RESEARCH ARTICLE
Biomedical Applications
Proteomics
www.proteomics-journal.com

DeepFunc: A Deep Learning Framework for Accurate Prediction of Protein Functions from Protein Sequences and Interactions

*Fuhao Zhang, Hong Song, Min Zeng, Yaohang Li, Lukasz Kurgan, and Min Li**

Annotation of protein functions plays an important role in understanding life at the molecular level. High-throughput sequencing produces massive numbers of raw proteins sequences and only about 1% of them have been manually annotated with functions. Experimental annotations of functions are expensive, time-consuming and do not keep up with the rapid growth of the sequence numbers. This motivates the development of computational approaches that predict protein functions. A novel deep learning framework, DeepFunc, is proposed which accurately predicts protein functions from protein sequence- and network-derived information. More precisely, DeepFunc uses a long and sparse binary vector to encode information concerning domains, families, and motifs collected from the InterPro tool that is associated with the input protein sequence. This vector is processed with two neural layers to obtain a low-dimensional vector which is combined with topological information extracted from protein–protein interactions (PPIs) and functional linkages. The combined information is processed by a deep neural network that predicts protein functions. DeepFunc is empirically and comparatively tested on a benchmark testing dataset and the Critical Assessment of protein Function Annotation algorithms (CAFA) 3 dataset. The experimental results demonstrate that DeepFunc outperforms current methods on the testing dataset and that it secures the highest $F_{max} = 0.54$ and AUC = 0.94 on the CAFA3 dataset.

1. Introduction

Proteins perform many cellular functions and play indispensable role in a large variety of biological processes.[1] Protein data is being produced at a fast and even increasing pace by high-throughput sequencing techniques but their functional understanding is lagging.[2,3] Only about 1% of proteins has been probed experimentally and was manually annotated in the UniProt database.[4] Protein functions can be elucidated via in vitro and in vivo experiments.[5] However, these experimental methods are expensive, time-consuming, and do not scale with the growth of the number of protein data. This motivates the need to develop runtime-efficient and accurate computational methods that predict protein functions directly from protein data.

Many computational methods have been proposed to predict protein functions. Generally, researchers develop a pipeline to predict functions of proteins by using protein sequences according to the following steps: select useful features to encode input proteins, construct training and testing datasets, select an appropriate algorithm, and evaluate the performance. One of the most popular computational methods is BLAST that uses functions of similar sequences to functionally annotate the input sequence. However, this approach has two limitations: 1) similar and functionally annotated proteins cannot be found for many input sequences; and 2) some proteins may have similar functions while having low sequence similarity. Thus, the results obtained by these homology-based approaches are not always accurate.[6] One way to overcome the challenge is to extract useful information from conserved subregions or residues in the input protein chain. For example, Das and his collaborators proposed a domain-based method to predict protein functions.[7] Wang and his collaborators proposed a motif-based protein function classifier.[8] Moreover, some methods predict protein functions utilizing residue-level information.[9] This information may include secondary structures extracted from input protein sequences,[10] or secondary structure, disordered regions, signal peptides, and motifs like in the case of the FFPred3 method.[11] Finally, several approaches rely on the PPI-derived information to accurately predict protein functions.[12–17] The crucial idea behind these methods is that proteins which share similar topological features in the PPI networks may share similar functions.[18]

F. Zhang, Dr. H. Song, Dr. M. Zeng, Dr. Y. Li, Prof. M. Li
School of Computer Science and Engineering
Central South University
Changsha, 410083, P. R. China
E-mail: limin@mail.csu.edu
Dr. Y. Li
Department of Computer Science
Old Dominion University
Norfolk, VA, 23529, USA
Prof. L. Kurgan
Department of Computer Science
Virginia Commonwealth University
Richmond, VA, 23284, USA

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/pmic.201900019

DOI: 10.1002/pmic.201900019

Moreover, some protein function predictors utilize other types of data, such as genetic interactions,[5] genomic context,[19] protein structure,[20–23] and gene expression.[24,25] We focus on two classes of current predictors: sequence-based methods that cover the use of domains, motifs and residue-level information,[26,27] and PPI-based methods that rely on information extracted from these networks.[17,28,29] These two classes of methods utilize somehow complementary information. While topological information will be used to characterize protein functions based on protein–protein interactions, sequence-based methods could be effective in identifying proteins that incorporate signal peptides or transmembrane proteins,[30] which are not necessarily easy to predict using PPIs.

