molecules

MDPI

Article

ProLanGO: Protein Function Prediction Using Neural Machine Translation Based on a Recurrent Neural Network

Renzhi Cao [1,*], Colton Freitas [1], Leong Chan [2], Miao Sun [3], Haiqing Jiang [4] and Zhangxin Chen [5,*]

1 Department of Computer Science, Pacific Lutheran University, Tacoma, WA 98447, USA; caora@plu.edu
2 School of Business, Pacific Lutheran University, Tacoma, WA 98447, USA; chanla@plu.edu
3 Baidu Inc. 1195 Bordeaux Dr, Sunnyvale, CA 94089, USA; miaosunwork@gmail.com
4 Hiretual Inc., San Jose, CA 95131, USA; stevenjiang@hiretual.com
5 School of electronic engineering, University of Electronic Science and Technology of China, Chengdu 610051, China
* Correspondence: caora@plu.edu (R.C.); zhangxinchen@uestc.edu.cn. (Z.C.); Tel.: +1-253-535-7409 (R.C.)

Received: 30 August 2017; Accepted: 11 October 2017; Published: 17 October 2017

Abstract: With the development of next generation sequencing techniques, it is fast and cheap to determine protein sequences but relatively slow and expensive to extract useful information from protein sequences because of limitations of traditional biological experimental techniques. Protein function prediction has been a long standing challenge to fill the gap between the huge amount of protein sequences and the known function. In this paper, we propose a novel method to convert the protein function problem into a language translation problem by the new proposed protein sequence language "ProLan" to the protein function language "GOLan", and build a neural machine translation model based on recurrent neural networks to translate "ProLan" language to "GOLan" language. We blindly tested our method by attending the latest third Critical Assessment of Function Annotation (CAFA 3) in 2016, and also evaluate the performance of our methods on selected proteins whose function was released after CAFA competition. The good performance on the training and testing datasets demonstrates that our new proposed method is a promising direction for protein function prediction. In summary, we first time propose a method which converts the protein function prediction problem to a language translation problem and applies a neural machine translation model for protein function prediction.

Keywords: protein function prediction; neural machine translation; machine learning

1. Introduction

With the wide application of next generation sequencing techniques, it takes a short time with low cost to generate millions of protein sequences [1]. Understanding the properties (e.g., protein function, protein structure) of each protein sequence becomes an urgent task, which not only helps us better understanding their role in our life, but more importantly their potential biomedical and pharmaceutical applications, such as drug discovery [2,3]. The traditional biological experimental methods to determine a protein's properties (e.g., protein function and structure) can be very slow and also resource-demanding [3,4]. Moreover, sometimes it has the limitation that may not faithfully reflect the protein's activity in vivo [3,5]. Due to these limitations, the computation method that could accurately and quickly predict protein function from its sequence is greatly desired. The computation protein function prediction method can be used to fill the gap between the large amount of sequence data and the unknown properties of these proteins.

Protein function prediction is usually treated as a multi-label classification problem. Researchers have tried different computation methods in the last few decades for this problem [6–12]. In general, the following methods are used for protein function prediction.

The first and the most widely used method for protein function prediction is the Basic Local Alignment Search Tool (BLAST) [13] used to search the query sequence against the existing protein databases, which contain experimentally determined protein function information, and then use these homologous proteins' function information for the function prediction of the query sequence. For example, the methods of Gotcha [14], OntoBlast [15], and Goblet [16]. Except for using the BLAST tool, some methods use the tool PSI-BLAST [13] to find the remote homologous, such as the PFP method [17].

The second includes network based methods. Most methods in this category use protein–protein interaction networks for protein function prediction based on the assumption that interacted proteins share similar functions [10,18–23]. Instead of protein–protein interaction networks, some methods use other kind of networks for protein function prediction, such as gene–gene interaction network and domain co-occurance networks [3,10,24].

The third is other information based methods. Some methods use protein structure or microarray gene expression data [25–28] or combinations of different resources for protein function prediction [29–32].

The most challenging method is to directly use protein sequence without searching any database or any other resource for protein function prediction. For this kind of methods, the machine learning techniques are usually used to predict protein function from scratch. Currently most machine learning methods use features generated from the protein sequences for model training, and use that model to classify a number of function categories for an input sequence [33,34]. The most commonly used features in these methods are protein sequence [35,36], protein secondary structure, hydrophobicity, subcellular location, solvent accessibility, etc. For example, PANNZER and MS-kNN use the machine learning method k-nearest neighbour to predict protein function [37]. SMISS uses an apriori algorithm to mine the association rules for protein function prediction [3], whereas methods like SVMProt use the machine learning method SVM for protein function prediction [38–40], and some methods use naïve Bayes classifiers [41,42]. Except for the protein sequence, most other features are usually predicted from the protein sequence, so more errors could be involved for these features and also for the final protein function prediction.

The latest state-of-the-art machine learning method—deep learning which uses multiple layers representation and abstraction of data has drastically improved the accuracy of speech-recognition, visual object recognition, language translation, protein structure prediction, etc. [43–48]. For example, Google developed its neural machine translation system based on Neural Machine Translation (NMT) in November 2016 to increase accuracy in Google Translate [49]. It would be interesting to apply these latest machine learning methods for the protein function prediction problem. Based on our knowledge, currently there are very few methods that only use protein sequence without any additional features. Researchers have tried to derive more features from protein sequence, and applied machine learning techniques for the protein function prediction problem [50], but most methods have additional features combined with protein sequence to prediction protein functions [3,37]. The reason is because protein function prediction from sequence only is really challenging, there is no such kind of method that achieves or is even close to the state-of-the-art performance. In addition, there is no method that applies the state-of-the-art machine learning method Neural Machine Translation (NMT) for the protein function prediction problem. In this work, we developed a novel method called ProLanGO, which first converts the protein sequence and protein function into the "ProLan" and "GOLan" language, respectively, and then builds the NMT model based on recurrent neural network to translate "ProLan" to "GOLan" for the protein function prediction. Our new proposed method based on the ProLanGO model provides a new way of predicting protein functions.

Despite the protein function prediction methods, how to test these methods is also important. There is a worldwide experiment—the Critical Assessment of Function Annotation (CAFA,

http://biofunctionprediction.org/) [2,51], which is designed to provide a large-scale unbiased benchmarking of different protein function prediction methods to test their ability to predict protein function on the same set of proteins within a specific time frame [2,51]. There are three stages for CAFA: prediction phase, the organizer releases protein sequences with unknown or incomplete function for collecting the submission of predictions (usually about four months); target accumulation phase, waiting for the biological experiment to determine protein's function (usually between 6 and 12 months); and analysis phase, the organizer evaluates and ranks the performance of different methods based on their prediction on proteins whose function is known during the target accumulation phase. Until now, three CAFA experiments have been conducted. The first CAFA (CAFA1) showed the good performance of applying a machine learning method to integrate multiple sequence hits and data types for protein function prediction. We have attended the third CAFA (CAFA3) in 2016 to blindly test our method on a large scale of protein sequences to evaluate the performance of our model.

