Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: 22 Mar 2019
Applications Note

Genome analysis

Predicting the Subcellular Location of Eukaryotic Proteins

Chang Liu

Computer Science, UCL, London, WC1E 6BT

Abstract

Motivation: Eukaryotic cells have gradually evolved into distinct compartments and each of it with the specific function. Proteins are utilized to guarantee the proper function of a compartment in the eukaryotic cell. The environment in which the proteins work is determined by the subcellular location. Thus, the study of subcellular location plays a crucial role in the determination of protein functions. In addition, we could also have an insight into the organization of the whole cell by knowing the subcellular location.
Results: By using a stacking model of DNN and RNN, we achieved an average predicted accuracy of 77% on non-homologous validation set.
Contact: uczlcl8@ucl.ac.uk

1 Introduction

A variety of structures–organelles which perform various functions are contained in the eukaryotic cells. Every organelle in the eukaryotic cells has a specific function which guarantee the normal operation of cells. These functions are achieved by proteins. Thus, understanding the role of proteins is important for us to have an insight into the operation and organization of a cell as the whole. However, predicting the function for a given protein has proved to be especially difficult where no clear homology to proteins of known function exists. The subcellular locations of proteins in the cell could give us a strong hint of their functions. There are different subcellular locations in the eukaryotic cells which decide the environment where these proteins work. In this paper, we classified the locations to four types: Cytosolic, Secreted, Nuclear and Mitochondrial.

Our work would focus on using feed-forward neural network and recurrent neural network for classification.

2 Literature Review

Considerable efforts have been made into the task of prediction subcellular locations. There are two classes of these efforts (Nakai, 2000). One part of these efforts are based on the recognition of protein N-terminal sorting signals. Another part of these efforts are based on the amino acid sequence information. In 1997, Nielsen et al. published a paper in which their work covers a large area on identifying individual sorting signals, such as signal peptides.

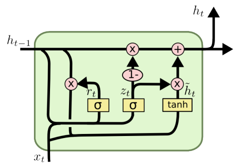
Recent years high amount of projects passing through the field of genome sequencing have produced extremely large amount raw sequence data. Thus, the work on the prediction of subcellular locations has gradually move onto the field of machine learning by using the amino acid sequence information. Furthermore, Horton in 2007 built the extension of PSORT II program and get WoLF PSORT. The K-nearest neighbour classifier for prediction, which belongs to the field of machine learning, is added into the WoLF PSORT.

2.1 Neural Network

Neural network was first put up with by Alan Turing on the paper Intelligent machinery in 1948. An artificial neural network could simply predict the label of an input which is widely used today. As shown in Figure 2, deep neural network is a combination of multiple simple neural network. The output of last layer would be the input of next layer. It could extract more features from the sequence and have a more stable performance. In 2000, Emanuelsson et al. developed a feed-forward neural network tool–TargetP to predict the large-scale subcellular location based on the N-terminal sequence information.

2.2 Recurrent Neural Network

Compared to the feed forward neural network, recurrent neural network is capable of short-memory for dealing with the sequence information because of the recurrent connection. RNN had been introduced firstly by Rumelhart, et al. at 1986. In contrast to the feed-forward neural network model, RNN model would feed its current information to itself at next timestep. So it could be used to analyze the pattern of a sequence, because simple feed-forward model could not remember the old information it had seen before. However, if the sequence is too long, the information at the start of the sequence would be much less than the recent information. To solve this problem, Cho, et al. introduced Gated Recurrent Unit (GRU)



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Fig. 1. Structure of GRU cell

in 2014. GRU is a variant of RNN. By changing the structure of memory cell in RNN, GRU is designed to be capable of long-term memory. In the GRU cell, there is a update gate and a reset gate to control what information should be updated and reset in the next hidden. Figure 1 shows the structure inside a GRU cell.

3 Data

3.1 Training Data

It would be helpful to know the descriptive statistics before we construct our model. The dataset contains 9,222 non-homologous eukaryotic proteins in FASTA format, each of which belongs to one out of four major subcellular locations(Cytosolic, Secreted, Nuclear and Mitochondrial). We only need the amino acids sequence and the location of this dataset to construct our model.

Total	Mean	Std	Max	Min
9,222	547	514	13100	11

As shown in the table above, the mean sequence length of amino acids is 547 with a large standard deviation of 514. The longest sequence has 13100 amino acids and the shortest sequence has only 11 amino acids. In this dataset, these sequences do not contain gaps(-) or translation stops(*), but they might consist of the code X which represent for unknown amino acid. We deal with X by assigning a uniform distribution on 20 amino acids or ignore it sometimes. Similarly, we take code B as mean of code D and N, and code Z as mean of code E and Q. We split the data to 7,377 and 1,845 for training and validation respectively.