This article explores the use of deep learning to efficiently process and combine the sequence-based and PPI-based approaches. While deep learning was shown to improve predictive performance in several related prediction problems,[31–37] it was used only once in the context of combining these two types of information to predict protein functions in the DeepGO model.[28] We design and comparatively test a novel deep learning model called DeepFunc. Our sequence-based approach relies on the generation of a high-dimensional vector of information (35 000 dimensions) that describes domains, families, and motifs which are extracted by InterPro.[38] These data must be reduced before they can be combined with a relatively low-dimensional data extracted from the PPI network. We combine functional linkages from EggNOG[39] and interactions from STRING[40] to construct the PPI network. We use the Deepwalk algorithm[41] to extract a comprehensive collection of topological features that describe the underlying PPI network. The innovative aspect of DeepFunc is the use of the deep network for two distinct purposes: to convert the high-dimensional sequence-based approach into an information-rich, low-dimensional format, and to effectively combine these data with the topological information obtained from the PPI network. Consequently, comparative empirical analysis on multiple benchmark datasets reveals that DeepFunc outperforms DeepGO, as well as a few other representative function predictors, such as FFPred3 and GOPDR. The results demonstrate that the improved predictive performance is directly attributed to the extraction of high-quality sequence-based and PPI-based features. Moreover, DeepFunc obtains comparable results when testing on particularly challenging low similarity proteins.

Significance Statement

Function annotation of proteins is crucial in molecular biology. However, existing computational methods usually focus on using one type of protein data (either protein sequences or PPI network) to predict protein functions, which may cause the loss of certain protein features. In this study, a powerful deep learning framework (DeepFunc) for predicting protein functions is proposed. By using the Deepwalk algorithm, InterProscan tool, and deep learning architecture, DeepFunc extracts high-quality features of protein sequences and PPI networks. DeepFunc combines these features to predict protein functions and achieves better performance than BLAST and DeepGO.

2. Experimental Section

2.1. Datasets and Assessment Metrics

The data introduced in the DeepGO article[28] was used that is available at https://github.com/bio-ontology-research-group/deepgo. This benchmark dataset contained 60 710 proteins annotated with functions based on experimental evidence codes that were filtered to exclude long sequences and sequences that contain ambiguous amino acid codes (B, O, J, U, X, and Z). This dataset included 31 530 proteins with annotated molecular functions (MFs) and focused on the top 589 MF terms that were assigned to at least 50 proteins. The dataset was divided into a training dataset (80% of randomly selected proteins) and a testing dataset (the remaining 20% of proteins). The training dataset contained 25 224 protein sequences and the testing dataset contained 6306 protein sequences. Only the training dataset was used to parametrize DeepFunc, while the testing dataset was used to evaluate the already parametrized model. A subset of 20% of the training proteins was selected to create validation dataset that was used to empirically select the best parameters for the models trained using the remaining 80% of the training dataset, that is, parameters were optimized to maximize predictive performance on the validation dataset. Moreover, an independent (blind) empirical assessment was provided on the dataset from CAFA3 and the dataset and results from selected other predictors were downloaded from https://github.com/bio-ontology-research-group/deepgo.

Predictive performance was evaluated with three commonly used measures that included, average precision (AvgPr), average recall (AvgRc), and maximum F-measure (F_{max}) that were used in the CAFA challenge.[42] Moreover, two additional measures were used, Area Under Curve (AUC) and Mathews Correlation Coefficient (MCC), which were utilized in recent related studies.[28,43–46]

2.2. Architecture of DeepFunc

The architecture of the DeepFunc framework for the prediction of protein functions is shown in **Figure 1**. The InterPro outputs were processed using two fully connected neural layers to extract small and dense vector of sequence-based features. Concurrently, the Deepwalk algorithm was utilized to capture topological features of the PPI network in the vicinity of the input protein sequences. The feature vectors produced from the PPI network and from the sequence were concatenated and fed into a fully connected deep network that predicts protein functions.

2.2.1. Extraction of the Sequence-Derived Features

A variety of features computed from sequences-based features were used in the prediction of protein functions. They include sequence similarity,[9,47] *k*-mer frequencies,[48] and presence of certain subsequences.[49] Our approach was to encode the raw protein sequence by the vector of protein families, domains, and motifs (subsequences) that were obtained by InterPro resource. InterPro releases 70.0 contained 35 020 entries and