The rest of the paper is organized as follows. In Section 2, we describe some results on training and testing datasets. In Section 3, we describe the data and detailed method to build and test our method. In Section 4, we summarize our work and discuss the future directions.

2. Results and Discussions

2.1. Performance during Training of Neural Machine Translation Model

The overall flowchart of our method is shown in Figure 1. In the training step, we use 419,192 protein sequences to train the ProLanGO model and another 104,798 protein sequences as testing data to evaluate the performance of our model. The perplexity has been used to evaluate the performance of our language model ProLanGO on training datasets and testing datasets. Figure 2 shows how the perplexity changes on training datasets and performance on testing datasets based on buckets. The buckets are techniques used in RNN to efficiently handle protein sequences with different length, which may not influence the performance too much. For our method, we select the bucket size based on our analysis of protein sequences in the protein database so that it could handle most protein sequences efficiently. We ran 300,000 steps for training, and the learning rate in the range of [0–1]. As we can see from the picture, the perplexity on training datasets is stable after 10,000 steps, and the perplexity for each buckets on testing datasets is still fluctuating. The result of first 10,000 steps is not included because RNN is not stable with few training steps and the perplexity is huge. Based on the performance for every 200 steps, we finally select the model at step 78,400 (learning rate is 0.5472) and 95,200 (learning rate is 0.5639) for the normal and extended neural machine translation model respectively, and combine these two models as our final ProLanGO model for prediction.

2.2. Performance on Validation Dataset

We use the UnitProtKB database on 1 June 2016 to train and test our model and blindly tested on around 100,000 protein sequences from CAFA3. Some of these protein sequences' function has been released later in UnitProtKB database, and we evaluate the performance of our model on those proteins. We have not performed a homology search against the database, so part of the proteins may have homologies in the database of 1 June 2016, and it may influence the evaluation. However, this problem would be solved after CAFA officially release their evaluation later. In total, we find 332 proteins that are added after 1 June 2016. We propagate both real and predicted GO terms to the root of the Gene Ontology Directed Acyclic tree structure as described in method section, and the evaluation of precision and recall is described in method section.

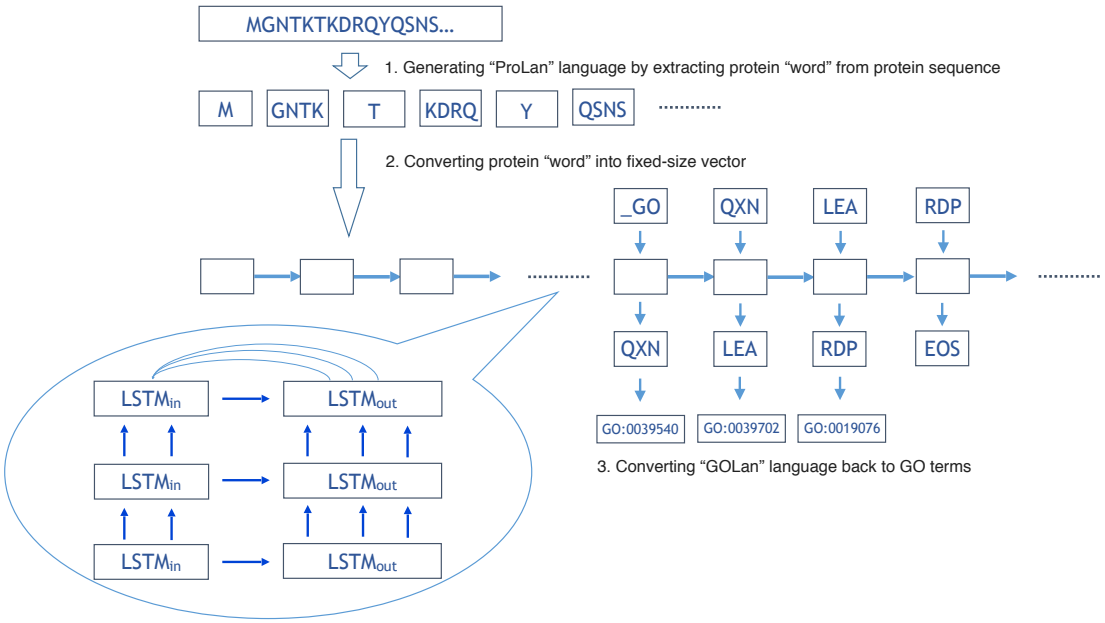

Figure 1. The flowchart of our method for protein function prediction. Three layers of RNN is used for the NMT model.

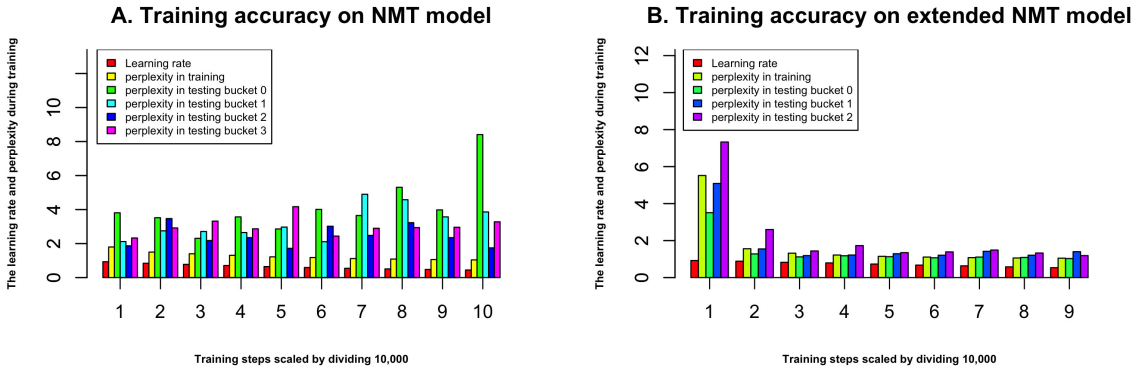

Figure 2. (**A**) The accuracy during training process for NMT model; (**B**) The accuracy during training process for extended NMT model. The x-axis represents the steps during the training, we scaled it by dividing 10,000, and the first 10,000 steps are not shown in the figure since the perplexity is very big.

