Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

Mathematical Biosciences and Engineering, 16(3): 1709–1717
DOI: 10.3934/mbe.2019081
Received: 17 December 2018
Accepted: 31 January 2019
Published: 27 February 2019

Research article

A viral protein identifying framework based on temporal convolutional network

Hanyu Zhao[1], Chao Che[1,*], Bo Jin[2] and Xiaopeng Wei[3]

[1] Key Laboratory of Advanced Design and Intelligent Computing, Ministry of Education, Dalian University, Dalian 116622, China

[2] School of Innovation and Entrepreneurship, Dalian University of Technology, Dalian, 116024, China

[3] School of Computer Science, Dalian University of Technology, Dalian 116024, China

* **Correspondence:** Email: chechao101@163.com.

Abstract: The interaction between viral proteins and small molecule compounds is the basis of drug design. Therefore, it is a fundamental challenge to identify viral proteins according to their amino acid sequences in the field of biopharmaceuticals. The traditional prediction methods suffer from the data imbalance problem and take too long computation time. To this end, this paper proposes a deep learning framework for virus protein identifying. In the framework, we employ Temporal Convolutional Network(TCN) instead of Recurrent Neural Network(RNN) for feature extraction to improve computation efficiency. We also customize the cost-sensitive loss function of TCN and introduce the misclassification cost of training samples into the weight update of Gradient Boosting Decision Tree(GBDT) to address data imbalance problem. Experiment results show that our framework not only outperforms traditional data imbalance methods but also greatly reduces the computation time with slight performance enhancement.

Keywords: viral protein identifying; data imbalance; deep learning; TCN; GBDT

1. Introduction

Protein is one of the main components of cells and performs a vast array of function in the organisms. In the biopharmaceutical field, the drug is designed according to the interaction between viral proteins and small molecule compounds. With the development of bioinformatics, extensive information of protein sequence continuously come forth. It is a time-consuming and labor-intensive work to identify viral proteins by molecular biology experiments. Therefore, exploring computer technology to identify viral proteins efficiently is an essential way in the biopharmaceutical field. At present,

the primary protein sequence recognition methods include prediction methods based on sequence homology comparison and machine learning methods based on feature extraction. Methods based on sequence homology comparison mainly include Basic Local Alignment Search Tool (BLAST) [1] and FAST-Aye (FASTA) [2]. The main problem faced by such methods is that there are some proteins with high homology that exhibit different functions. The machine learning methods mainly construct the feature set by mining the features of the protein sequence and then use the machine learning algorithm to perform prediction. The performance of machine learning methods is superior to the prediction method based on sequence homology comparison in identifying proteins. However, since viral protein sequences account for only a small fraction of all protein sequences, traditional feature extraction methods cannot efficiently extract features of the rare class, namely, viral protein. TCN [3] can extract sequence information more efficiently by stacking more convolutional layers, using a larger dilation factor, increasing the size of the filter and expanding the receptive field. Compared with the traditional sequence modeling, RNN [4], which must wait for the former element to be processed before the next element can be processed and is extremely computationally intensive, TCN require less memory and time for training especially for long input sequence like protein sequence. Thus, this paper employs an improved deep learning framework based on TCN to identify the viral protein. In the framework, we propose Cost-TCN algorithm to extract features of the protein sequence and Cost-GBDT algorithm to identify the viral proteins, respectively. To address the data imbalance problem, Cost-TCN customizes the loss function of TCN, and Cost-GBDT changes the update rules of GBDT[5].

2. Related work

The first step in the machine learning method for identifying viral protein is to construct a sequence feature set by mining the protein sequence. For example, Wang et al. [6] proposed a feature construction method of 3-gram. However, the dimensions of the extracted feature by this approach will exponentially increase when n takes a big value. Lin et al. [7] put forward a 188-dimensional method which combined n-gram and physicochemical properties to reduce dimensional catastrophe.