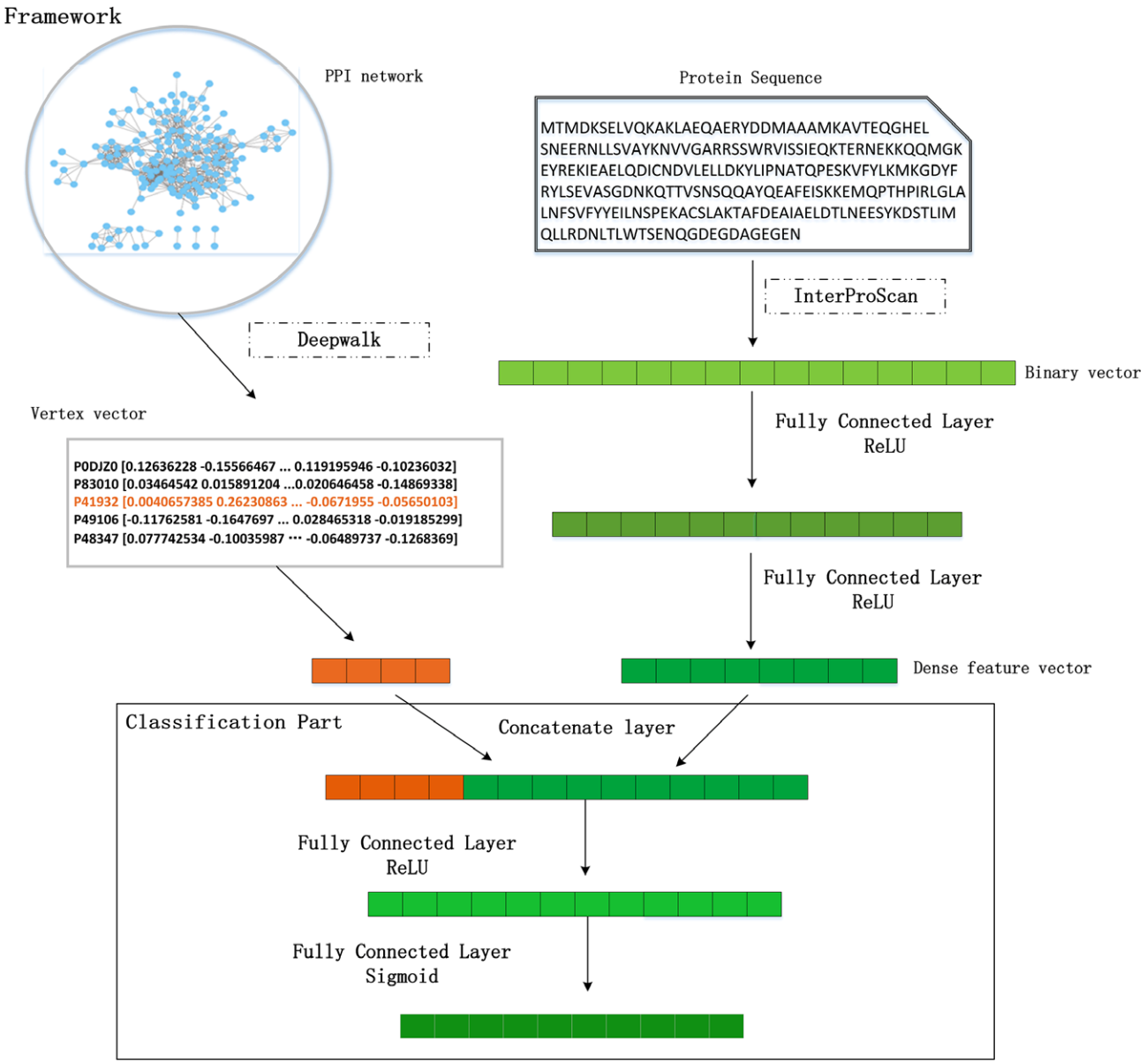

Figure 1. An overview of our proposed deep learning framework for identifying protein functions.

combined diverse information coming from 14 databases, such as CCD,[50] Pfam,[51] CATH-Gene3D,[52] and SUPERFAMILY.[53] It provided InterProScan package (http://www.ebi.ac.uk/interpro/download.htm) that scanned protein sequences and annotated information about the input sequences with 2865 superfamilies, 21 695 families, 9268 domains, 280 repeats, and 912 sites. Then, InterPro was used to encode the families, domains, and motifs information of input protein sequence into a 35 020-dimensional binary vector. In this vector, 1 means that this sequence was assigned to a given superfamily/family/domain or had a given repeat or site, otherwise, the value of 0 was assigned. This sparse and high-dimensional vector (all but a handful of the thousands of values equal 0) was not suitable as an input for the deep network. Thus, two fully connected neural layers were used to convert this vector into a substantially shorter and dense feature vector that could be effectively used to predict protein functions. The 35 020-dimensional binary input vector was processed by two fully connected layers of 1024 neurons that output 512-dimensional vector. The two layers were defined using the ReLU activation functions

$$a^l = \sigma(wa^{l-1} + b) \tag{1}$$

where a^l is the output of given fully connected layer, a^{l-1} is the corresponding input, σ is a nonlinear activation function, w is a weight matrix, and b is the bias term. The values of w and b were optimized using the training dataset and the back-propagation algorithm.

2.2.2. Extraction of the PPI Network–Derived Features

The PPI network was constructed with the help of the STRING and EggNOG resources. STRING data was downloaded on June 7, 2018 from https://string-db.orgm and the EggNOG data on June 10, 2018 from http://eggnogdb.embl.de/#/app/downloads. The network was based on the functionally annotated proteins collected from SwissProt on June 7, 2018. The SwissProt identifiers were mapped into the STRING records and a subset of these proteins was selected that have interaction confidence score of at least 300. It was worthy noting that there were some proteins without interaction information in the STRING database. Thus, functional linkages from EggNOG database were used as missing interaction information to construct the PPI network. Specifically, if two nodes both had interaction information and functional linkage, the interaction information was used as the edge between them. In terms of two nodes without interaction information, functional linkage was regarded as the interaction information. By using this strategy, coverage of PPI interactions was improved by adding functional linkages from the EggNOG database. The resulting network contained 354 687 nodes and 54 552 077 edges.