Figure 3 demonstrates that the NMT model achieves the relatively lower recall, and the probabilistic based model achieves relatively higher recall. The reason is because NMT model makes six predictions for most proteins, and the maximum number of predictions is 10, however, the probabilistic based model makes more predictions than NMT. The extended NMT model gets relatively higher performance compared to NMT model. The ProLanGO model, which is a combination of NMT and extended NMT model, achieves better performance on both recall and precision. This result shows that our ProLanGO model has potential to accurately predict protein function from sequences.

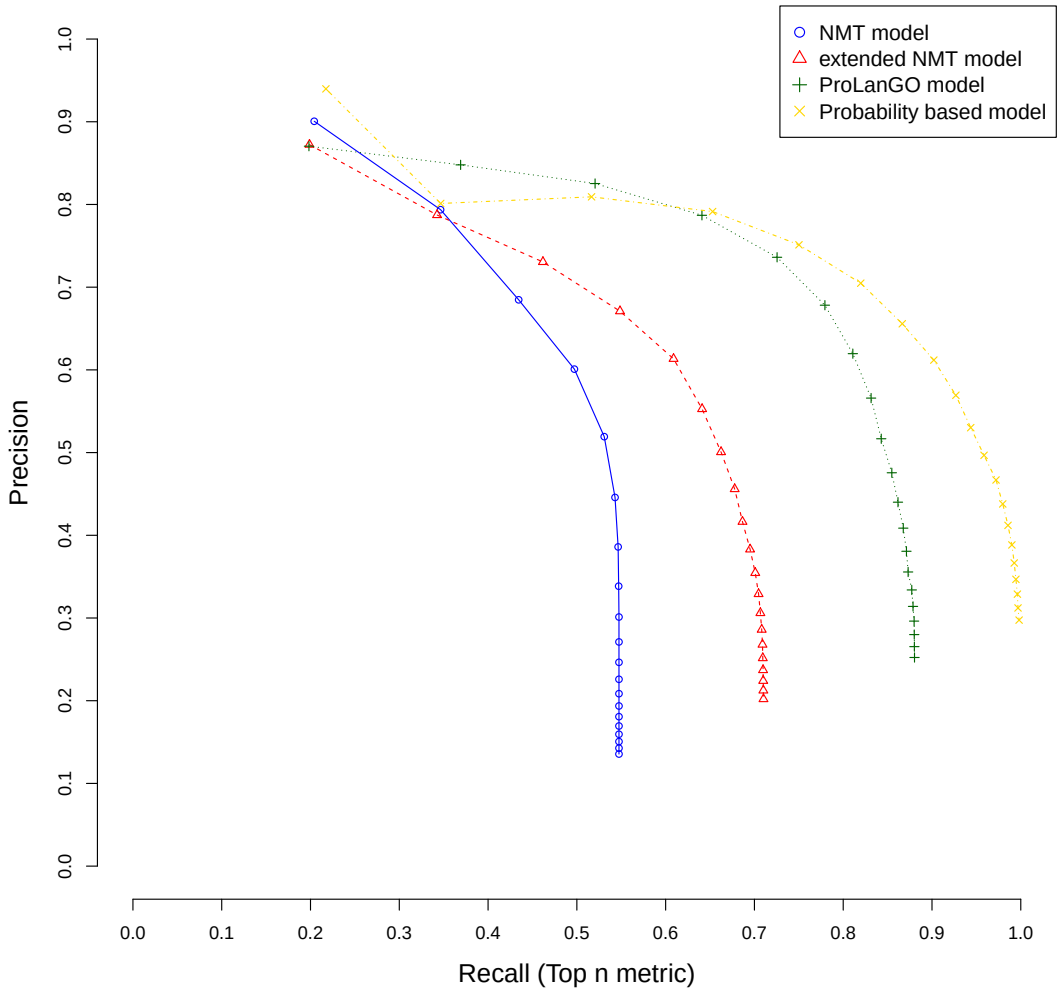

Figure 3. Precision and recall of different methods on selected proteins. The top n metric is used, and n is set to be 20. The area under curve (AUC) for NMT, extended NMT, ProLanGO, and Probability based model is 0.286, 0.305, 0.333, and 0.390, respectively.

2.3. Comparison with Other Selected Methods on CAFA3

CAFA3 officially releases true functions for part of CAFA3 targets around 5 June 2017. In total, 674 targets' BPO (Biological Process Ontology) function, 539 targets' CCO (Cellular Component Ontology), and 684 target's MFO (Molecular Function Ontology) function are released. In order to evaluate the effectiveness of our ProLanGO model in CAFA3 blind experiment, we compare the performance with four state-of-the-art protein function prediction methods in CAFA2: the SMISS model [3,52], PANNZER model [53], FANN-GO model [54], and DeepGO model [55]. The FANN-GO and DeepGO model are both sequence-based protein function prediction methods, PANNZER is a k-nearest neighbour method, and the SMISS model contains both sequence-based and network-based predictions. We calculate the average GO similarities of all GO pairs between predicted and true GO for each targets. The GO similarity is defined as the number of shared GO inner nodes divided by the largest number of GO inner nodes when propagating predicted and true GO term in the Gene Ontology Directed Acyclic tree. Figures 4 and 5 describe the performance of the ProLanGO model and three different scores from the SMISS model, and also PANNZER model. Most protein function prediction methods search against protein function databases, and very few of them try to predict protein function from sequence only without using additional database information. The SEQ score from SMISS model is only based on protein sequence, and the NETscore is based on protein–protein and gene–gene interaction networks information. From this figure, we can see that our model achieves

a better performance, on average, compared to the SEQ score from SMISS for most of the different thresholds. At the same time, I also want to mention that the gap between sequence based protein function prediction and methods using other information is still big. The MIS score from the SMISS model only uses information from PSI-BLAST searches, and we can find out that the MIS score and PANNZER have better performance comparing to our method. We mention that the average GO similarity score between predicted and true for PANNZER is around 0 for a threshold more than 0.9—the reason could be that our target protein sequences from CAFA are not included in protein sequence databases at the time of our submission. This issue would be solved later when these protein sequences are annotated and added to the database. The result could be different than what we have got after PANNZER updates their database. Our method achieves a similar performance based on Threshold metrics compared to the DeepGO model, while DeepGO performs better than our method for thresholds between 0.10 and 0.80. The FANN-GO model performs better than our method and DeepGO for threshold larger than 0.10. A similar pattern has been observed based on Top n metrics in Figure 5. However, our model doesn't do especially well for the top one GO term prediction. This is probably because of the lack of ranking GO terms predictions in RNN output. Some additional information might be used in future to improve the ranking of GO terms in our model. Overall, our model shows better performance than the SEQ score in SMISS model, but it also shows that our method is not performing well for the prediction of top GO terms, which need to be improved in future. Supplementary Table S4 describes additional resources for protein function prediction methods that we have tried but not succeed because of the availability of these methods.