Nevertheless, the above methods still cannot effectively extract the dependence between functional regions in the sequence. In recent years, the neural network has made significant achievements in language modeling. With the continuous research on protein sequences, more and more scholars have focused their attention on deep learning models. Timothy et al. [8] attempted to classify proteins using deep learning networks such as Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). Moreover, the classification accuracy based on GRU network had reached 0.9484. Li Hongshun et al. [9] first used CNN to detect the functional domain of proteins and then used LSTM to study the long-term dependence between functional domains, and finally achieved an accuracy of 0.9592 in the prediction of RNA-binding proteins. Although LSTM reduces the gradient explosion or gradient disappearance that often occurs in RNN to a certain extent, the problem cannot be completely avoided. However, the path of the TCN in the backpropagation is different from the time direction of the sequence, which can effectively avoid such problems.

In protein sequence data, viral protein sequences account for only a small fraction of all sequences and the data imbalance severely restricts the performance of protein identification. At present, there are two strategies to solve the data imbalance problem. One is to reduce the imbalance by changing the sample distribution of the training set. For example, Chawla et al.[10] proposed Synthetic Mi-

nority Over-sampling Technique(SMOTE) algorithm to balance the training data by generating small class without repetition. Another method is to use the learning algorithm, which mainly refers to cost-sensitive learning methods. Fan et al. [11] proposed Cost-sensitive Adaptive Boosting(Cost-Adaboost) algorithm to improve the classification accuracy of boosting algorithm on imbalance samples by introducing the error cost of each training sample into the weight updating rule of boosting algorithm [12]. However, the above methods cannot effectively extract the features of the rare class in protein sequence. This paper proposes a viral protein identifying model based on Cost-TCN and Cost-GBDT, which introduces the error cost of the sample into the both feature extraction and classification iterative processes so that the model can pay more attention to rare class and solves the imbalance problem.

3. Methods

In order to predict the viral protein in the protein sequence, we designed a three-stage deep learning framework consisting of protein embedding, feature extraction and protein classification. We first encode amino acids in the protein sequence so that the computer can recognize the protein sequence. Then we propose a deep learning model called Cost-TCN to extract features from the encoded protein sequence automatically. Finally, Cost-GBDT algorithm which combines GBDT with cost-sensitive learning is employed to classify proteins and the extracted feature is used as the input.

3.1. Protein embeddings

A protein sequence can be described as $s = a_1a_2...a_n$, where n is the length of the protein sequence, a_i is an amino acid which comes from the collection $\sum = A, B, ..., Z$. In order to represent the protein sequences, the 20 amino acids in Table 1 are encoded, and each amino acid is mapped to a specific real number. Suppose s is a 10-length protein sequence $s = MDKFRVOGPT$, the amino acids in the sequence are encoded according to Table 1, and each sequence is converted into a 1-dimensional vector.

$$S_i = encoding(s) = (18, 7, 10, 14, 11, 4, 15, 3, 12, 8) \tag{3.1}$$

Table 1. Amino Acid Embeddings Cross Reference Table.

Amino Acids	Letters	code	Amino Acids	Letters	code
Leucine	L	1	Alanine	A	2
Glycine	G	3	Valine	V	4
Glutamic	E	5	Serine	S	6
Aspartic	D	7	Threonine	T	8
Isoleucine	I	9	Lysine	K	10
Arginine	R	11	Proline	P	12
Asparagine	N	13	Phenylalanine	F	14
Glutamine	Q	15	Tyrosine	Y	16
Histidine	H	17	Methionine	M	18
Tryptophan	W	19	Cysteine	C	20
Illegal Amino acids	B,j,O,U,X,Z	0			

