DELPHI: accurate deep ensemble model for protein interaction sites prediction

Yiwei Li[1], Lucian Ilie[1,*],

1 Department of Computer Science
University of Western Ontario

*** ilie@uwo.ca**

Abstract

Motivation: Proteins usually perform their functions by interacting with other proteins, which is why accurately predicting protein-protein interaction (PPI) binding sites is a fundamental problem. Experimental methods are slow and expensive. Therefore, great efforts are being made towards increasing the performance of computational methods.

Results: We propose DELPHI (DEep Learning Prediction of Highly probable protein Interaction sites), a new sequence-based deep learning suite for PPI binding sites prediction. DELPHI has an ensemble structure with data augmentation and it employs novel features in addition to existing ones. We comprehensively compare DELPHI to nine state-of-the-art programs on five datasets and show that it is more accurate.

Availability: The trained model, source code for training, predicting, and data processing are freely available at `https://github.com/lucian-ilie/DELPHI`. All datasets used in this study can be downloaded at `http://www.csd.uwo.ca/~ilie/DELPHI/`.

Contact: ilie@uwo.ca

Introduction

Protein-protein interactions (PPI) play a key role in many cellular processes such as signal transduction, transport and metabolism [53]. Proteins interact by forming chemical bonds with other proteins. The bonding amino acid residues are protein-protein interaction binding sites. Detecting PPI binding sites helps understand cell regulatory mechanisms, locating drug target, predicting protein functions [8]. Databases like PDB [7] store protein binding sites information deriving from the 3D structure of each protein, but the available protein structures are limited. Experimental methods such as two-hybrid assay and affinity systems are usually time and labour intensive [39]. Computational methods are needed to bridge the gap, and many have been developed [?, 3, 4, 9–11, 15, 18, 20, 22, 23, 25, 28–31, 33, 34, 40, 44, 45, 48, 49, 51, 52, 54]. Out of the above mentioned twenty six computational methods, all but one are machine learning based. Computational methods can be classified into three categories, sequenced based, structure based, and combined. Among them, sequence-based approaches are usually faster and cheaper. They are also more universal because comparing to sequence information, structure information is still limited.

Machine learning methods use feature groups to represent each protein sequence. Widely used features such as position-specific scoring matrix (PSSM), evolutionary conservation (ECO), putative relative solvent accessibility (RSA) have been assessed in [55]. High-scoring segment pair (HSP) has been used in previous methods for PPI prediction [27]. One-hot vectors [51, 52] and amino acid embedding [5, 6, 19] have also been empirically explored to represent protein sequences.

The learning structure is crucial to PPI binding sites classification problems. Previously explored architectures include random forest [?, 45], SVM [45], logistic regression [54], Bayes classifier [33], artificial neural networks [40]. Recently, convolutional neural network (CNN) [51] and recurrent neural network (RNN) [52] have also been applied to solve this problem.

We introduce a new sequence-based PPI binding sites prediction method, DELPHI (DEep Learning Prediction of Highly probable protein Interaction sites), that combines a CNN and a RNN structure. It uses twelve feature groups to represent protein sequences including three novel features, high-scoring segment pair (HSP), position information, and a reduced 3-mer amino acid embedding (ProtVec1D).

We have comprehensively compared DELPHI with nine state-of-the-art programs on five datasets. DELPHI provides the best predictions in all metrics. The contributions of the DELPHI study are as follows. First, a novel fine tuned ensemble model combing CNN and RNN is constructed in which three features are used the first time in this problem. Second, a many-to-one structure, that not only serves as a data augmentation technique but also improves the prediction performance, is applied. Third, a data processing and feature construction suite, including new features, is provided, aiming to alleviating the difficulty of tedious feature computation by the users.

Materials and Methods

Datasets

Training and validation datasets

A large, high quality dataset was provided in [55]. In this dataset, Uniprot sequences are annotated with Protein, DNA, RNA, and small ligands binding information at the residue level. We further processed this dataset as follows. First, we kept only the sequences with protein-protein binding information to focus on protein-protein binding. 1 Then we removed any sequences from training dataset sharing more than 25% similarities, as measured by PSI-CD-hit [17, 26], with any sequences in testing datasets. It is well acknowledged that similar sequences between training and testing datasets negatively affect the generalization of the evaluated performance of a machine learning model. Also, proteins with higher levels of similarity can be accurately predicted by the alignment-based methods [53]. The similarity threshold is picked differently by different programs ranging from 25% to 50%. We picked the strictest value of 25% to match to one of the closest competing programs, SCRIBER [54], for a fair comparison. We used PSI-CD-HIT because it is fast, accurate and well maintained in the CD-HIT suite. Also, it is able to cluster sequences with similarity at low as 25%, whereas CD-HIT works only down to 40%. Finally, we ran PSI-CD-hit again on the rest of the training protein sequences so no sequences shared more than 25% similarities among training data. This ensures the training data is as diverse as possible. A dataset of 9,982 protein sequences was constructed. From it, we randomly pick eight ninth (8,872) as the training dataset and one ninth (1,110) as the validation dataset.