3.2 Blind Test Data

Our final model would be tested on the blind test dataset which consists of 20 sequences of amino acids. The descriptive statistics of test data was shown in the table below, and it could be seen that there was no much difference on the structure of training and test dataset.

Total	Mean	Std	Max	Min
20	547	421	1876	141

4 Methods

4.1 Feature engineering

In order to predict the location of a protein, we use Biopython to capture some informative properties of these amino acids. The set of features was chosen included: (1) Sequence length, (2) Global amino acids composition, (3) Local amino acids composition over the first 50 amino acids, (4) Local amino acids composition over the last 50 amino acids, (5) Isoelectric point, (6) Molecular weight, (7) Sequence patterns over the first 100 amino acids, (8) Sequence patterns over the last 100 amino acids, (9) Aromaticity, (10) Hydrophobicity, (11) Hydrophilicity. Notice that Some of these sequences have length less than 50(or 100), so we need to append "*" at the end of these amino acids sequences to the required length.

Fig. 2. Structure of DNN

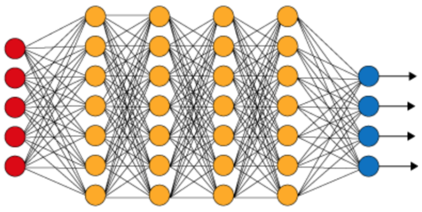
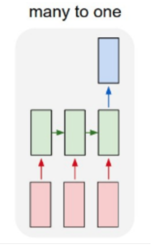


Fig. 3. Structure of RNN



4.2 Model

4.2.1 Deep Neural Network

For features (2), (3) and (4), we used Deep Neural Network in predicting location. The structure of our deep neural network is similar to the DNN in Figure 2, four simple fully connected hidden layers was used, each of which has 512, 256, 128 and 64 units respectively. Between these layers, we add a ReLu layer and a Dropout layer to regularise the results.

The input of this deep neural network contains 24 units, which represent for the proposition of 24 amino acids code. The final output contains four units which is the probability of this protein belongs to each of four possible subcellular location. We combined features (1), (5), (6), (9), (10), (11) and use a deep neural network with the same structure. The input contains 6 units and the output is the probability of this protein belongs to each of four possible subcellular location.

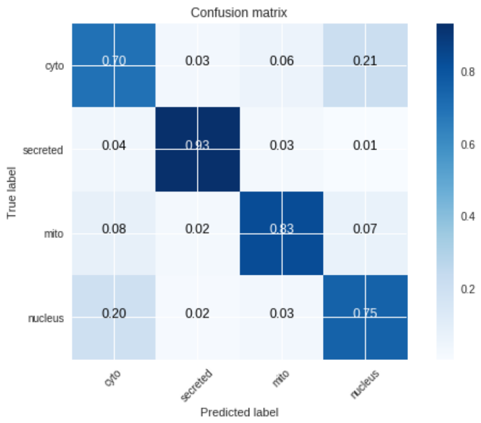
4.2.2 Recurrent Neural Network(GRU)

For features (7) and (8), we used Gated Recurrent Unit(GRU) which is a type of Recurrent Neural Network(RNN) to predict the location. There are several types of RNN: one-to-one, one-to-N, N-to-one, N-to-N. For the purpose of using amino acids sequence to predict location, we chose N-to-one RNN as shown in Figure 3. The reason we choose to use a RNN model instead of a simple NN model is that the order of how the amino acids present is important. Since each of the amino acids in the dataset are represented by a one-letter code and GRU only take vectors as input, we need to transform them to vectors. Usually, people tend to use a one-hot encoder. In our model, we choose the embedding size to be 32, so we could extract more features, e.g. the relation between amino acids, from these sequences. The input should have 25 units, because there are 25 codes shown in the amino acids sequences. Inside of the GRU cells, we decided to use 64 hidden units. The model updated by minimizing categorical cross-entropy loss between the predicted location and the actual location using âŁ˜AdamâŁ™ optimizer.

4.2.3 Stacking of Models

Stacking is an ensemble learning technique that combines multiple classification or regression models through a meta-classifier or a meta-regressor. The base level models are trained based on the training set, and then the meta-model is trained on the outputs of the base level models as features. The base level models often consists of different learning

Fig. 4. Confusion Matrix



algorithms and therefore stacking ensembles are often heterogeneous. Stacking could achieve higher accuracy than individual model and it could also prevent over-fitting. We would discuss this further later. In our model, we choose linear regression as our meta-model which combine the output from our DNN model and RNN model.