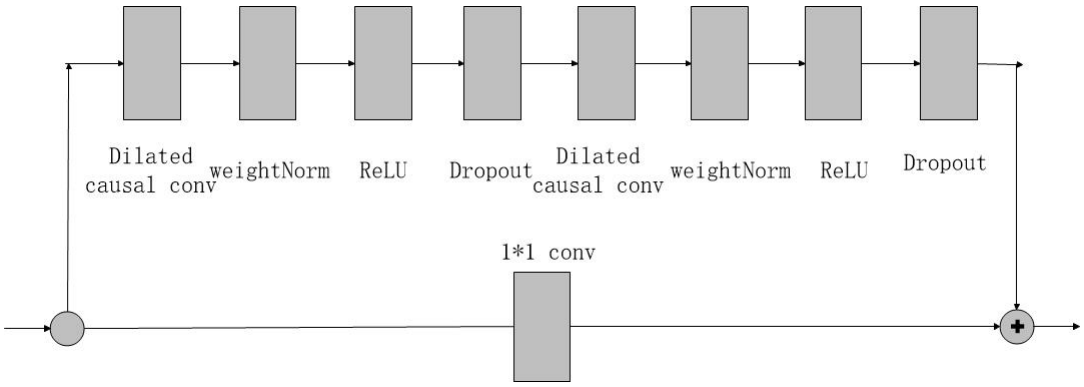

Figure 1. Residual Block.

3.2. Cost-TCN feature extraction

TCN combines the two structures of 1D fully-convolutional [13] and causal convolutions [14]. The use of a 1D fully-convolutional network allows TCN to produce output sequences of the same length as the input sequence, and each hidden layer can keep the same length with the output layer after padding. By using causal convolutions, TCN can guarantee that the prediction of the previous time step does not need future information because the output of time step t only relies on the convolutions operation on $t-1$ and the previous time step. The formula is as follows:

$$y_t = \prod_{t=1}^{T} P(x_t|x_1x_2...x_t - 1) \tag{3.2}$$

However, in order to construct a long-term memory of sequences, causal convolutions require a huge number of layers or a large convolutions kernel to broaden the receptive field, which costs a significant number of computational resources. To solve this problem, TCN uses dilated convolutions [15] to enable an exponentially large receptive field. The dilated convolutions add some weights to the convolutions kernel with zero value while keeping the input unchanged, which can increase the length of the observed sequence while not increasing the amount of calculation. The size of the 1D fully-convolutions kernel is 2, the dilation factor of the first layer is 1, dilated convolutions reduce to a regular convolution. Moreover, the following dilation factor is sequentially increased. A regular convolutions operation can only observe five inputs from left to right, while a hole convolutions can observe 16 input data. Formally, for a one-dimensional input sequence $x \in R^n$ and convolutions kernel $f : 0, 1, ..., k-1 \rightarrow R$, the formula for dilated convolutions is:

$$F(s) = (x * df)(s) = \sum_{i=0}^{k-1} f(i)s_{x-di} \tag{3.3}$$

Where d is the dilation factor, k is the filter size, and $s - d * i$ accounts for the direction of the past. The dilation factor controls the number of zeros inserted between each two convolutions kernels. The larger dilation factor allows the neurons at the output to characterize a broader range of input sequences, which can more effectively expand the receptive field. Therefore, when we use dilated convolutions, we generally increase the expansion factor exponentially with the increase of network depth. In addition,

TCN adds a jump connection block within residual convolutions [16] and a $1*1$ convolutions operation to improve accuracy. Residual block is shown in Figure1.

Within the residual block of the TCN, there are two layers of dilated convolutions and loss functions ReLU, and the weights of each convolutions kernel are normalized, and dropout is added after each dilated convolutions to perform regularization. Since the input and output dimensions of the residual convolutions are different in the TCN, we additionally use a $1*1$ convolutions to ensure that the features added from the lower layer to the upper layer receive the same shape tensor.