Testing datasets

Five datasets are used in the comparative assessment. Four of them are publicly available dataset from previous studies: Dset_186 [33], Dset_72 [21], Dset_164 [13], and

Table 1. The datasets used for training, validation, and the comparative testing. The first column are the dataset names. The second column contains the number of proteins in each dataset. The third, fourth, and fifth columns represent the total number of residue, the number of binding, and the number of non-binding residues in each dataset. The last column represents the percentage of the binding residues out of total.

Dataset	Proteins	Residues			% binding
		total	binding	non-binding	out of total
Dset_448	448	116,500	15,810	100,690	13.57%
Dset_355	355	95,940	11,467	84,473	11.95%
Dset_186	186	36,219	5,517	30,702	15.23%
Dset_72	72	18,140	1,923	16,217	10.60%
Dset_164	164	33,681	6,096	27,585	18.10%
Training + validation	9,982	4,254,198	427,687	3,826,511	10.05%

Dset_448 [54]. The first three have been widely used and explored by previous studies while Dset_448 is more recent. The raw data of Dset_448 was from the BioLip database [50] where binding sites are defined if the distance between an atom of a residues and an atom of a given protein partner ¡0.5 Å plus the sum of the Van der Waals radii of the two atoms. The raw data was further processed by the authors of [54], removing protein fragments, mapping BioLip sequences to UniProt sequences, clustering so that no similarities above 25% is shared within the dataset. The number of sequences, binding, non-binding, and the ratio of binding residues are shown in Table 1. The training dataset of DLPred has some overlaps with Dset_448, therefore we removed 93 proteins in Dset_448 that share more than 40% similarities with DLPred training data and formed a reduced testing dataset of 355 proteins (Dset_355). According to the PSI-CD-HIT results, the training datasets of DLPred and SCRIBER still contain some proteins that share more than 25% similarity to Dset_186, Dset_72, and Dset_164. However, we kept these testing datasets untouched because otherwise we would have to remove too many proteins from them. The training datasets of the competing programs can not be changed because the models are pre-trained. Note that all testing data share less than 25% similarity to the training dataset of DELPHI.

Input features

DELPHI uses 12 features groups, shown in Table 2, including high-scoring segment pair (HSP), a variation of 3-mer amino acid embedding (ProtVec1D), position information, position-specific scoring matrix (PSSM), evolutionary conservation (ECO), putative relative solvent accessibility (RSA), relative amino acid propensity (RAA), putative protein-binding disorder, hydropathy index, physicochemical characteristics, physical properties, and PKx. Each input is represented by a 39 dimensional feature vector profile. To the best of our knowledge, this study is the first time that HSP and ProtVec1D are being used in binding sites classification problems. The computation of each of these two new features is described next. High-scoring segment pair (HSP): An HSP is a pair of similar sub-sequences between two proteins. The similarities between two sub-sequence of the same length are measured by scoring matrices such as PAM and BLOSUM. SPRINT [27] is used for computing all HSPs as it detects similarities fast and accurately among all proteins in training and testing. After obtaining the HSPs, the score for the ith residue, $P[i]$, of a testing protein P, denoted $\text{HSP}_{\text{score}}(P[i])$, is calculated as follows. Assume we have an HSP, (u, v), between P and a training protein Q such that u covers the residue $P[i]$, that is, position i in P is within the range covered by u. Let j be the position in Q that corresponds to i, that is, the distance in

Table 2. Feature groups used by DELPHI. The first column indicates the name of each feature. The second column describes the program used to obtain the feature. "Load" means the value for a specific amino acid is fixed, and it is loaded in DELPHI program. "Compute" means DELPHI further computes the feature. The last column shows the dimension of each feature group.

Feature	Program	Dimension
High-scoring segment pair (HSP)	SPRINT and compute	1
3-mer amino acid embedding (ProtVec1D)	Load and compute	1
Position information	Compute	1
Position-specific scoring matrix (PSSM)	Psi-Blast	20
Evolutionary conservation (ECO)	Hhblits	1
Putative relative solvent accessibility (RSA)	ASAquick	1
Relative amino acid propensity (RAA)	Load	1
Putative protein-binding disorder	ANCHOR	1
Hydropathy index	Load	1
Physicochemical characteristics	Load	3
Physical properties	Load	7
PKx	Load	1

P from the beginning of u to i is the same as the distance in Q from the beginning of v to j. If $Q[j]$ is a known interacting residue, then we add the PAM120 score between $P[i]$ and $Q[j]$ to the HSP score of $P[i]$:

$$\text{HSP}_{\text{score}}(P[i]) = \sum_{\substack{\text{HSPs covering } P[i] \\ Q[j] \text{ interacting residue}}} \max(0, \text{PAM120}(P[i], Q[j])) \ .$$

The 3-mer amino acid embedding (ProtVec1D): We developed this feature based on ProtVec [6]. ProtVec uses word2vec [32] to construct a one hundred dimensional embedding for each amino acid 3-mer. It is shown in [6] that ProtVec can be applied to problems such as protein family classification, protein visualization, structure prediction, disordered protein identification, and protein-protein interaction prediction. Since using the ProtVec embedding in our program slows down significantly the deep learning model, especially during training, we replaced the one hundred dimensional vector by one dimensional value, which is the sum of the one hundred components; we call this ProtVec1D. According to our tests, ProtVec1D achieves, in connection with the other feaures, the same prediction performance as ProtVec.