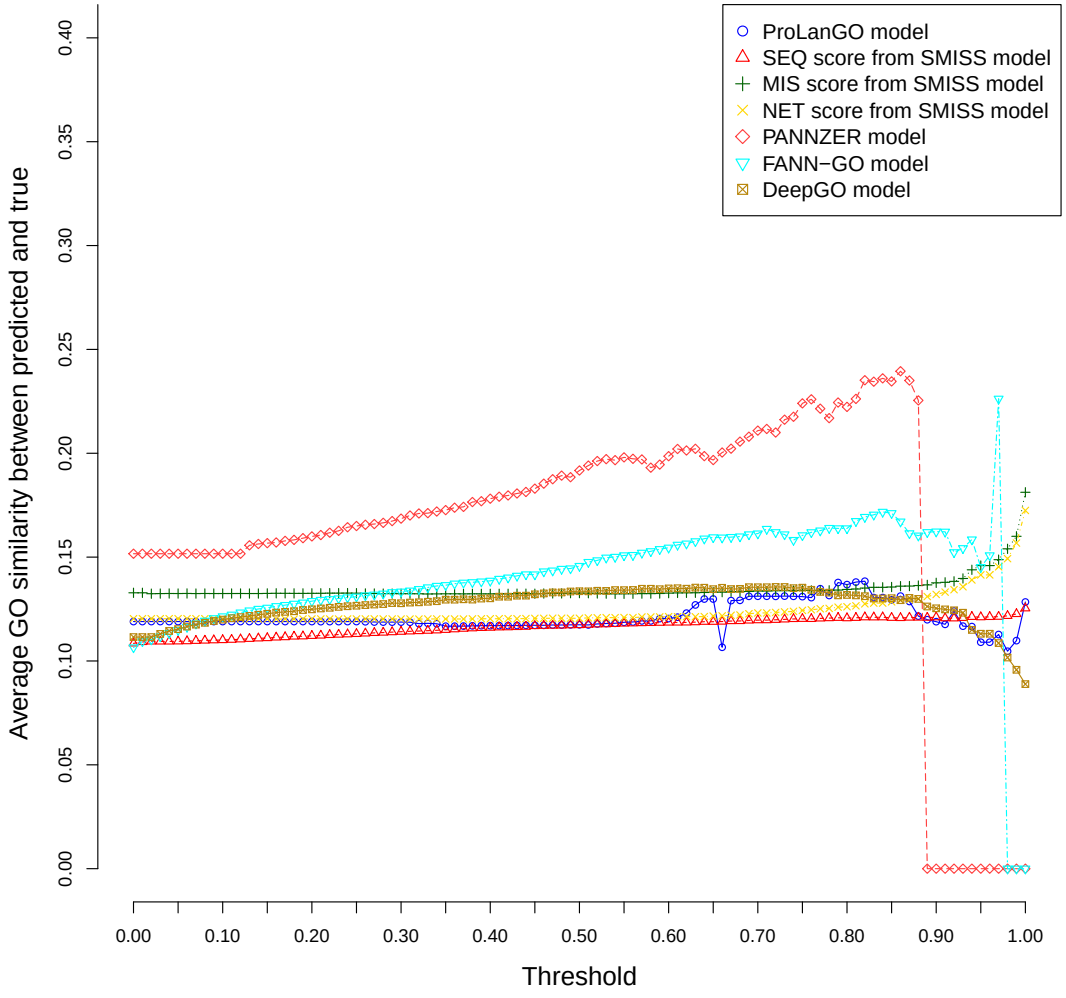

Figure 4. Average GO similarities of ProLanGO and other methods on threshold metric.

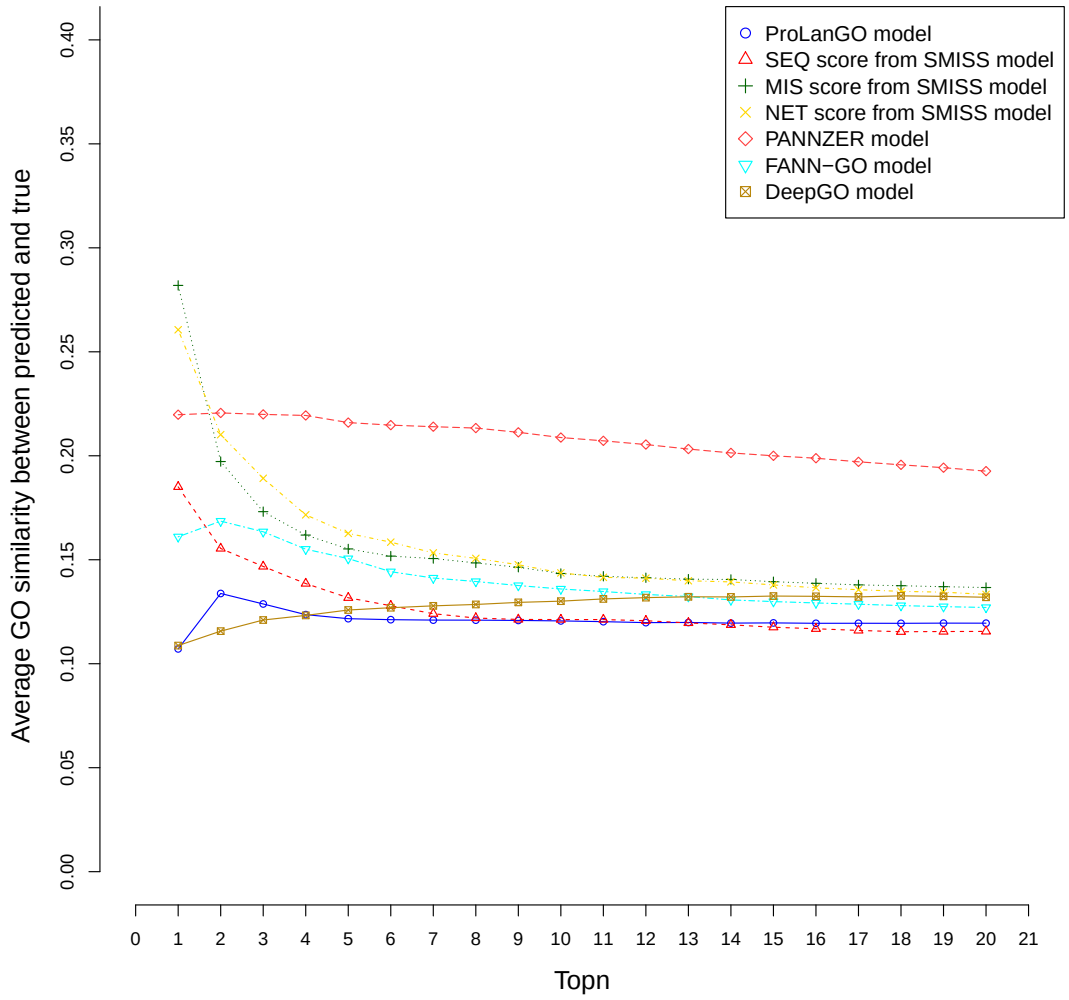

Figure 5. Average GO similarities of ProLanGO and other methods on top n metric.