Networks had been widely used to model the structure of various biological systems and played an important role in biological prediction problems.[54–56] Thus, it was aimed to extract PPI network characteristics that were useful for the prediction of protein functions. Deep learning techniques were recently used to analyze several network-based datasets.[57–59] A few new representations of such datasets had been proposed by drawing from the natural language processing area, such as node2vec,[60] LINE,[61] and Deepwalk techniques.[41] The Deepwalk method was applied motivated by a couple of recent studies that used similar random walk approaches to capture topological features of PPI networks.[13,62] Deepwalk used each vertex (protein) as the starting point to traverse nearby vertices by using a random walk algorithm. It applied the Skip-Gram model[63] to characterize the surrounding vertices for each given central vertex by maximizing the co-occurrence likelihood between the central vertex and its neighbors. This model generated a dense, low-dimensional vector for each vertex in the PPI network that represented topological features of the underlying PPI network. In order to cover all neighbors of a central vertex as many as possible, a sampling method was used. The formula is as follows

$$\left(1 - \frac{1}{p}\right)^{k} \leq \alpha \tag{2}$$

where p is the ratio of edges to vertices. After k iterations starting from a central vertex to perform random walks, the probability that one neighbor of the central vertex was not picked at least once is small than α. In this study, our PPI network had 354 687 vertices and 54 552 077 edges; $\alpha = 0.1$ was set, and the approximate value of walk number was 300. Using the training dataset, the Deepwalk model was iteratively computed that was parametrized with walk-length = 20, window-size = 10, and the output vector size = 256. During training and testing process, a zero vector was assigned to those proteins without topological features of PPI network.

Table 1. The predictive performance of DeepFunc and other two methods on the testing dataset.

Method	F_{max}	AvgPr	AvgRc	MCC	AUC
BLAST	0.37	0.37	0.38	-	-
DeepGO	0.47	0.58	0.40	0.44	0.93
DeepFunc	**0.56**	**0.67**	**0.48**	**0.52**	**0.94**

MCC and AUC cannot be computed for the binary predictions generated with BLAST. Best results for each quality measure are highlighted in bold.

2.2.3. Design of the Deep Neural Network

The deep neural network was implemented with PyTorch,[64] a popular deep learning framework that was developed by Facebook. The topology of the network (including the two neural layers used to produce the sequence-derived features) was optimized to maximize predictive performance on the validation dataset. The 512-dimensional vector of sequence-derived information was concatenated with the 256-dimensional vector of the PPI-derived information and the resulting 768 inputs were fed into the first fully connected hidden layer with 1024 nodes that used the Rectified Linear Unit (ReLU) activation function. The second hidden layer is also fully connected and includes 1024 neurons that utilized sigmoid function to map the outputs to the range that could be interpreted as propensity for protein functions. The Adam optimizer was used with batch size = 128 and initial learning rate = 0.002 to train the deep network.

3. Results

3.1. Comparison on the Testing Dataset

We comparatively assess DeepFunc on the testing dataset against BLAST and the most related other method, DeepGO, which similarly relies on deep learning. While both DeepFunc and DeepGO provide numeric propensity scores (likelihood that a given protein has a given function), BLAST's predictions that are based on the function annotations of the most similar protein from the training dataset are binary (a given protein either has or has not a given function). The latter means that we cannot quantify MCC and AUC value for BLAST's predictions. **Table 1** shows that the predictive performance of DeepFunc is consistently better (over all five quality measures) than the predictive performance of the other two predictors. Specifically, DeepFunc secures $F_{max} = 0.56$, AvgPr = 0.67, and AvgRc = 0.48, which are better by $(0.56 - 0.37)/0.37 = 51.3$, 81.0, and 26.3% than the BLAST's predictions, respectively. Similarly, the relative improvements over DeepGO equal 19.1, 15.5, 20.0%, respectively. Moreover, the DeepFunc's MCC and AUC values are also substantially higher than the corresponding DeepGO's values (0.52 vs 0.44 and 0.94 vs 0.93).

The testing dataset includes highly similar (nearly identical) proteins when compared to the proteins in the training dataset, which are rather trivial to predict given that they would share the same functions. In order to investigate whether DeepFunc can perform well on the low similar protein sequences, raw testing set removes all protein sequences that are highly similar to the

Table 2. The predictive performance of DeepFunc on the testing dataset and low similarity testing dataset that only includes sequences that share low (<50%) similarity to the training dataset.

Dataset	F_{max}	AvgPr	AvgRc	MCC	AUC
Raw testing dataset	**0.56**	**0.67**	**0.48**	**0.52**	**0.94**
Low similarity testing dataset	0.55	0.64	**0.48**	0.51	0.93

Best results for each quality measure are highlighted in bold.

Table 3. The predictive performance of DeepFunc, DeepGO, FFPred3, and GoFDR on the CAFA3 dataset.