Position information: In natural language processing tasks, position information is shown useful. The popular network Bert [?] utilizes this information to guide its translation process. It is also shown by DeepPPISP [51] that the global information of a protein helps the prediction of interfaces. Inspired by the two networks, we use the position information of each amino acid as an input feature hoping that it provides certain global information of a protein. The position of an amino acid in a protein is in the range of 1 to the length of the protein. Then the position is divided by protein's length so that the value is between 0 to 1.

Position-specific scoring matrix (PSSM): PSSM matrices are widely used in protein interaction related problems. They contain the evolutionary conservation of each amino acid position by aligning an input sequence with protein databases. The PSSM matrices are computed using PSI-Blast [2] with the expectation value (E-value) set to 0.001 and the number of iterations set to 3. PSI-Blast performs multiple alignment on each input sequence against the non-redundant database.

Evolutionary conservation (ECO): ECO also contains evolutionary conservation, but in a more compact way. To compute the ECO score, the faster multiple alignment tool HHBlits [38] is run against the non-redundant Uniprot20 database with default parameters. The one dimensional conservation value is computed using the formula described in [55].

Putative relative solvent accessibility (RSA): The solvent accessibility is predicted using ASAquick [16]. The values are obtained in the from rasaq.pred file in each output directory.

Relative amino acid propensity (RAA): The AA propensity for binding is quantified as relative difference in abundance of a given amino acid type between binding residues and the corresponding non-binding residues located on the protein surface. The RAA for each amino acid type is computed in [55] by using the program of [43].

Putative protein-binding disorder: The putative protein-binding disorder is computed using the ANCHOR program [14].

Hydropathy index: Hydrophobicity scales is experimentally determined transfer free energies for each amino acid. It contains energetics information of protein-bilayer interactions [47]. The values are computed in [24].

Physicochemical characteristics: For each protein, this includes three features: the number of atoms, electrostatic charges and potential hydrogen bonds for each amino acid. They are taken from [52].

Physical properties: We use a 7-dimensional property of each amino acid type. They are a steric parameter (graph-shape index), polarizability, volume (normalized van der Waals volume), hydrophobicity, isoelectric point, helix probability and sheet probability. The pre-computed values are taken from [52].

PKx: This is the negative of the logarithm of the dissociation constant for any other group in the molecule. The values for each amino acid type is taken from [52].

After computing all the feature vectors, the values in in each row vector are normalized to a number between 0 to 1 using formula (1) where v is the original feature value, and max and min are the biggest and smallest value observed in the training dataset, resp. This is to ensure each feature group are of the same numeric scale and help the model converge better:

$$v_{\text{norm}} = \frac{v - \min}{\max - \min} \tag{1}$$

0.1 Model architecture

DELPHI has an architecture that is inspired by ensemble learning. The intuition of the design is that different components of the model capture different information, and another deep neural network is trained to only select the most useful ones. As shown in Fig. 1, the model consists of three parts, a convolutional neural network (CNN) component, an recurrent neural network (RNN) component, and an ensemble component. The core layers of the CNN and RNN components are convolution and bidirectional gated recurrent units (GRU) layers. The ensemble model decodes the output of the first two components.

Another very useful characteristic of the model is its many-to-one structure, meaning that the information of many residues are used to prediction the binding propensity of the centered single residue. As illustrated in Fig. 2, for each amino acid as the prediction target, a window of size 31, centred on the amino acid position, is used to collect information from the neighbouring 30 residues to help the prediction. A sliding window is used to capture each 31-mer. The size 31 is determined experimentally. The beginning and the ending part of the sequence are padded with zeros. The many-to-one structure has two advantages. Firstly, it serves as a data augmentation technique. Deep learning models need large amount of data to train, and comparing to image classifiers,

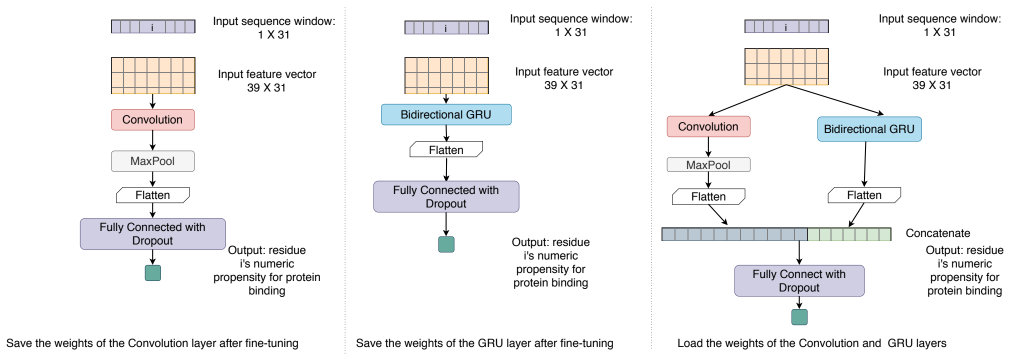
Input sequence window: 1 X 31
Input feature vector 39 X 31
Convolution
MaxPool
Flatten
Fully Connected with Dropout
Output: residue i's numeric propensity for protein binding
Save the weights of the Convolution layer after fine-tuning
Bidirectional GRU
Save the weights of the GRU layer after fine-tuning
Concatenate
Fully Connect with Dropout
Load the weights of the Convolution and GRU layers

Figure 1. The architecture of DELPHI. Left: the CNN component of the model. Middle: the RNN component of the model. Right: The ensemble model.