3. Methods

In this paper, we construct the method ProLanGO which uses a neural machine translation model based on a recurrent neural network to predict protein function from protein sequence, and also we construct a baseline method which uses probability theory as a comparison.

3.1. Data Preparation

The training data of our model comes from UniProtKB knowledge database [56] with the version of 1 June 2016. We only consider Gene Ontology (GO) terms ID which are valid from the date of 1 June 2016. In total, 523,990 protein sequences and 42,819 GO terms are used to train our model. We blindly test our method on CAFA3, which includes 130,787 protein sequences in 23 species.

3.2. Understanding Protein Sequence as a Language

Our assumption is that protein sequences are similar to human language; each protein sequence can be represented as a sentence consisting of protein "words". Proteins are composed of amino acids, and we use 20 letters to represent each of 20 standard amino acids found in protein. Each protein sequence is a string of these 20 letters, and we use to represent the set of these 20 letters. We let A^k be k-mers or fragment with length k, which is a k-th cartesian power of A or a string of k letters from A. The k-mers are considered as protein "word" [57–60]. The following algorithm has been used to divide protein sequence into a set of k-mers or protein "word":

Step 1. Scan the whole training dataset - UniProtKB knowledge database to get a protein "word" database, which includes all k-mers whose frequency f^k is larger than 1,000, where $k \in [3, 5]$.

Step 2. For each protein sequence s, we divide it into a set of protein "word" based on Step 1. In this step, the longer protein "word" are preferred for matching, and each amino acid can only belong to one protein "word". If t continuous amino acids cannot match with any fragment from Step 1, we let these continuous amino acids be a new protein "word" A^t, $t \in [1, \infty)$.

We ranked all k-mers based on its frequency in training dataset, and the frequency of k-mers ranked at 1000th is 47,802, 7167, 1232, 650, 565 for each k of three, four, five, six, and seven respectively. We did not evaluate performance of k-mers with k larger than five, because low frequency in training dataset for these large k could be less reliable. We have also tried to add six-mers and seven-mers (the best loss on testing dataset is 8.62, 9.26, 8.59 for $k \in$ [3–5], [3–6], and [3–7] respectively), and no significant improvement has been found in our experiment. More details of our experiments are in Supplementary Tables S1–S3.

3.3. Understanding Gene Ontology Terms as a Language

The traditional protein function prediction methods directly predict Gene Ontology (GO) terms for a protein sequence. The GO terms are represented by GO term ID using a seven digit number (e.g., GO:0000001), and these GO terms are used to form a directed acyclic tree structure based on the relationship between each other (e.g., some GO terms are part-of other GO terms). There are three trees to represent the GO terms in each of the three biological categories (Biology Process, Cellular Component, Molecular Function). The traditional GO term ID does not considered the relationship between each GO term (e.g., GO:0015772 is a GO:2001088, but it is not reflected from the GO ID). Moreover, it needs seven digits to describe a GO term. In this paper, we describe a new way to encode GO terms, and consider the encoded GO terms as a new language to describe functions of proteins.

We do a depth-first search (DFS) on Biological Process, Cellular Component, and Molecular Function tree respectively, and assign the 26-base Alphabet number to each GO term as a new ID (Alphabet ID). For example, GO:0008150 is the root of Biological Process (BP) tree, and there are 28,678 GO terms in BP tree, so GO:0008150 is the number 28,678th node to be visited by DFS, and we can convert 10-base number 28,678 to 26-base Alphabet ID: BQKZ. In theory, a four letter Alphabet ID can represent 26^4 GO terms, which is more than the current total number of GO terms. In summary, the new Alphabet ID has a maximum length of four, and considers the relationship of GO terms in the Gene Ontology tree. For a protein with multiple functions (GO terms), we simply convert each GO term to a Alphabet ID, and the order of these Alphabet IDs is based on the order of the respective GO term from the UniProtKB knowledge database (See Figure 1).

3.4. ProLanGO Model by Neural Machine Translation Based on Recurrent Neural Network

In the previous sections, we convert protein sequences into a new language based on the protein "word" generated from UniProtKB database; we call it the "ProLan" language. Also, the functions of each protein are converted into a new language based on the Alphabet ID, we let it be the "GOLan" language (See Figure 1). We now convert the protein function prediction problem into a language translation problem. The neural machine translation model based on two recurrent neural networks is used to solve the language translation problem [61,62]. The first recurrent neural network encodes a sentence in "ProLan" into vectors of fixed length, and the second recurrent neural network decodes the representation into a sentence of "GOLan". The encoder and decoder RNN are trained together to maximize the conditional probability of "GOLan" sentence given "ProLan" sentence. In general, there are two multi-label classification methods [63]: problem transformation methods and algorithm adaptation methods. Our method based on RNN is one type of problem transformation method. The RNN could be considered as a chain of repeating modules of a neural network, and each neural network is used to handle a single-label classification problem by outputting a "word" in a

"GOLan" sentence. In another word, our method is suitable for protein function prediction (multi-label classification problem) by transforming it into several single-label classification problems and solving each of them with a module of a neural network in RNN. In addition, the attention mechanism [64] is used in the system to align and translate jointly, which would be helpful to solve the problem when the "ProLan" sentence is relatively long, because it would be difficult for encoder RNN to convert a long "ProLan" sentence into a fixed length of vector.

There are in total 420,127 different protein "words" in "ProLan", and 25,160 different Alphabet IDs in "GOLan". In order to simplify the training and decoding complexity with this big "ProLan" and "GOLan" data, we use sampled softmax [65] based on importance sampling which samples on a small subset of the target "GOLan" Alphabet ID for decoding. In addition, the bucketing and padding technique is used to efficiently handle "ProLan" and "GOLan" with different length L1 and L2. We use a number of buckets and a special symbol PAD to pad each "ProLan" and "GOLan" into the buckets. The following buckets have been used:

$$\text{buckets} = [(64, 3), (109, 4), (164, 6), (300, 10)].$$

The buckets are used to improve the efficiency of RNN. For example, let's assume there are three protein "words" in a ProLan sentence and four outputs in a GOLan sentence, then the length for input (L1) is three, and the length for output (L2) is four. Even though the input length L1 can fit the first bucket (64, 3), the output length L2 is larger than the output limit three for the first bucket. In this case, we finally use the second bucket (109, 4), and add special symbol PAD to fill the rest so that the input length L1 would be 109, and output length L2 would be four. Figure 1 illustrates the overall flowchart of our method. In order to decide the upper and lower bounds for buckets, we analyze the number of input "words" and outputs from ProLan sentences and GOLan sentences in the training data sets. This bounds number is selected so that there are a similar number of data falling into each bucket for training data sets.