Method	F_{max}	AvgPr	AvgRc	MCC	AUC
FFPred3	0.38	0.35	0.40	0.29	0.86
GoFDR	0.52	**0.89**	0.36	**0.60**	0.84
DeepGO	0.47	0.61	0.39	0.37	0.90
DeepFunc	**0.54**	0.62	**0.48**	0.44	**0.94**

Best results for each quality measure are highlighted in bold.

sequences in the training dataset. Specifically, we create a low similarity subset from the raw testing dataset by using BLAST which calculates the pair-wise sequence identity of all proteins with experimental annotations. A sequence having less a certain sequence identity value is selected from the raw testing dataset and placed in the low similarity subset as the low similarity testing set. The selection of this certain value draws on prior studies that observe that functional similarity is characteristic for proteins that share >50% similarity,[65–67] and thus use of lower similarity sequences would rely on nontrivial relationships. The raw testing dataset contains 6306 protein sequences. After preprocessing, the low similarity testing set contains 1835 protein sequences. **Table 2** summarizes results for these challenging testing proteins. The results show that the performance of evaluating new testing set is slightly lower than the performance of evaluating raw testing set. When evaluating new testing set, the F_{max}, AvgPr, MCC, and AUC drops from 0.56, 0.67, 0.52, and 0.94 to 0.55, 0.64, 0.51, and 0.93, respectively. In conclusion, DeepFunc obtains satisfactory results no matter when evaluating raw testing set or low similarity testing set.

3.2. Comparison on the CAFA3 Dataset

We empirically compare DeepFunc on the CAFA3 dataset with DeepGO and two recently published and relatively highly cited methods: FFPred3 (published in August 2016; 19 citations in Google Scholar as of January 2019)[5] and GoFDR (published in January 2016; 18 citations in Google Scholar as of January 2019).[9] We use public source code of GoFDR to run its predictions. FFPred3's prediction was downloaded from http://bioinfadmin.cs.ucl.ac.uk/downloads/ffpred/cafa3/. None of the four methods (DeepFunc, DeepGO, FFPred3, and GoFDR) has used protein annotations from the CAFA3 dataset during their training.

Table 3 compares predictive performance on the CAFA3 dataset. DeepFunc secures the best values of F_{max}, AvgRc, and AUC. DeepFunc obtains $F_{max} = 0.54$ and AUC = 0.94 outperforming FFPred3 (0.38 and 0.86, respectively), GoFDR (0.52 and 0.84, respectively), and DeepGO (0.47 and 0.90, respectively). The corresponding relative improvements in F_{max} range between $(0.54 - 0.52) / 0.52 = 3.8\%$ compared to GoFDR and 42.1% when compared to FFPred3, while the increases in AUC range between 4.4% when contrasted with DeepGO and 11.9% when compared to GoFDR. Moreover, we observe that DeepFunc's performance is better than the performance of FFPred3 and DeepGO in all assessment metrics. Comparison with GoFDR reveals a tradeoff in the average precision (0.89 for GoFDR vs 0.62 for DeepFunc) and the average recall (0.36 for GoFDR vs 0.48 for DeepFunc). However, an arguably most informative AUC value, which is independent of the somehow arbitrary binarization threshold, reveals a large advantage for DeepFunc (0.94 vs 0.84). Overall, we conclude that DeepFunc outperforms the other three recently published predictors.

Table 4. The predictive performance of DeepFunc and comparison to DeepGO_Seq (DeepGO that applies only the sequence-derived inputs), DeepGO, DeepFunc_Seq (DeepFunc that applies only the sequence-derived inputs), and DeepFunc_PPI (DeepFunc that applies only the PPI network–derived inputs) on the testing dataset.

Method	F_{max}	AvgPr	AvgRc	MCC	AUC
DeepGO_Seq	0.36	0.45	0.30	0.33	0.87
DeepGO	0.47	0.58	0.40	0.44	0.93
DeepFunc_Seq	0.54	**0.67**	0.46	0.50	0.91
DeepFunc_PPI	0.48	0.58	0.42	0.46	0.93
DeepFunc	**0.56**	**0.67**	**0.48**	**0.52**	**0.94**

Best results for each quality measure are highlighted in bold.

3.3. Ablation Study

We also dissect the DeepFunc model to investigate impact of the two types of its inputs: sequence derived and PPI network derived. We empirically compare predictions of the corresponding three version of DeepFunc: complete DeepFunc model, DeepFunc_Seq that applies only the sequence-derived inputs, and DeepFunc_PPI that uses only the PPI network–derived inputs. We contrast these predictions with the outputs produced by DeepGO and DeepGO_Seq that applies only the sequence-derived inputs (we were not able to implement the DeepGO_PPI version). The results produced by these five models on the testing dataset are compared in **Table 4**. We observe that DeepFunc outperforms all other considered models on all five metrics.

The comparison of the three versions of DeepFunc reveals that inclusion of each of the two inputs improves the resulting predictions. The removal of the sequence-derived inputs results in drop in AUC from 0.94 to 0.93 and in F_{max} from 0.56 to 0.48. The exclusion of the PPI network–derived features also has a strong negative impact. It lowers AUC from 0.94 to 0.91 and F_{max} from 0.56 to 0.54. The fact that combining the two input types together improves predictive performance suggests that these two inputs are complementary.