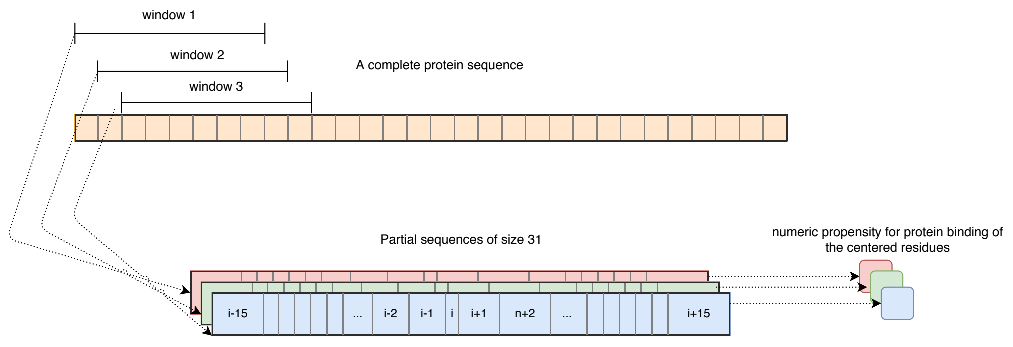
window 1
window 2
window 3
A complete protein sequence
Partial sequences of size 31
numeric propensity for protein binding of the centered residues
i-15
...
i-2
i-1
i
i+1
n+2
...
i+15

Figure 2. The many-to-one prediction. Sliding windows of size 31, stride 1 are put on top of an input protein sequence. Each time, a sub-sequence of length 31 is extracted. The model predicts the protein-binding propensity of the middle amino acid for each sub-sequence.

models in proteomics have access to orders of magnitude less data. Using each residue multiple times during the training process helps the model learn better. Secondly, it makes the model more robust. The lengths of protein sequence vary from less than one hundred to several thousand, and most a many-to-many models have a fixed input length of near 500. During training, sequences around length 500 are often picked. However, during testing, input sequences are random and need to be either padded or cut into pieces. The different average lengths between training and testing could potentially make the model less general.

Architecture of the CNN network

The CNN model one has a concise structure: one convolution layer, one maxpooling layer, one flatten layer, and two fully connected layers. For each input sub-sequence of size 31, a 2D feature profile of size 39 by 31 is constructed. The 2D vector is reshaped into 3D and then passed to a convolution 2D layer, followed by a maxpooling layer. The intuition of using the convolution and maxpooling layers is that a 2D protein profile vector can be considered as an image with one channel, and the CNN model captures the combination of several features in a partial image. The results are flattened and then fed into two fully connected layers with dropout for regularization. The last fully connected layer has one unit with activation function sigmoid, so that the output is a single value between 0 to 1. The higher the value, the more confident the CNN model claims that the residue is a PPI binding site.

Architecture of the RNN network

The RNN component has the following structure: one bidirectional GRU layer, one flatten layer, and two fully connected layers. Similar to the CNN component, a 2D feature profile of size 39 by 31 is built for each 31-mer. The feature profile is passed to a bidirectional gated recurrent units (GRU) layer with the intention to memories the dependency and relationship among the 31 residues. We set the GRU layer to return a whole sequence as opposed to return a single value. The results are flattened and fed into two fully connected layers with dropout. The output of the RNN network is also a single value between 0 to 1.

Architecture of the ensemble network

The final model combines the core layers of the above mentioned CNN and RNN models and tries to further extract essential information of protein binding. The ensemble network takes a sequence of length 31 as its input. Similar to the CNN and RNN components, a 39 by 31 feature vector is constructed and passed to both a convolution layer and a bidirectional GRU layer. The output of the convolution layer is passed on to a maxpool layer and then flattened. The GRU output is also flattened. Then the outputs of the two flatten layers are concatenated and passed on to two fully connected layers with dropout. The last fully connected layer has one output unit with a sigmoid activation function, so the final output is a single value between 0 to 1, indicating the propensity of being binding sites. This is the final output of the entire model.

Fine tuning is used in this ensemble model. The convolution layer in the CNN network and the bidirectional GRU layer in the RNN network are tuned separated using the same training/validation dataset. After achieving the best performance on the CNN and the RNN components, the weights of the convolution and the GRU layer are saved to files. In the ensemble model, the convolution and the GRU layer load the saved weights from the file and freeze the weights, so that during the process of training, the convolution and the GRU layer stay unchanged. Training and validation data are used again only to train the fully connected layers in the ensemble model.

Implementation

The program is written in Keras [12] with TensorFlow GPU [1] back end. All features are computed from sequence only. We alleviate the burden of feature computation from users by providing all computation programs and a pipeline script. We ease the system configuration process by providing users a pip package list which enables one-command installation.

Classifying protein binding residue is an imbalanced problem. To cope with that, different class weights [42] are assigned to the positive and negative samples, so that the model pays more attention to the minority class, which is the binding sites. The values are determined by the inverse of the class distribution in the training datasets. In our program, the weights are 0.55 and 4.97 for the non-binding and binding sites respectively.