5 Experiment

We implemented this model on Google Cobal using Keras. First, we preprocessed the dataset as mentioned before. Then, we draw the features from the preprocessed dataset. Notice that we use Biopython to analyze some features of proteins. In our experiment, we choose the size of hidden units to be 64 and the number of layers to be one for the RNN model. For DNN model, after we get the output from every linear layer, we use a activation function on it. Then we used a dropout layer with rate at 0.3 to prevent over-fitting. After we got the predicted probability from features, we concatenate them together and then apply a regression model on it. In the training process, we updated the model using a butch size of 32 and run for 50 epochs.

6 Discussion and Discussion

6.1 Accuracy

The accuracy is the proportion of correct prediction of location on the validation set. The final accuracy of our model could achieve 0.77. The accuracy of single model is much lower than the meta-model. For our DNN model, the accuracy of feature (2) could achieve around 60, feature (3) could achieve around 54, feature (4) could achieve around 46. The combination of feature (1), (5), (6), (9), (10), (11) could achieve around 45. For our RNN model, the prediction is better. The accuracy of feature (7) could achieve 65 and feature (8) could achieve 49.

6.2 Confusion Matrix

To analyze our model further, we build a confusion matrix. As shown in Figure 4, it is difficult to distinguish proteins in "Cytosol" and "nuclear". 21% of Cyto was predicted to Nuclear and 20% of nuclear was predicted to Cyto. "secreted" has the highest accuracy where 93% of them was predicted correctly.

Table 1. Prediction of 20 blind test set.

Proteins	Location	Confidence
SEQ677	Secreted	0.62
SEQ231	Secreted	1.00
SEQ871	Secreted	1.00
SEQ388	Nuclear	0.89
SEQ122	Nuclear	0.57
SEQ758	Cytosolic	0.55
SEQ333	Nuclear	0.50
SEQ937	Cytosolic	0.81
SEQ351	Nuclear	0.51
SEQ202	Mitochondrial	0.51
SEQ608	Mitochondrial	0.21
SEQ402	Mitochondrial	0.24
SEQ433	Secreted	1.00
SEQ821	Secreted	1.00
SEQ322	Nuclear	0.87
SEQ982	Nuclear	1.00
SEQ951	Cytosolic	0.42
SEQ173	Cytosolic	0.64
SEQ862	Cytosolic	0.21
SEQ224	Cytosolic	0.57

6.3 Prediction of the blind test set

Finally, Table 1 would show the predicted location of the 20 proteins in the blind test set. Besides, it would give a measure of confidence for each protein. We would discuss these confidence further in the next section. We could see form the table that Secreted was given as a prediction, it usually comes with a high confidence. But for Mitochondrial and Cytosolic, the conference would be much lower.

6.4 Measure of confidence

In this paper, we define the confidence of a prediction to be

$$\frac{1}{2} \times [\text{largest probability} + (\text{largest probability} - \text{second largest probability})]$$

So if the model predict a location with probability 1, the confidence of the prediction would be 1. But when there are two relative high values in the predicted probability, the confidence of the prediction would decrease.

7 Conclusion and Future Work

In this paper, we have shown that the combination of DNN model and RNN model could be used to predict the location of a given protein sequence. Our model could achieve great generalization on unseen protein sequences with accuracy around 77% which is better than previous work. This still need validation on large scale dataset. our model would give high confidence on predicting Secreted proteins, but would give a relatively low confidence when predicting Cytosolic and Mitochondrial proteins. We could work on improving the accuracy on prediction of Cytosolic and Mitochondrial proteins further.

Reference

Nakai K. Protein sorting signals and prediction of subcellular localization[J]. Advances in protein chemistry, 2000, 54: 277-344.

Nielsen H, Engelbrecht J, Brunak S, et al. Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites[J]. Protein engineering, 1997, 10(1): 1-6.

Emanuelsson O, Nielsen H, Brunak S, et al. Predicting subcellular localization of proteins based on their N-terminal amino acid sequence[J]. Journal of molecular biology, 2000, 300(4): 1005-1016.

Horton P, Park K J, Obayashi T, et al. WoLF PSORT: protein localization predictor[J]. Nucleic acids research, 2007, 35(suppl2): W585-W587.

Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. Cognitive modeling, 1988, 5(3): 1.

Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." arXiv preprint arXiv:1406.1078 (2014).

Smolyakov V, Ensemble Learning to Improve Machine Learning Results. https://blog.statsbot.co/ensemble-learning-d1dcd548e936