While Section 3.1 already compares DeepFunc and DeepGO, here we focus on the side-by-side comparison of their versions that utilize only the sequence-derived inputs. DeepGO primarily relies on a simple 3-mer-based representation of the input sequence while DeepFunc uses likely more informative features that encode information about domains, families, and motifs associated with the input protein chain. The higher quality of the DeepFunc's sequence-derived inputs results in a substantially higher predictive performance. The improvements are present across all five metrics, with AUC = 0.91 for DeepFunc_Seq versus 0.87 for DeepGO_Seq, $F_{max} = 0.54$ versus 0.36 and AUC = 0.50 versus 0.33. These results indicate that the domain/family/motif information is effective for the prediction of protein functions. We also compare DeepFunc_PPI with DeepGO. Table 4 shows that DeepFunc_PPI has a slight advantage although unlike DeepGO it does not use the sequence-derived input. The corresponding F_{max} and MCC for DeepFunc_PPI equal 0.48 and 0.46 versus 0.47 and 0.44 for DeepGO, respectively. This suggests that the PPI network and its encoding utilized by DeepFunc are better than the network used in DeepGO. In summary, DeepFunc effectively combines protein sequences and PPI networks, and extract higher-quality features for protein function prediction, which is the main factor behind the improvements offered by DeepFunc over the other deep learning-based predictor, DeepGO.

4. Discussion

We design, test, and comparatively assess a deep learning framework for protein function prediction, DeepFunc. Our method uses deep neural network to make accurate predictions from the protein sequence- and network-derived information. The DeepFunc combines topological features of PPI network and subsequence-based features concerning motifs, domains, and family assignments associated with the protein sequences. The topological features and protein sequence used in DeepFunc have been previously used individually or in combination with other features for the protein function prediction. They do not bias our model toward the GO terms any more than in the other published methods. The main contribution in our article is related to the use of the deep learning techniques to effectively represent the high-dimensional vector of the InterProScan-derived information and to combine the topological features extracted from the "enhanced" PPI network with this reduced InterProScan-derived information. These advances are responsible for the favorable predictive performance of DeepFunc.

Empirical tests show that DeepFunc secures comparable results on the two benchmark datasets: AUC = 0.94 and $F_{max} = 0.56$ on the testing dataset and AUC = 0.94 and $F_{max} = 0.54$ on the CAFA3 dataset. These tests demonstrate that DeepFunc outperforms the other deep learning-based solution, DeepGO, and predictions that rely on the sequence alignment with BLAST, including a challenging scenario where the test proteins share relatively low similarity to the training proteins. Comparison with three recently published methods (Deep GO, FFPred3, and GoFDR) on CAFA3 dataset reveals that DeepFunc obtains the highest values of F_{max} and AUC. The improvements over the second best result on this dataset are 0.94 versus 0.90 in AUC and 0.54 versus 0.52 in F_{max}. Overall, these empirical results suggest that DeepFunc provides the most accurate predictions.

The ablation study shows that extracting higher-quality features from protein sequences and PPI network that are utilized by DeepFunc contribute to its high-predictive performance. Moreover, detailed comparison of the two deep learning-based tools suggests that the sequence-derived inputs and PPI network used by DeepFunc are superior to the same type of inputs used by the DeepGO predictor.

As part of future work, we will consider inclusion of additional sources of sequence-derived information to study whether this can lead to further improvements in the predictive performance. Example of potentially useful inputs that were successfully used by some of the older predictors include co-expression data,[68] phylogenetic information,[69] and quantitative biophysical properties.[70]

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants (No. 61832019, No. 61622213, and No. 61728211), the 111 Project (No. B18059), and the Hunan Provincial Science and Technology Program (2018WK4001).

Conflict of Interest

The authors declare no conflict of interest.

Keywords

deep learning, functional linkages, protein domains, protein functions, protein–protein interactions, protein sequences

Received: January 12, 2019
Revised: March 18, 2019
Published online: May 27, 2019

[1] M. Li, W. Li, F. X. Wu, Y. Pan, J. Wang, *J. Theor. Biol.* **2018**, *447*, 65.
[2] B. Rekapalli, K. Wuichet, G. D. Peterson, I. B. Zhulin, *BMC Genomics* **2012**, *13*, 634.
[3] M. Li, X. Meng, R. Zheng, F. X. Wu, Y. Li, Y. Pan, J. Wang, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2017**. https://doi.org/10.1109/TCBB.2017.2749571
[4] E. Boutet, D. Lieberherr, M. Tognolli, M. Schneider, P. Bansal, A. J. Bridge, S. Poux, L. Bougueleret, I. Xenarios, *Plant Bioinformatics*, Springer, New York **2016**, 23.
[5] M. Costanzo, B. VanderSluis, E. N. Koch, A. Baryshnikova, C. Pons, G. Tan, W. Wang, M. Usaj, J. Hanchard, S. D. Lee, *Science* **2016**, *353*, aaf1420.
[6] G. Pandey, V. Kumar, M. Steinbach, *Twin Cities*, Department of Computer Science and Engineering, University of Minnesota, **2006**.
[7] S. Das, D. Lee, I. Sillitoe, N. L. Dawson, J. G. Lees, C. A. Orengo, *Bioinformatics* **2015**, *31*, 3460.
[8] X. Wang, D. Schroeder, D. Dobbs, V. Honavar, *Inf. Sci.* **2003**, *155*, 1.
[9] Q. Gong, W. Ning, W. Tian, *Methods* **2016**, *93*, 3.
[10] R. D. King, A. Karwath, A. Clare, L. Dehaspe, *Int. J. Genomics* **2000**, *1*, 283.

