. Wei, J. Y. Yang, H. B. Shen, D. J. Yu, A cascade random forests algorithm for predicting protein-protein interaction sites, IEEE Transactions on Nanobioscience 14 (7) (2015) 746–760.

[17] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735.

[18] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, in: Conference on Empirical Methods in Natural Language Processing, 2014, pp. 1724–1734.

[19] H. He, E. A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge & Data Engineering 21 (9) (2009) 1263–1284.

[20] Z. H. Zhou, X. Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Transactions on Knowledge & Data Engineering 18 (1) (2006) 63–77.

[21] Y. Sun, A. K. C. WONG, M. S. KAMEL, Classification of imbalanced data: a review, International Journal of Pattern Recognition & Artificial Intelligence 23 (04) (2009) 687–719.

[22] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, R. Togneri, Cost-sensitive learning of deep feature representations from imbalanced data, IEEE Transactions on Neural Networks & Learning Systems 29 (8) (2018) 3573–3587.

[23] M. Galar, A. Fernndez, E. Barrenechea, F. Herrera, Eusboost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling, Pattern Recognition 46 (12) (2013) 3460–3471.

[24] R. L. D. J. Guoli Wang, Pisces: a protein sequence culling server, Bioinformatics 19 (12) (2003) 1589–91.

[25] W. Li, A. Godzik, Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences, Bioinformatics 22 (13) (2006) 1658.

[26] G. H. Liu, H. B. Shen, D. J. Yu, Prediction of protein-protein interaction sites with machine-learning-based data-cleaning and post-filtering procedures, Journal of Membrane Biology 249 (1-2) (2016) 141–153.

[27] H. Hwang, B. Pierce, J. Mintseris, J. Janin, Z. Weng, Protein-protein docking benchmark version 3.0, Proteins: Structure, Function & Bioinformatics 73 (3) (2008) 705–709.

[28] J. Mihel, M. Šikić, S. Tomić, B. Jeren, K. Vlahoviček, Psaia - protein structure and interaction analyzer, BMC Structural Biology 8 (1) (2008) 21.

[29] W. Kabsch, C. Sander, Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features, Biopolymers 22 (12) (1983) 2577–2637.

[30] H. Naderi-Manesh, M. Sadeghi, S. Arab, A. A. Moosavi Movahedi, Prediction of protein surface accessibility with information theory, Proteins 42 (4) (2001) 452–459.

[31] S. F. Altschul, E. M. Gertz, R. Agarwala, A. A. Schaäffer, Y.-K. Yu, Psi-blast pseudocounts and the minimum description length principle, Nucleic Acids Research 37 (3) (2009) 815–824.

[32] M. Jens, M. Michael, Z. Anita, S. Felix, Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks, Journal of Molecular Modeling 7 (9) (2001) 360–369.

[33] W. WC, W. SH, Experimentally determined hydrophobicity scale for proteins at membrane interfaces., Nature Structural Biology 3 (10) (1996) 842–848.

[34] K. Jack, F. D. Russell, A simple method for displaying the hydropathic character of a protein , Journal of Molecular Biology 157 (1) (1982) 105–132.

[35] L. Nan, S. Zhonghua, J. Fan, Prediction of protein-protein binding site by using core interface residue and support vector machine, BMC Bioinformatics 9 (1) (2009) 553.

[36] D. R. Lide, CRC handbook of chemistry and physics, CRC Press, Inc, 1992.

[37] J. Bowie, R. Luthy, D. Eisenberg, A method to identify protein sequences that fold into a known three-dimensional structure, Science 253 (5016) (1991) 164–170.

[38] C. Fan, D. Liu, R. Huang, Z. Chen, L. Deng, Predrsa: a gradient boosted regression trees approach for predicting protein solvent accessibility, BMC Bioinformatics 17 (1) (2016) S8.

[39] L. Quan, Q. Lv, Y. Zhang, Strum: structure-based prediction of protein stability changes upon single-point mutation, Bioinformatics 32 (19) (2016) 2936.

[40] B. Zhang, J. Li, L. Qiang, Prediction of 8-state protein secondary structures by a novel deep learning architecture, BMC Bioinformatics 19 (1).

[41] R. Jozefowicz, W. Zaremba, I. Sutskever, An empirical exploration of recurrent network architectures, in: Proceedings of the 32nd International Converenfe on Machine Learning (ICML), 2015, pp. 171–180.

[42] Y. Lecun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[43] A. Graves, Generating sequences with recurrent neural networks, ArXiv e-prints abs/1308.0850. arXiv:1308.0850.
URL http://arxiv.org/abs/1308.0850

[44] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[45] K. Greff, R. K. Srivastava, J. Koutnk, B. R. Steunebrink, J. Schmidhuber, Lstm: A search space odyssey, IEEE Transactions on Neural Networks & Learning Systems 28 (10) (2017) 2222–2232.

[46] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, M. Ranzato, Learning longer memory in recurrent neural networks, CoRR abs/1412.7753. arXiv:1412.7753.
URL http://arxiv.org/abs/1412.7753

[47] D. Balduzzi, M. Ghifary, Strongly-typed recurrent neural networks, in: International Conference on Machine Learning, 2016, pp. 1292–1300.

[48] T. Lei, Y. Zhang, Y. Artzi, Training rnns as fast as cnns, ArXiv e-prints abs/1709.02755. arXiv:1709.02755.
URL http://arxiv.org/abs/1709.02755

[49] J. Bradbury, S. Merity, C. Xiong, R. Socher, Quasi-recurrent neural networks, ArXiv e-prints abs/1611.01576. arXiv:1611.01576.
URL http://arxiv.org/abs/1611.01576

[50] M. Schuster, K. K. Paliwal, Bidirectional recurrent neural networks, IEEE Transactions on Signal Processing 45 (11) (1997) 2673–2681.

[51] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, Los Alamitos, CA, USA, 2016, pp. 770–778. doi:doi.ieeecomputersociety.org/10.1109/CVPR.2016.90.