It should be mentioned here that the maximum number of outputs is 10, based on the bucketing and padding technique, which means we can predict a maximum of 10 GO term functions for each protein sequence. Some proteins may contain more than 10 functions and in order to solve this problem, we add an extended NMT model. For this extended NMT model, we still keep the same "ProLan" language, and change a little bit of the "GOLan" language. For the GO terms of the protein in the training dataset, we add all descendants of each of the GO terms in the directed acyclic tree to extend the sentence in "GOLan" language. The same NMT framework is used except the following buckets are used instead:

$$\text{buckets} = [(64, 25), (164, 50), (250, 60)]$$

In this extended NMT model, the maximum number of function prediction could be 60, which is big enough for most proteins. As we could imagine, the new added descendants could be less accurate especially when the descendant is much deeper from the real GO term in the tree structure. A decreased rate is added to each new added descendant GO term, so that the one far away from the original true GO term would have lower score.

Finally, the ProLanGO model is a combination of the NMT and extended NMT model by giving different weights for each model based on their performance on training data.

The prediction part is straightforward, for each protein sequence, we first convert it to a "ProLan" sentence, and then applied the trained neural machine translation model to translate the "ProLan" sentences to "GOLan" sentences and extended "GOLan" sentences. Finally, we apply the ProLanGO model to combine these predicted "GOLan" sentences and convert these "GOLan" sentences back to GO terms by a simple mapping.

3.5. Baseline Method on Probability Theory

In order to benchmark our neural machine translation model for protein function prediction, we create a benchmark method which applies simple probability theory to make the protein function

prediction. Let x be the fragments vectors from the "ProLan" space, and y be the Alphabet ID for all possible protein functions. Here, x is a vector [x_1,x_2,...,x_n], and the n is decided by the total number of fragments generated for each protein sequence. Here, y is a vector [y_1, y_2, ..., y_m] and m is the total number GO terms. We use the following probability theory to calculate the probability of each y_j given the x generated for each protein sequence:

$$P(x_1, x_2, ..., x_n) = 1 - \prod_{i=1}^{n}(1 - P(y_j|x_i))$$

$P(y_j \mid x_i)$ is calculated from the UniProtKB knowledge database by counting the frequency of y_j in all proteins with x_i

3.6. Evaluation Method

We evaluate the performance of different methods based on precision and recall of top n predictions ranked by the confidence score of predictions. Here the n is in the range of 1 and 10. The precision and recall are calculated by the following formula:

$$Precision = \frac{total number of corrected predictions}{total number of GO predictions}$$

$$Recall = \frac{total number of corrected predictions}{total number of true GO terms}$$

The true GO terms are determined by biological experiment for each protein and here we simply use the UnitProtKB database to extract the true GO terms for each protein. We propagate all true and predicted GO terms to the root of Gene Ontology Directed Acyclic tree structure, and any propagated GO term predictions that exist in the path of true GO terms to the root are considered to be correct. The top n metric is used for the evaluation, which pick n GO term predictions based on the confidence score to calculate the precision and recall.

4. Conclusions

In this paper, we propose a novel language model ProLanGO for the protein function prediction problem. We first convert protein sequences into a language space "ProGO" based on the frequency of k-mers on around 500,000 protein sequences, and also encode Gene Ontology terms into a language space "LanGO". It uses at most four letters to represent a GO term, while the relationship of these GO terms in Gene Ontology Directed Acyclic Graph is considered during the translation. In addition, we convert the protein function prediction problem to a language translation problem which is on the new proposed language space, and apply the state-of-the-art Neural Machine Translation Model based on the recurrent neural network to predict protein function. We evaluate the performance of our ProLanGO model and also compare it with another probability based model proposed by us. The result shows that our ProLanGO model (Combination of NMT and extended NMT model) achieves better performance compared to the NMT and extended NMT model on both recall and precision, and similar performance are achieved compared to the probability based model. We also compare with four other state-of-the-art protein function prediction methods, and our methods show better performance than the SEQ score from SMISS model, but there is still a big gap between our method and the top homologous based method. There are 332 proteins selected by us for the evaluation, but more proteins with real functions would be released later , and the official evaluation result from CAFA will also be released. We could rigorously evaluate our method at that time and compare our method with other protein function prediction methods.

In general, we propose a new method for protein function prediction. We compare our method with four state-of-the-art protein function prediction methods. The good performance of our model ProLanGO demonstrates the potential application for protein function prediction problem. There are a lot of improvements that can be done in future. For example, the cross-validation can be included in the training part to improve the reliability of the ProLanGO model. Also, in the Prolan language, we use the frequency of k-mers while the k in the range of three and five is used, the accuracy could

be improved by a larger k. In addition, instead of using the frequency of k-mers, a more biologically fragment could be used to divide the protein sequences, such as using a fragment of binding site for a protein. For the bucketing technique, different sizes for each bucket could be tested to additionally improve the performance. The GO terms predicted by our model could be ranked by using other information. Finally, it will be very interesting to see how the order of these k-mers improve the performance.

Supplementary Materials: Supplementary materials are available online. Table S1: The loss on training, testing dataset and training steps for k in range of 3 and 5, Table S2: The loss on training, testing dataset and training steps for k in range of 3 and 6, Table S3: The loss on training, testing dataset and training steps for k in range of 3 and 7, Table S4: Reference papers of protein function prediction methods that are not used in this manuscript.

Acknowledgments: This work was supported by the National Natural Science Foundation of China under grant 61201273 and Natural Sciences Summer Undergraduate Research Program by Pacific Lutheran University.