[11] D. Cozzetto, F. Minneci, H. Currant, D. T. Jones, *Sci. Rep.* **2016**, *6*, 31865.
[12] J. Q. Jiang, L. J. McQuay, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2012**, *9*, 1059.
[13] W. Peng, M. Li, L. Chen, L. Wang, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2017**, *14*, 360.
[14] M. Kirac, G. Ozsoyoglu, presented at Annual Int. Conf. on Research in Computational Molecular Biology, Singapore, March 30–April 2, **2008**.
[15] J. Hou, *New Approaches of Protein Function Prediction from Protein Interaction Networks*, Academic Press, London **2017**.
[16] C. D. Nguyen, K. J. Gardiner, K. J. Cios, *J. Biomed. Inf.* **2011**, *44*, 824.
[17] H. Rahmani, H. Blockeel, A. Bender, *JMLR* **2009**, *8*, 82.
[18] V. Gligorijević, M. Barot, R. Bonneau, *Bioinformatics* **2018**, *34*, 3873.
[19] J. Li, S. K. Halgamuge, C. I. Kells, S.-L. Tang, *BMC Bioinformatics* **2007**, *8*, S6.
[20] J. Konc, M. Hodošček, M. Ogrizek, J. T. Konc, D. Janežič, *PLoS Comput. Biol.* **2013**, *9*, e1003341.
[21] E. W. Stawiski, A. E. Baucom, S. C. Lohr, L. M. Gregoret, *Proc. Natl. Acad. Sci. U.S.A.* **2000**, *97*, 3954.
[22] C. Zhang, P. L. Freddolino, Y. Zhang, *Nucleic Acids Res.* **2017**, *45*, W291.
[23] H. A. Maghawry, M. G. Mostafa, T. F. Gharib, *J. Comput. Biol.* **2014**, *21*, 936.
[24] X.-L. Li, Y.-C. Tan, S.-K. Ng, *BMC Bioinf.* **2006**, *7*, S23.
[25] L. H. Tran, *JOACE* **2015**, *3*, 164.
[26] C. Cai, L. Han, Z. L. Ji, X. Chen, Y. Z. Chen, *Nucleic Acids Res.* **2003**, *31*, 3692.
[27] W. Peng, J. Wang, J. Cai, L. Chen, M. Li, F.-X. Wu, *BMC Syst. Biol.* **2014**, *8*, 35.
[28] M. Kulmanov, M. A. Khan, R. Hoehndorf, *Bioinformatics* **2017**, *34*, 660.
[29] R. Sharan, I. Ulitsky, R. Shamir, *Mol. Syst. Biol.* **2007**, *3*, 88.
[30] Ö. S. Saraç, V. Atalay, R. Cetin-Atalay, *PLoS One* **2010**, *5*, e12382.
[31] L. Wei, Y. Ding, R. Su, J. Tang, Q. Zou, *J. Parallel Distrib. Comput.* **2018**, *117*, 212.
[32] Q. Zou, P. Xing, L. Wei, B. Liu, *RNA* **2019**, *25*, 205.
[33] L. Wei, R. Su, B. Wang, X. Li, Q. Zou, X. Gao, *Neurocomputing* **2019**, *324*, 3.
[34] M. Zeng, M. Li, Z. Fei, F.-X. Wu, Y. Li, Y. Pan, presented at 2018 IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM), Madrid, Spain, December **2018**.
[35] M. Zeng, M. Li, Z. Fei, F. Wu, Y. Li, Y. Pan, J. Wang, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2019**. https://doi.org/10.1109/TCBB.2019.2897679
[36] S. Seo, M. Oh, Y. Park, S. Kim, *Bioinformatics* **2018**, *34*, i254.
[37] R. Vinayakumar, K. Soman, K. Naveenkumar, *bioRxiv* **2018**. https://doi.org/10.1101/414128
[38] A. Mitchell, H.-Y. Chang, L. Daugherty, M. Fraser, S. Hunter, R. Lopez, C. McAnulla, C. McMenamin, G. Nuka, S. Pesseat, *Nucleic Acids Res.* **2014**, *43*, D213.
[39] J. Huerta-Cepas, D. Szklarczyk, K. Forslund, H. Cook, D. Heller, M. C. Walter, T. Rattei, D. R. Mende, S. Sunagawa, M. Kuhn, *Nucleic Acids Res.* **2015**, *44*, D286.
[40] D. Szklarczyk, A. Franceschini, S. Wyder, K. Forslund, D. Heller, J. Huerta-Cepas, M. Simonovic, A. Roth, A. Santos, K. P. Tsafou, *Nucleic Acids Res.* **2014**, *43*, D447.
[41] B. Perozzi, R. Al-Rfou, S. Skiena, presented at Proc. of the 20th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, New York, August 24–27, **2014**.