Traditional TCN classification model uses cross-entropy loss function. Considering the data imbalance problem, we introduce cost-sensitive learning into TCN network to better extract the feature of the rare class. The loss function formula of Cost-TCN is defined as below:

$$\begin{cases} \beta = \frac{cpos}{cneg+cpos} \\ L(x) = -(\beta P(x) \sum_{j \ni Y_+} \log q(y_i = 0|x) + (1-\beta) P(x) \sum_{j \ni Y_-} \log q(y_i = 1|x)) \end{cases} \tag{3.4}$$

Where $cpos$ denotes the positive number, $cneg$ represents the negative number, $p(x)$ is the true value, and $q(x)$ is the predicted value. By weighting the loss values of positive and negative samples, the network pays more attention to the classification results of the rare class, so as to better identify the viral proteins.

3.3. Cost-GBDT protein classification

In the imbalanced dataset, rare classes are the focus of classification, and it is more valuable to identify the samples of rare class correctly. Cost-sensitive learning solves the data imbalance problem by assigning different misclassification costs to different classes to minimize the total cost. Borrowing from the idea of cost-sensitive learning, we proposed Cost-GBDT by changing the update rules of GBDT weights and effectively improve the classification performance of GBDT for the rare class.

Algorithm 1 LK TreeBoost
Input: $X = x_1, x_2, .., X_n$
Output: $F_{km}(x)$
1. $F_{k0} = 0, \quad k = 1, K$
2. $For \quad m = 1 \quad to \quad M \quad do:$
3. $\quad P_k(x) = \frac{e^{F_K(x)}}{\sum_{l=1}^{k} e^{F_l(x)}}, \quad k = 1, K$
4. $\quad For \quad k = 1 \quad to \quad K \quad do:$
5. $\quad\quad L(y_k, F_k(x)_1^2) = -\sum_{k=1}^{2} \frac{y_k}{y} y_k \log p_k(x)$
6. $\quad\quad \tilde{y} = y_{ik} - P_{k,m-1}(x_i)$
7. $\quad\quad \{R_{jkm}\}_{j=1}^{J} = J \quad terminal \quad node \quad tree(\{\tilde{y}_{ik}, x_i\}_1^N)$
8. $\quad\quad \gamma_{jkm} = \frac{K-1}{K} \frac{\sum_{x_{j \in R_{jkm}}} \tilde{y}_{ik}}{\sum_{x_{j \in R_{jkm}}} \|\tilde{y}_{ik}\|(1-\|\tilde{y}_{ik}\|)}$
9. $\quad\quad F_{km}(x) = F_{k,m-1}(x) + \sum_{j=1}^{J} \gamma_{jkm} 1(x \in R_{klm})$
10. $\quad$ end For
11. end For
12. end Algorithm

GBDT is a boosting ensemble learning algorithms, which promotes iterative learning from weak learner to strong learner. Because of its high efficiency, accuracy, and interpretability, GBDT has been widely used in various machine learning tasks. The traditional GBDT algorithm integrates the decision tree and trains the decision tree successively by fitting the steepest descent in each iteration. In order to make GBDT better solve the data imbalance problem, we update the weights of the samples according to the size of the class of the sample. We increase the weights of error samples of the rare classed in the training process so that the weak classifier will pay more attention to the rare classes. The Cost-GBDT algorithm flow is shown as Algorithm 1.

4. Results and discussion

4.1. Dataset

In the experiment, we selected 278866 protein sequences obtained from the RCSB PDB protein database as the raw data, and only use the protein types with more than 2000 samples. In the dataset, 8495 viral protein sequences are marked as positive examples and 270371 protein sequences of other types are marked as negative examples. We selected 80% of the data as the training set, and 20% of the data as the test set. The dataset we used in the experiment is shown in Table 2.

Table 2. Division and Construction of Data Sets.

Data	Non VIRAL	VIRAL	Total
Original Set	270371	8495	278866
Train Set	216296	6796	223092
Test Set	54075	1699	55774

To verify the effectiveness of our framework in solving data imbalance problem, we employed AUC, Sensitive and Specificity [17] to evaluate the model performance.