During training, we shuffle the data before each epoch. Since the sliding window is used to extract each 31-mer, adjacent data entries are very similar; only the first and the last residue differ from the previous and the next data entry. Shuffling the whole training data diversifies the input in each batch. We experimentally trained the model with and without data shuffling, and shuffling the data rendered better predictions.

Parameter tuning

Parameters and hyper-parameters are chosen based on the training dataset while applying early stopping [37] on the validation set. Early stopping halts the training process when a performance drop on the validation set is detected. This is to avoid overfitting the training dataset. We chose all parameters with the purpose to maximize area under the precision-recall curve (AUPRC) of the training data. All testing results are then carried using the already tuned model. All parameters and hyper-parameters used in this model are shown in Table 3.

The DELPHI model takes 2.1 hours to train the CNN component, 0.5 hour to train the RNN component and 1.3 hours to train the ensemble model on our testing cluster.

Table 3. Parameters used in DELPHI. Parameters are divided into four groups: CNN, RNN, ensemble model, and hyper-parameters.

Parameter	Value
Epoch in CNN	8
Kernel size in CNN	5
Stride in CNN	1
Padding in CNN	valid
Number of filters in CNN	64
Fully connected unit in CNN	64, 1
Epoch in RNN	9
GRU unit	32
Fully connected unit in RNN	64, 1
Epoch in ensemble	5
Fully connected unit in ensemble	96, 1
Batch size	1024
Dropout rate	0.3
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.999$)
Patience in early stop	4
Loss function	binary cross entropy
Learning rate	0.002

Results

Competing methods

We have comprehensively compared DELPHI with nine state-of-the-art machine learning based methods. The methods are selected using the following criteria. First, the program is a sequence-based method as sequence information is readily available for most proteins. Second, the program is available in the form of source code or web server. Lastly, the program takes in any input sequence in FASTA format and produces the results on an average-length protein within thirty minutes. Following these criteria, DLpred [52], SCRIBER [54], SSWRF [45], SPRINT [41], CRF-PPI [46], LORIS [13], SPRINGS [40], PSIVER [33], and SPPIDER [36] are selected.

All competing methods are pre-trained using their own training and validation datasets. The most revent two programs, DLPred and SCRIBER, use 5719 and 843 training proteins respectively. The training dataset of DLPred is obtained from CullPDB datasets [**?**] and further filtered by the authors. The SCRIBER training dataset is originally from the BioLip database. This dataset contains also protein binding information with DNA, RNA, and ligand, which is used by SCRIBER.

All tests have been performed on a Linux (Ubuntu 16.04) machine with 24 CPUs (Intel Xeon v4, 3.00GHz), 256GB memory, and a Nvidia Tesla K40c GPU.

Evaluation scheme

Similar to previous studies, we use sensitivity, specificity, precision, accuracy, F1-measure, (F1), Matthews correlation coefficient (MCC), area under the receiver operating characteristic curve (AUROC), and area under the precision-recall (AUPRC) to measure the prediction performance. All programs output a prediction value for each amino acid, and thus the receiver operating characteristic (ROC) curve and the precision-recall (PR) curve can be drawn. AUROC and AUPRC are computed based on the curves using Scikit-learn [35]. We focus more on AUROC and AUPR because they are threshold independent and convey an overall performance measurement of a program. The rest of the metrics are calculated using a binding threshold which is determined after obtaining the prediction scores from each program. Since each program's output is of different scale, for each program, we pick the threshold such that for a given testing dataset, the number of predicted scores above the threshold is equal to the real number of binding sites in the dataset.

The formulas for calculating the metrics are as follows, where true positives (TP) and true negatives (TN) are the correctly predicted binding sites and non-binding sites, respectively, and false positives (FP) and false negative (FN) are incorrectly predicted binding sites and non-binding sites, respectively.

$$Sensitivity = \frac{TP}{TP + FN} \tag{2}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3}$$

$$Precision = \frac{\mathrm{TP}}{TP + FP} \tag{4}$$

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{5}$$

$$F1 = 2 \times \frac{Sensitivity \times Precision}{Sensitivity + Precision} \tag{6}$$

$$MCC = \frac{TP \times TN - FN \times FP}{\sqrt{(TP+FP)\times(TP+FN)\times(TN+FP)\times(TN+FN)}} \tag{7}$$

Comparative assessment of predictive performance

Performance comparison on Dset_448

We first compare the DELPHI model with eight programs on Dset_448. This dataset is the most recently published and has the largest number of proteins, so we emphasize the importance of this dataset. As shown in Table 4, DELPHI surpasses competitors in

Table 4. Performance comparison of nine programs on Dset_448. Programs are sorted in asending order by AUPRC. Darker colours represent better results. The evaluation of the programs marked with * are carried by [54].