[42] P. Radivojac, W. T. Clark, T. R. Oron, A. M. Schnoes, T. Wittkop, A. Sokolov, K. Graim, C. Funk, K. Verspoor, A. Ben-Hur, *Nat. Methods* **2013**, *10*, 221.
[43] Q. Zou, K. Qu, Y. Luo, D. Yin, Y. Ju, H. Tang, *Front. Genet.* **2018**, *9*, 515.
[44] C. Wang, L. Kurgan, *Briefings Bioinf.* **2018**. https://doi.org/10.1093/bib/bby069
[45] F. Meng, V. N. Uversky, L. Kurgan, *Cell. Mol. Life Sci.* **2017**, *74*, 3069.
[46] J. Zhang, L. Kurgan, *Briefings Bioinf.* **2018**, *19*, 821.
[47] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, D. J. Lipman, *Nucleic Acids Res.* **1997**, *25*, 3389.
[48] D. Cozzetto, D. W. Buchan, K. Bryson, D. T. Jones, *BMC Bioinformatics* **2013**, *14*, S1.
[49] R. Cao, J. Cheng, *Methods* **2016**, *93*, 84.
[50] A. Marchler-Bauer, M. K. Derbyshire, N. R. Gonzales, S. Lu, F. Chitsaz, L. Y. Geer, R. C. Geer, J. He, M. Gwadz, D. I. Hurwitz, *Nucleic Acids Res.* **2014**, *43*, D222.
[51] E. L. Sonnhammer, S. R. Eddy, R. Durbin, *Proteins: Struct., Funct., Bioinf.* **1997**, *28*, 405.
[52] I. Sillitoe, T. E. Lewis, A. Cuff, S. Das, P. Ashford, N. L. Dawson, N. Furnham, R. A. Laskowski, D. Lee, J. G. Lees, *Nucleic Acids Res.* **2014**, *43*, D376.
[53] D. A. de Lima Morais, H. Fang, O. J. Rackham, D. Wilson, R. Pethica, C. Chothia, J. Gough, *Nucleic Acids Res.* **2010**, *39*, D427.
[54] M. Li, P. Ni, X. Chen, J. Wang, F. Wu, Y. Pan, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2017**. https://doi.org/10.1109/TCBB.2017.2665482
[55] G. Li, M. Li, J. Wang, Y. Li, Y. Pan, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2018**. https://doi.org/10.1109/TCBB.2018.2889978
[56] M. Li, H. Gao, J. Wang, F. X. Wu, *Briefings Bioinf.* **2018**. https://doi.org/10.1093/bib/bby088
[57] M. Li, Z. Fei, M. Zeng, F. Wu, Y. Li, Y. Pan, J. Wang, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **2018**. https://doi.org/10.1109/TCBB.2018.2817488
[58] X. Qin, Y. Luo, N. Tang, G. Li, *Big Data Mining Analytics* **2018**, *1*, 75.
[59] M. Zeng, M. Li, Z. Fei, Y. Yu, Y. Pan, J. Wang, *Neurocomputing* **2019**, *324*, 43.
[60] A. Grover, J. Leskovec, presented at Proc. of the 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, San Francisco, CA, August 13–17, **2016**.
[61] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, presented at Proc. of the 24th Int. Conf. on World Wide Web, Florence, Italy, May 18–22, **2015**.
[62] M. Xie, T. Hwang, R. Kuang, *Bioinformatics* **2012**, *1*, 1.
[63] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, presented at 26th Int. Conf. on Neural Information Processing Systems, Lake Tahoe, NV, December 05–10, **2013**.
[64] A. Paszke, S. Gross, S. Chintala, G. Chanan, presented at 2017 Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, US, December 9, **2017**.
[65] S. Addou, R. Rentzsch, D. Lee, C. A. Orengo, *J. Mol. Biol.* **2009**, *387*, 416.
[66] M. J. Mizianty, X. Fan, J. Yan, E. Chalmers, C. Woloschuk, A. Joachimiak, L. Kurgan, *Acta Crystallogr., Sect. D: Biol. Crystallogr.* **2014**, *70*, 2781.
[67] R. Rentzsch, C. A. Orengo, *BMC Bioinformatics* **2013**, *14*, S5.
[68] M. N. Wass, G. Barton, M. J. Sternberg, *Nucleic Acids Res.* **2012**, *40*, W466.
[69] B. E. Engelhardt, M. I. Jordan, J. R. Srouji, S. E. Brenner, *Genome Res.* **2011**, *21*, 1969.
[70] D. Ofer, M. Linial, *Bioinformatics* **2015**, *31*, 3429.