Author Contributions: R.C. and Z.C design and build the system, R.C., C.F., and M.S. carries out the experiments, R.C. builds the server and install programs, R.C, C.F., L.C., M.S., H.J. and Z.C. write the manuscript and give suggestions.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Cao, R.; Bhattacharya, D.; Hou, J.; Cheng, J. DeepQA: Improving the estimation of single protein model quality with deep belief networks. *BMC Bioinform.* **2016**, *17*, 495.
2. Radivojac, P.; Clark, W.T.; Oron, T.R.; Schnoes, A.M.; Wittkop, T.; Sokolov, A.; Graim, K.; Funk, C.; Verspoor, K.; Ben-Hur, A.; et al. A large-scale evaluation of computational protein function prediction. *Nat. Methods* **2013**, *10*, 221–227.
3. Cao, R.; Cheng, J. Integrated protein function prediction by mining function associations, sequences, and protein–protein and gene–gene interaction networks. *Methods* **2016**, *93*, 84–91.
4. Liolios, K.; Chen, I.M.A.; Mavromatis, K.; Tavernarakis, N.; Hugenholtz, P.; Markowitz, V.M.; Kyrpides, N.C. The Genomes On Line Database (GOLD) in 2009: Status of genomic and metagenomic projects and their associated metadata. *Nucleic Acids Res.* **2009**, *38*, D346–D354.
5. Ashburner, M.; Ball, C.A.; Blake, J.A.; Botstein, D.; Butler, H.; Cherry, J.M.; Davis, A.P.; Dolinski, K.; Dwight, S.S.; Eppig, J.T.; et al. Gene Ontology: Tool for the unification of biology. *Nat. Genet.* **2000**, *25*, 25.
6. Rost, B.; Liu, J.; Nair, R.; Wrzeszczynski, K.O.; Ofran, Y. Automatic prediction of protein function. *Cell. Mol. Life Sci.* **2003**, *60*, 2637–2650.
7. Watson, J.D.; Laskowski, R.A.; Thornton, J.M. Predicting protein function from sequence and structural data. *Curr. Opin. Struct. Biol.* **2005**, *15*, 275–284.
8. Friedberg, I. Automated protein function prediction—The genomic challenge. *Brief. Bioinformat.* **2006**, *7*, 225–242.
9. Lee, D.; Redfern, O.; Orengo, C. Predicting protein function from sequence and structure. *Nat. Res. Mol. Cell Biol.* **2007**, *8*, 995.
10. Wang, Z.; Zhang, X.C.; Le, M.H.; Xu, D.; Stacey, G.; Cheng, J. A protein domain co-occurrence network approach for predicting protein function and inferring species phylogeny. *PLoS ONE* **2011**, *6*, e17906.
11. Rentzsch, R.; Orengo, C.A. Protein function prediction—The power of multiplicity. *Trends Biotechnol.* **2009**, *27*, 210–219.
12. Wan, S.; Duan, Y.; Zou, Q. HPSLPred: An ensemble multi-label classifier for human protein subcellular location prediction with imbalanced source. *Proteomics* **2017**, *17*.
13. Altschul, S.F.; Madden, T.L.; Schäffer, A.A.; Zhang, J.; Zhang, Z.; Miller, W.; Lipman, D.J. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res.* **1997**, *25*, 3389–3402.
14. Martin, D.M.; Berriman, M.; Barton, G.J. GOtcha: A new method for prediction of protein function assessed by the annotation of seven genomes. *BMC Bioinform.* **2004**, *5*, 178.
15. Zehetner, G. OntoBlast function: From sequence similarities directly to potential functional annotations by ontology terms. *Nucleic Acids Res.* **2003**, *31*, 3799–3803.

16. Groth, D.; Lehrach, H.; Hennig, S. GOblet: A platform for Gene Ontology annotation of anonymous sequence data. *Nucleic Acids Res.* **2004**, *32*, W313–W317.
17. Hawkins, T.; Chitale, M.; Luban, S.; Kihara, D. PFP: Automated prediction of gene ontology functional annotations with confidence scores using protein sequence data. *Proteins: Struct. Funct. Bioinform.* **2009**, *74*, 566–582.
18. Deng, M.; Zhang, K.; Mehta, S.; Chen, T.; Sun, F. Prediction of protein function using protein–protein interaction data. *J. Comput. Biol.* **2003**, *10*, 947–960.
19. Letovsky, S.; Kasif, S. Predicting protein function from protein/protein interaction data: A probabilistic approach. *Bioinformatics* **2003**, *19*, i197–i204.
20. Vazquez, A.; Flammini, A.; Maritan, A.; Vespignani, A. Global protein function prediction in protein-protein interaction networks. *Nat. Biotechnol.* **2003**, 697–770. doi:10.1038/nbt825.
21. Hishigaki, H.; Nakai, K.; Ono, T.; Tanigami, A.; Takagi, T. Assessment of prediction accuracy of protein function from protein–protein interaction data. *Yeast* **2001**, *18*, 523–531.
22. Chua, H.N.; Sung, W.K.; Wong, L. Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions. *Bioinformatics* **2006**, *22*, 1623–1630.
23. Zeng, X.; Zhang, X.; Zou, Q. Integrative approaches for predicting microRNA function and prioritizing disease-related microRNA using biological interaction networks. *Brief. Bioinform.* **2015**, *17*, 193–203.
24. Cao, R.; Cheng, J. Deciphering the association between gene function and spatial gene-gene interactions in 3D human genome conformation. *BMC Genom.* **2015**, *16*, 880.
25. Pal, D.; Eisenberg, D. Inference of protein function from protein structure. *Structure* **2005**, *13*, 121–130.
26. Pazos, F.; Sternberg, M.J. Automated prediction of protein function and detection of functional sites from structure. *Proc. Natl. Acad. Sci. USA* **2004**, *101*, 14754–14759.
27. Laskowski, R.A.; Watson, J.D.; Thornton, J.M. Protein function prediction using local 3D templates. *J. Mol. Biol.* **2005**, *351*, 614–626.
28. Huttenhower, C.; Hibbs, M.; Myers, C.; Troyanskaya, O.G. A scalable method for integration and functional analysis of multiple microarray datasets. *Bioinformatics* **2006**, *22*, 2890–2897.
29. Troyanskaya, O.G.; Dolinski, K.; Owen, A.B.; Altman, R.B.; Botstein, D. A Bayesian framework for combining heterogeneous data sources for gene function prediction (in Saccharomyces cerevisiae). *Proc. Natl. Acad. Sci. USA* **2003**, *100*, 8348–8353.
30. Lee, I.; Date, S.V.; Adai, A.T.; Marcotte, E.M. A probabilistic functional network of yeast genes. *Science* **2004**, *306*, 1555–1558.
31. Kourmpetis, Y.A.; Van Dijk, A.D.; Bink, M.C.; van Ham, R.C.; ter Braak, C.J. Bayesian Markov Random Field analysis for protein function prediction based on network data. *PLoS ONE* **2010**, *5*, e9293.
32. Sokolov, A.; Ben-Hur, A. Hierarchical classification of gene ontology terms using the GOstruct method. *J. Bioinform. Comput. Biol.* **2010**, *8*, 357–376.
33. Zhang, C.J.; Tang, H.; Li, W.C.; Lin, H.; Chen, W.; Chou, K.C. iOri-Human: Identify human origin of replication by incorporating dinucleotide physicochemical properties into pseudo nucleotide composition. *Oncotarget* **2016**, *7*, 69783–69793.
34. Pan, Y.; Liu, D.; Deng, L. Accurate prediction of functional effects for variants by combining gradient tree boosting with optimal neighborhood properties. *PLoS ONE* **2017**, *12*, e0179314.
35. Wei, L.; Xing, P.; Tang, J.; Zou, Q. PhosPred-RF: A novel sequence-based predictor for phosphorylation sites using sequential information only. *IEEE Trans. Nano Biosci.* **2017**, *16*, 240–247.
36. Wei, L.; Xing, P.; Su, R.; Shi, G.; Ma, Z.S.; Zou, Q. CPPred-RF: A sequence-based predictor for identifying cell-penetrating peptides and their uptake efficiency. *J. Proteome Res.* **2017**, *16*, 2044–2053.
37. Lan, L.; Djuric, N.; Guo, Y.; Vucetic, S. MS-k NN: Protein function prediction by integrating multiple data sources. *BMC Bioinform.* **2013**, *14*, S8.
38. Cai, C.; Han, L.; Ji, Z.L.; Chen, X.; Chen, Y.Z. SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Res.* **2003**, *31*, 3692–3697.
39. Chen, Y.C.; Lin, Y.S.; Lin, C.J.; Hwang, J.K. Prediction of the bonding states of cysteines using the support vector machines based on multiple feature vectors and cysteine state sequences. *Proteins: Struct. Funct. Bioinform.* **2004**, *55*, 1036–1042.
40. Cai, C.; Wang, W.; Sun, L.; Chen, Y. Protein function classification via support vector machine approach. *Math. Biosci.* **2003**, *185*, 111–122.