Predictor	Sensitivity	Specificity	Precision	Accuracy	F1	MCC	AUROC	AUPRC
Dset_448								
SPPIDER*	0.202	0.870	0.194	0.781	0.198	0.071	0.517	0.159
SPRINT*	0.183	0.873	0.183	0.781	0.183	0.057	0.570	0.167
PSIVER*	0.191	0.874	0.191	0.783	0.191	0.066	0.581	0.170
SPRINGS*	0.229	0.882	0.228	0.796	0.229	0.111	0.625	0.201
LORIS*	0.264	0.887	0.263	0.805	0.263	0.151	0.656	0.228
CRFPPI*	0.268	0.887	0.264	0.805	0.266	0.154	0.681	0.238
SSWRF*	0.288	0.891	0.286	0.811	0.287	0.178	0.687	0.256
SCRIBER	0.334	0.896	0.332	0.821	0.333	0.230	0.715	0.287
DELPHI	0.371	0.901	0.371	0.829	0.371	0.272	0.737	0.337

Table 5. Performance comparison of SCRIBER, DLPred, and DELPHI on Dset_355, Dset_186, Dset_164, and Dset_72. Darker colours represent better results. Each dataset is tested separately using the same metrics. The average metrics of the three datasets is also shown.

Predictor	Sensitivity	Specificity	Precision	Accuracy	F1	MCC	AUROC	AUPRC
Dset_355								
SCRIBER	0.322	0.908	0.322	0.838	0.322	0.230	0.719	0.275
DLPred	0.308	0.906	0.308	0.835	0.308	0.214	0.724	0.272
DELPHI	0.364	0.914	0.364	0.848	0.364	0.278	0.746	0.326
Dset_186								
SCRIBER	0.279	0.870	0.279	0.780	0.279	0.150	0.647	0.246
DLPred	0.320	0.878	0.320	0.793	0.320	0.198	0.694	0.290
DELPHI	0.351	0.884	0.351	0.803	0.351	0.235	0.710	0.319
Dset_164								
SCRIBER	0.327	0.851	0.327	0.756	0.327	0.179	0.657	0.301
DLPred	0.338	0.854	0.338	0.760	0.338	0.192	0.672	0.330
DELPHI	0.352	0.857	0.352	0.765	0.352	0.209	0.685	0.332
Dset_72								
SCRIBER	0.232	0.909	0.232	0.837	0.232	0.141	0.680	0.198
DLPred	0.246	0.901	0.246	0.826	0.246	0.148	0.688	0.215
DELPHI	0.274	0.914	0.274	0.847	0.274	0.189	0.711	0.237
Average								
SCRIBER	0.290	0.885	0.290	0.803	0.290	0.175	0.676	0.255
DLPred	0.303	0.885	0.303	0.803	0.303	0.188	0.695	0.277
DELPHI	0.335	0.892	0.335	0.816	0.335	0.227	0.713	0.304

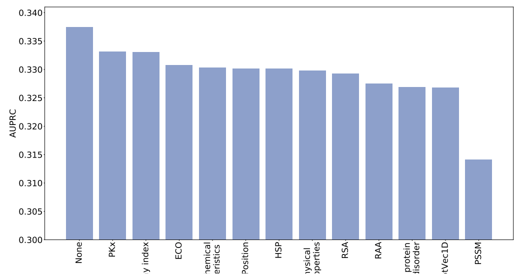

Figure 3. The areas under PR curves with the removal of one out of the twelve features on Dset_448. One feature is removed each time, and the DELPHI model is trained, validated, and tested using the remaining eleven features. The x-axis shows the removed features where 'None' indicates using all twelve features, and the y-axis is the AUPR achieved. The features are sorted by the AUPR values.

all metrics with an improvement of 3.08% and 17.4% on AUROC and AUPR respectively comparing to the second best program SCRIBER.

In order to have a more comprehensive comparison, we compare DELPHI with another recent sequence-based program, DLPred. However, the training data in DLPred has a big overlap with Dset_448, so we removed the highly similar sequence in Dset_448 using the protocol described in section and produced a reduced dataset Dset_355 which is tested in Section 0.1.

Performance comparison on Dset_355, Dset_186, Dset_164, and Dset_72

To further compare DELPHI with other programs, we used Dset_355 along with another three previously published datasets: Dset_186, Dset_164, and Dset_72. Since DLPred and SCRIBER are the most recent programs and have been shown to outperform older programs, we compare DELPHI only with SCRIBER and DLPred on the remaining datasets. As shown in Table 5, DELPHI clearly outperforms the competitors in all metrics on all datasets it shares the least similarities to the testing datasets. As binding site prediction is a highly imbalanced task, the area under the PR curve is a better indication of the performance as it emphasises more on the minority class and ROC curve pays attention to both minority and the majority class [**?**]. The AUPR is improved by 18.5%, 10.0%, 0.6%, 10.2% comparing to the second best program on each dataset. We present also the average values over Dset_355, Dset_186, Dset_164, and Dset_72 in Table 5 for each specificity value. The performance of DELPHI again surpasses all competing programs. The average AUPR improvement is 9.7%.

Feature evaluation

We conducted an another experiment to show that all twelve feature are necessary for DELPHI. We pruned one feature each time, and the remaining eleven features are used to train and then evaluate the DELPHI model. As shown in Fig. 3, the performance decreases with the removal of any feature, showing that there are no redundant features. It is perhaps expected that removing PSSM creates the biggest performance drop, but our newly introduced features, HSP, ProtVec1D, and Position are shown to be very useful as well.

Software availability

DELPHI is an open source program under the GPLv3 License. The trained model, source code of training, predicting, and data processing is freely available at `https://github.com/lucian-ilie/DELPHI`. All datasets used in this study can be downloaded at `http://www.csd.uwo.ca/~ilie/DELPHI/`.