4.2. Feature extraction methods comparison

To verify the advantages of TCN network in extracting protein features from protein datasets, we compared the traditional feature extraction methods with the deep learning models. We choose 3-gram method as traditional feature extraction method and employ four RNN models including LSTM, GRU, Bi-Directional LSTM(BiLSTM) and Bi-Directional GRU(BiGRU) as deep learning models. All the methods used softmax to do classification. The experimental results are shown in Table 3.

Table 3. The Comparison Results of Different Feature Extraction Methods.

Feature Extraction methods	Sensitivity	Specificity	AUC	Time
3_gram	0.7836	0.9732	0.8783	-
LSTM	0.8423	0.9980	0.9201	8328s
GRU	0.8529	0.9982	0.9255	6521s
BiLSTM	0.8634	0.9972	0.9303	15501s
BiGRU	0.8623	0.9981	0.9302	12812s
TCN	0.8735	0.9973	0.9353	2085s

As we can see from Table 3, the deep learning models have significant advantages over the traditional method in extracting features from large-scale protein datasets. The traditional method cannot get the location information between the motifs, while the deep learning models can automatically learn the dependence between motif and motif. TCN only performs slightly better than RNN models in sensitivity and AUC, and even a little worse in specificity. However, it shows tremendous advantages in computational efficiency and only takes about 1/3-1/6 of the time of RNN models. TCN can further reduce the computation time by parallel computation, while RNN model can not be implemented in parallel. When dealing with the protein sequence problems, TCN can better control the dependence of protein sequences by using causal convolutions and dilated convolutions, which consider the sequence of amino acids, expand the receptive field and can better control the length of memory.

4.3. Cost-TCN vs. TCN

For the validity of feature extraction on imbalanced samples, we compared Cost-TCN method with the traditional TCN network. The comparison results are shown in Table 4.

Table 4. The Performances of COST-TCN and TCN.

Feature Extraction methods	Sensitivity	Specificity	AUC
TCN+softmax	0.8735	0.9973	0.9353
Cost-TCN+softmax	0.8917	0.9969	0.9443

Table 4 shows that Cost-TCN using the cost-sensitive cross-entropy loss function can classify the imbalanced data more effectively, especially for the rare class. Since the number of non-viral proteins is much larger than the virus protein in the training set, TCN is more inclined to classify proteins into non-viral proteins in the iterative process. Meanwhile, Cost-TCN algorithm pays more attention to the classification results of the rare class in the iterative process by assigning bigger weight to the samples virus protein in training. This can effectively improve the classification performance of the rare class, namely, the viral proteins.

4.4. Comparison of methods for data imbalance

As can be seen from Table 4, the classifier tends to classify the sample in an unbalanced data set into the categories with a large sample size, which results in a low recall for the categories with small sample size. To identify the viral protein with small sample size, we extract protein feature using Cost-TCN and employ Cost-GBDT to predict the viral protein. To verify the effectiveness of the framework combining Cost-TCN and Cost-GBDT for solving the data imbalance problem, we compared our framework with two traditional methods for data imbalance problem, namely, SMOTE and the downsampling method, Edited Nearest Neighbors (ENN). The feature extracted by three methods are all fed into Cost-GBDT to predict the viral protein. The results are shown in Table 5.

Table 5 shows that Cost-TCN+Cost-GBDT can improve the identifying performance of viral protein significantly. Although the traditional method based on data sampling can solve the sample imbalance problem to a certain extent, it still has major defects for example, the SMOTE algorithm based on the oversampling balances the training samples by generating new samples around small sample sequences. However, it sometimes pays too much attention to these samples, which will result in over-

fitting. The ENN algorithm based on down-sampling lose too much detailed information of the data, which will decrease identifying performance. Compared with the traditional method, Cost-TCN+Cost-GBDT will make the rare class samples get more attention in the process of classification by giving them a larger weight in the iterative process.

Table 5. The Performances of COST-TCN and TCN.