41. Halperin, I.; Glazer, D.S.; Wu, S.; Altman, R.B. The FEATURE framework for protein function annotation: Modeling new functions, improving performance, and extending to novel applications. *BMC Genom.* **2008**, *9*, S2.
42. Gustafson, A.M.; Snitkin, E.S.; Parker, S.C.; DeLisi, C.; Kasif, S. Towards the identification of essential genes using targeted genome sequencing and comparative analysis. *BMC Genom.* **2006**, *7*, 265.
43. Manavalan, B.; Lee, J. SVMQA: Support–vector-machine-based protein single-model quality assessment. *Bioinformatics* **2017**, *33*, 2496–2503.
44. Manavalan, B.; Lee, J.; Lee, J. Random Forest-Based Protein Model Quality Assessment (RFMQA) Using Structural Features and Potential Energy. *PLoS ONE* **2014**, *9*, e106542.
45. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444.
46. Cao, R.; Adhikari, B.; Bhattacharya, D.; Sun, M.; Hou, J.; Cheng, J. QAcon: Single model quality assessment using protein structural and contact information with machine learning techniques. *Bioinformatics* **2017**, *33*, 586–588.
47. Sun, M.; Han, T.X.; Liu, M.C.; Khodayari-Rostamabad, A. Multiple Instance Learning Convolutional Neural Networks for Object Recognition. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; pp. 3270–3275.
48. Zou, W.Y.; Wang, X.; Sun, M.; Lin, Y. Generic object detection with dense neural patterns and regionlets. *arXiv* **2014**, arXiv:1404.4316.
49. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv* **2016**, arXiv:1603.04467.
50. Ofer, D.; Linial, M. ProFET: Feature engineering captures high-level protein functions. *Bioinformatics* **2015**, *31*, 3429–3436.
51. Jiang, Y.; Oron, T.R.; Clark, W.T.; Bankapur, A.R.; D'Andrea, D.; Lepore, R.; Funk, C.S.; Kahanda, I.; Verspoor, K.M.; Ben-Hur, A.; et al. An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome Biol.* **2016**, *17*, 184.
52. Cao, R.; Zhong, Z.; Cheng, J. SMISS: A protein function prediction server by integrating multiple sources. *arXiv* **2016**, arXiv:1607.01384.
53. Koskinen, P.; Törönen, P.; Nokso-Koivisto, J.; Holm, L. PANNZER: High-throughput functional annotation of uncharacterized proteins in an error-prone environment. *Bioinformatics* **2015**, *31*, 1544–1552.
54. Clark, W.T.; Radivojac, P. Analysis of protein function and its prediction from amino acid sequence. *Proteins: Struct. Funct. Bioinform.* **2011**, *79*, 2086–2096.
55. Kulmanov, M.; Khan, M.A.; Hoehndorf, R. DeepGO: Predicting protein functions from sequence and interactions using a deep ontology-aware classifier. *arXiv* **2017**, arXiv:1705.05919.
56. Apweiler, R.; Bairoch, A.; Wu, C.H.; Barker, W.C.; Boeckmann, B.; Ferro, S.; Gasteiger, E.; Huang, H.; Lopez, R.; Magrane, M.; et al. UniProt: The universal protein knowledgebase. *Nucleic Acids Res.* **2004**, *32*, D115–D119.
57. Lai, H.Y.; Chen, X.X.; Chen, W.; Tang, H.; Lin, H. Sequence-based predictive modeling to identify cancerlectins. *Oncotarget* **2017**, *8*, 28169.
58. Zhu, P.P.; Li, W.C.; Zhong, Z.J.; Deng, E.Z.; Ding, H.; Chen, W.; Lin, H. Predicting the subcellular localization of mycobacterial proteins by incorporating the optimal tripeptides into the general form of pseudo amino acid composition. *Mol. BioSyst.* **2015**, *11*, 558–563.
59. Chen, X.X.; Tang, H.; Li, W.C.; Wu, H.; Chen, W.; Ding, H.; Lin, H. Identification of bacterial cell wall lyases via pseudo amino acid composition. *BioMed Res. Int.* **2016**, *2016*, 1654623.
60. Yang, H.; Tang, H.; Chen, X.X.; Zhang, C.J.; Zhu, P.P.; Ding, H.; Chen, W.; Lin, H. Identification of secretory proteins in mycobacterium tuberculosis using pseudo amino acid composition. *BioMed Res. Int.* **2016**, *2016*, 5413903.
61. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
62. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; pp. 3104–3112.
63. Tsoumakas, G.; Katakis, I. Multi-label classification: An overview. *Int. J. Data Warehous. Min.* **2006**, *3*, 1–13.

64. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
65. Jean, S.; Cho, K.; Memisevic, R.; Bengio, Y. On using very large target vocabulary for neural machine translation. *arXiv* **2014**, arXiv:1412.2007.