Conclusion

We have presented a new deep learning model and program DELPHI, for predicting PPI binding sites. We compared DELPHI with nine current state-of-the-art programs on five datasets and demonstrated that DELPHI has a higher prediction performance. There is still plenty room for improvement on this topic as the highest AUC in all test is 0.747. We hope in the future, the model architecture, the usage of the two new features, and the many-to-one structure can be extended to predicting protein biding with other types of molecules, such as DNA, RNA, and ligand.

Acknowledgements

We would like to thank Jian Zhang and Buzhong Zhang for running their programs on some of the testing datasets as well as the insightful information they provided regarding SCRIBER and DLPred. We appreciate Giancarlo Colmenares for the suggestions on tuning the network and Min Zeng for the explaination on DeepPPISP.

Funding

This work was supported by a Discovery Grant (R3143A01) and a Research Tools and Instruments Grant (R3143A07) from the Natural Sciences and Engineering Research Council of Canada.

References

1. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

2. S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997.

3. A. Amos-Binks, C. Patulea, S. Pitre, A. Schoenrock, Y. Gui, J. R. Green, A. Golshani, and F. Dehne. Binding site prediction for protein-protein interactions and novel motif discovery using re-occurring polypeptide sequences. *BMC bioinformatics*, 12(1):225, 2011.

4. E. B. Asadabadi and P. Abdolmaleki. Predictions of protein-protein interfaces within membrane protein complexes. *Avicenna journal of medical biotechnology*, 5(3):148, 2013.

5. E. Asgari, A. C. McHardy, and M. R. Mofrad. Probabilistic variable-length segmentation of protein sequences for discriminative motif discovery (dimotif) and sequence embedding (protvecx). *Scientific reports*, 9(1):1–16, 2019.

6. E. Asgari and M. R. Mofrad. Continuous distributed representation of biological sequences for deep proteomics and genomics. *PloS one*, 10(11), 2015.

7. H. M. Berman, T. Battistuz, T. N. Bhat, W. F. Bluhm, P. E. Bourne, K. Burkhardt, Z. Feng, G. L. Gilliland, L. Iype, S. Jain, et al. The protein data bank. *Acta Crystallographica Section D: Biological Crystallography*, 58(6):899–907, 2002.

8. L. Bonetta. Interactome under construction. *Nature*, 468(7325):851–852, 2010.

9. B. Cao, A. Porollo, R. Adamczak, M. Jarrell, and J. Meller. Enhanced recognition of protein transmembrane domains with prediction-based structural profiles. *Bioinformatics*, 22(3):303–309, 2006.

10. P. Chen and J. Li. Sequence-based identification of interface residues by an integrative profile combining hydrophobic and evolutionary information. *BMC bioinformatics*, 11(1):402, 2010.

11. X.-w. Chen and J. C. Jeong. Sequence-based prediction of protein interaction sites with an integrative method. *Bioinformatics*, 25(5):585–591, 2009.

12. F. Chollet et al. Keras. `https://keras.io`, 2015.

13. K. Dhole, G. Singh, P. P. Pai, and S. Mondal. Sequence-based prediction of protein–protein interaction sites with l1-logreg classifier. *Journal of theoretical biology*, 348:47–54, 2014.

14. Z. Dosztányi, B. Mészáros, and I. Simon. Anchor: web server for predicting protein binding regions in disordered proteins. *Bioinformatics*, 25(20):2745–2746, 2009.

15. X. Du, J. Cheng, and J. Song. Improved prediction of protein binding sites from sequences using genetic algorithm. *The protein journal*, 28(6):273–280, 2009.

16. E. Faraggi, Y. Zhou, and A. Kloczkowski. Accurate single-sequence prediction of solvent accessible surface area using local and global features. *Proteins: Structure, Function, and Bioinformatics*, 82(11):3170–3176, 2014.

17. L. Fu, B. Niu, Z. Zhu, S. Wu, and W. Li. Cd-hit: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–3152, 2012.

18. H. Geng, T. Lu, X. Lin, Y. Liu, and F. Yan. Prediction of protein-protein interaction sites based on naive bayes classifier. *Biochemistry research international*, 2015, 2015.

19. M. Heinzinger, A. Elnaggar, Y. Wang, C. Dallago, D. Nechaev, F. Matthes, and B. Rost. Modeling aspects of the language of life through transfer-learning protein sequences. *BMC bioinformatics*, 20(1):723, 2019.

20. H. Hwang, D. Petrey, and B. Honig. A hybrid method for protein–protein interface prediction. *Protein Science*, 25(1):159–165, 2016.

21. H. Hwang, B. Pierce, J. Mintseris, J. Janin, and Z. Weng. Protein–protein docking benchmark version 3.0. *Proteins: Structure, Function, and Bioinformatics*, 73(3):705–709, 2008.

22. J. Jia, Z. Liu, X. Xiao, B. Liu, and K.-C. Chou. ippbs-opt: a sequence-based ensemble classifier for identifying protein-protein binding sites by optimizing imbalanced training datasets. *Molecules*, 21(1):95, 2016.

23. D. T. Jones, D. W. Buchan, D. Cozzetto, and M. Pontil. Psicov: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. *Bioinformatics*, 28(2):184–190, 2012.