Methods	Sensitivity	Specificity	AUC
SMOTE +GBDT	0.8377	0.9979	0.9157
ENN +GBDT	0.8710	0.9974	0.9342
Cost-TCN+softmax	0.8917	0.9969	0.9443
Cost-TCN+GBDT	0.9417	0.9949	0.9683
Cost-TCN+Cost-GBDT	0.9508	0.9950	0.9728

5. Conclusions

This paper proposed a deep learning framework based on TCN to identify the viral protein in the protein sequence. Our framework greatly improved the computational efficiency without loss of performance. In the future, we will test our framework on different protein databases and continue to optimize the Cost-TCN model and parameters. We are going to build a deeper Cost-TCN model, which can better extract protein sequence features.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 61402068, No. 91546123) and Support Program of Outstanding Young Scholar in Liaoning Universities. (No. LJQ2015004).

Conflict of interest

All authors declare no additional conicts of interest in this paper.

References

1. O. P. Zhirnov, A. L. Ksenofontov and N. D. Klenk, Influenza A virus M1 matrix protein is similar to protease inhibitors, *Dokl. Akad. Nauk*, **367**(1999), 690–693.
2. S. Niu, T. Huang and K. Feng, et al., Prediction of tyrosine sulfation with mRMR feature selection and analysis, *J. Proteome Res.*, **9**(2010), 6490–6497.
3. S. Bai, J. Z. Kolter, and V. Koltun, An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, preprint, `arXiv:0707.0078`.
4. Z. C. Lipton, J. Berkowitz, and C. Elkan, A critical review of recurrent neural networks for sequence learning, preprint, `arXiv:1506.00019`.

5. J. H. Friedman, Greedy function approximation: A gradient boosting machine, *Ann. Stat.*, **29**(2001), 1189–1232.
6. D. Wang, N. K. Lee and T. S. Dillon, et al., Protein sequences classification using radial basis function (RBF) neural networks, *In: International Conference on Neural Information Processing*, **2**(2002), 764–768.
7. C. Lin, Y. Zou and J. Qin, et al., Hierarchical Classification of Protein Folds Using a Novel Ensemble Classifier, *PLoS One*, **8**(2013), e56499.
8. T. K. Lee and T. Nguyen, Protein Family Classication with Neural Networks, Stanford University, 2016, Available online: https://cs224d.stanford.edu/reports/LeeNguyen.pdf.
9. H. Li, H. Yu and X. Gong, A Deep Learning Model for Predicting RNA-Binding Proteins Only from Primary Sequences, *J. Comput. Res. Dev.*, **55**(2018), 93–101.
10. N. V. Chawla, K. W. Bowyer and L. O. Hall, et al., SMOTE: synthetic minority over-sampling technique, *J. Artif. Intell. Res.*, **16**(2002), 321-357.
11. W. Fan, S. J. Stolfo and J. Zhang, et al., AdaCost: Misclassification Cost-Sensitive Boosting, *In: Sixteenth International Conference on Machine Learning*, (1999), 97–105.
12. Y. Freund and R.E. Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, *J. Comput. Syst. Sci.*, **55**(1997), 119–139.
13. J. Long, E. Shelhamer and T. Darrell, Fully convolutional networks for semantic segmentation, *In: IEEE Conference on Computer Vision and Pattern Recognition*, **39**(2015), 640–651.
14. T. J. Brazil, Causal-Convolution-A New Method for the Transient Analysis of Linear Systems at Microwave Frequencies, *IEEE T. Microw. Theory*, **43**(1995), 315–323.
15. A. V. D. Oord, S. Dieleman and H. Zen, et al., WaveNet: A Generative Model for Raw Audio, preprint, `arXiv:1609.03499`.
16. K. He, X. Zhang and S. Ren, et al., Deep Residual Learning for Image Recognition, *In: IEEE Conference on Computer Vision and Pattern Recognition*, (2016), 770–778.
17. P. Branco, L. Torgo and R. Ribeiro, A survey of predictive modelling under imbalanced distributions. preprint, `arXiv:1505.01658`.