24. J. Kyte and R. F. Doolittle. A simple method for displaying the hydropathic character of a protein. *Journal of molecular biology*, 157(1):105–132, 1982.

25. E. Laine and A. Carbone. Local geometry and evolutionary conservation of protein surfaces reveal the multiple recognition patches in protein-protein interactions. *PLoS computational biology*, 11(12), 2015.

26. W. Li and A. Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

27. Y. Li and L. Ilie. Sprint: ultrafast protein-protein interaction prediction of the entire human interactome. *BMC bioinformatics*, 18(1):485, 2017.

28. G.-H. Liu, H.-B. Shen, and D.-J. Yu. Prediction of protein–protein interaction sites with machine-learning-based data-cleaning and post-filtering procedures. *The Journal of membrane biology*, 249(1-2):141–153, 2016.

29. N. London, D. Movshovitz-Attias, and O. Schueler-Furman. The structural basis of peptide-protein binding strategies. *Structure*, 18(2):188–199, 2010.

30. S. Maheshwari and M. Brylinski. Prediction of protein–protein interaction sites from weakly homologous template structures using meta-threading and machine learning. *Journal of Molecular Recognition*, 28(1):35–48, 2015.

31. S. Maheshwari and M. Brylinski. Template-based identification of protein–protein interfaces using efindsiteppi. *Methods*, 93:64–71, 2016.

32. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

33. Y. Murakami and K. Mizuguchi. Applying the naïve bayes classifier with kernel density estimation to the prediction of protein–protein interaction sites. *Bioinformatics*, 26(15):1841–1848, 2010.

34. Y. Ofran and B. Rost. Isis: interaction sites identified from sequence. *Bioinformatics*, 23(2):e13–e16, 2007.

35. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

36. A. Porollo and J. Meller. Prediction-based fingerprints of protein–protein interactions. *Proteins: Structure, Function, and Bioinformatics*, 66(3):630–645, 2007.

37. L. Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998.

38. M. Remmert, A. Biegert, A. Hauser, and J. Söding. Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment. *Nature methods*, 9(2):173, 2012.

39. B. A. Shoemaker and A. R. Panchenko. Deciphering protein–protein interactions. part i. experimental techniques and databases. *PLoS computational biology*, 3(3), 2007.

40. G. Singh, K. Dhole, P. P. Pai, and S. Mondal. Springs: prediction of protein-protein interaction sites using artificial neural networks. Technical report, PeerJ PrePrints, 2014.

41. G. Taherzadeh, Y. Yang, T. Zhang, A. W.-C. Liew, and Y. Zhou. Sequence-based prediction of protein–peptide binding sites using support vector machine. *Journal of computational chemistry*, 37(13):1223–1229, 2016.

42. K. M. Ting. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(3):659–665, 2002.

43. V. Vacic, V. N. Uversky, A. K. Dunker, and S. Lonardi. Composition profiler: a tool for discovery and visualization of amino acid composition differences. *BMC bioinformatics*, 8(1):211, 2007.

44. D. D. Wang, R. Wang, and H. Yan. Fast prediction of protein–protein interaction sites based on extreme learning machines. *Neurocomputing*, 128:258–266, 2014.

45. Z.-S. Wei, K. Han, J.-Y. Yang, H.-B. Shen, and D.-J. Yu. Protein–protein interaction sites prediction by ensembling svm and sample-weighted random forests. *Neurocomputing*, 193:201–212, 2016.

46. Z.-S. Wei, J.-Y. Yang, H.-B. Shen, and D.-J. Yu. A cascade random forests algorithm for predicting protein-protein interaction sites. *IEEE transactions on nanobioscience*, 14(7):746–760, 2015.

47. W. C. Wimley and S. H. White. Experimentally determined hydrophobicity scale for proteins at membrane interfaces. *Nature structural biology*, 3(10):842–848, 1996.

48. Z. Xie, X. Deng, and K. Shu. Prediction of protein–protein interaction sites using convolutional neural network and improved data sets. *International Journal of Molecular Sciences*, 21(2):467, 2020.

49. L. C. Xue, D. Dobbs, and V. Honavar. Homppi: a class of sequence homology based protein-protein interface prediction methods. *BMC bioinformatics*, 12(1):244, 2011.

50. J. Yang, A. Roy, and Y. Zhang. Biolip: a semi-manually curated database for biologically relevant ligand–protein interactions. *Nucleic acids research*, 41(D1):D1096–D1103, 2012.

51. M. Zeng, F. Zhang, F.-X. Wu, Y. Li, J. Wang, and M. Li. Protein–protein interaction site prediction through combining local and global features with deep neural networks. *Bioinformatics*, 2019.

52. 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*, 357:86–100, 2019.

53. J. Zhang and L. Kurgan. Review and comparative assessment of sequence-based predictors of protein-binding residues. *Briefings in bioinformatics*, 19(5):821–837, 2018.

54. J. Zhang and L. Kurgan. Scriber: accurate and partner type-specific prediction of protein-binding residues from proteins sequences. *Bioinformatics*, 35(14):i343–i353, 2019.

55. J. Zhang, Z. Ma, and L. Kurgan. Comprehensive review and empirical analysis of hallmarks of dna-, rna-and protein-binding residues in protein chains. *Briefings in bioinformatics*, 20(4):1250–1268, 2019.