Classification of Macromolecule Type Based on Sequences of Amino Acids Using Deep Learning

Sarwar Khan
Academia Sinica, Taiwan

Faisal Ghafr Khan
Academia Sinica, Taiwan

Imad Ali
University of Swat, Pakistan

Qazi Mazhar
NTUT, Taiwan

ABSTRACT

The classification of amino acids and their sequence analysis plays a vital role in life sciences and is a challenging task. This article uses and compares state-of-the-art deep learning models like convolution neural networks (CNN), long short-term memory (LSTM), and gated recurrent units (GRU) to solve macromolecule classification problems using amino acids. These models have efficient frameworks for solving a broad spectrum of complex learning problems compared to traditional machine learning techniques. We use word embedding to represent the amino acid sequences as vectors. The CNN extracts features from amino acid sequences, which are treated as vectors, then fed to the models mentioned above to train a robust classifier. Our results show that word2vec as embedding combined with VGG-16 performs better than LSTM and GRU. The proposed approach gets an error rate of 1.5%.

Keywords: CNN, LSTM, Amino Acid, Macromolecules

INTRODUCTION

The last decade has witnessed the great success of deep learning as it has brought revolutionary advances in many application domains, including computer vision, natural language processing, and signal processing. The key idea behind deep learning is to consider feature learning and classification in the same network architecture, using back-propagation to update model parameters and learn discriminative feature representations. More importantly, many novel deep learning methods have been devised and improved classification performance significantly [4], [11], [16].

Lee et al. [10] targeted learning as an informative feature representation of protein sequence as the input of neural network models to obtain the final predicting output of the belonging protein family. Hou et al. [5] proposed a framework with a deep 1D convolution neural network (CNN), which is robust in both fold recognition and the study of sequence-structure relationships to classify protein sequences. Nguyen et al. [13] developed a framework with a convolution neural network that used the idea of translation to convert DNA sequences to word sequences for final classification.

The revolution in machine learning, mainly deep learning [7]– [9], made it possible to study and extract a complex pattern from data to make the machine model more robust. Studying deoxyribonucleic acid (DNA) in life sciences is essential for understanding organisms. Current sequencing technologies make it possible to read DNA sequences at a lower cost. DNA databases are increasing daily, and we need to use the power of modern computing to help understand DNA. One of the most critical and essential tasks is to classify DNA sequences.
Four major classes of organic macromolecules are always found in all lifeforms on Earth, from the tiniest bacterium to the giant sperm whale. They are essential to life, e.g., carbohydrates, lipids (or fats), proteins, and nucleic acids. The significant macromolecule classes are similar in that they are large polymers assembled from small repeating monomer subunits. Proteins are large, complex molecules that play many critical roles in the body. They are made up of hundreds or thousands of smaller units called amino acids, which are attached in long chains. Twenty different types of amino acids can be combined to make a protein. These 20 standard amino acids are as follows: alanine, arginine, asparagine, aspartic acid, cysteine, glutamic acid, glutamine, glycine, histidine, isoleucine, leucine, lysine, methionine, phenylalanine, proline, serine, threonine, tryptophan, tyrosine, and valine. The sequence of amino acids determines each protein's unique 3-dimensional structure and specific function.

Carbohydrates are polymers that include both sugars and polymers of sugars, and they serve as fuel and building materials both within and outside the cells. For instance, fructose and glucose are examples of carbohydrates that are essential to life. Nucleic acids are polymeric macromolecules critical to all known life forms. The two types of nucleic acids are DNA and RNA, both found in nuclei of cells. They allow organisms to reproduce their complex components.

The interaction of the protein with protein and protein with DNA/RNA play a pivotal role in protein function. Experimental detection of residues in protein-protein interaction surfaces must come from determining the structure of protein-protein, protein-DNA, and protein-RNA complexes. However, experimental determination of such complexes lags far behind the number of known protein sequences. Hence, there is a need to develop reliable computational methods for identifying protein-protein, protein-RNA, and protein-DNA interface residues. Identifying macromolecules and detecting specific amino acid residues that contribute to the strength of interactions is a fundamental problem with broad applications ranging from rational drug design to the analysis of metabolic and signal transduction networks.

To cope with this challenge, this article aims to utilize deep learning models to classify the macromolecule types given the amino acid sequence and residue count. We use and compare state-of-the-art deep learning models like CNN, long short-term memory (LSTM), and gated recurrent units (GRU). We use word embedding to represent the amino acid sequences as vectors. The CNN extracts features from amino acid sequences, which are treated as vectors, then fed to the models mentioned above to train a robust classifier. Our results show that word2vec as embedding combined with VGG-16 performs better than LSTM and GRU.

The rest of the article is organized as follows: Sec. II describes the materials and methods, while Sec. III presents the deep learning models with the proposed framework. Sec. IV describes the experiments and results, while Sec. V concludes this article.
**Table 1: Types of macromolecule proteins**

| Type                  | Label                      | Data structure | Entries |
|-----------------------|----------------------------|----------------|---------|
| Structure ID          | structureid                | object         | 140250  |
| Chain ID              | chainid                    | object         | 2837    |
| Sequence              | protein sequence           | object         | 104813  |
| Residue Count         | No. of residues ATCG’s     | integer        | 4737    |
| Macromolecule Type    | type of Macro-molecule     | object         | 14      |

**MATERIALS AND METHODS**

In this section, we describe the dataset and its data mining steps.

**A. Dataset**

We used the protein data bank dataset (https://www.rcsb.org/) that contains two files with a different number of entries. The dataset has 467304 entries with five columns. **Table 1** shows four macromolecule proteins: DNA, RNA, and protein/DNA/RNA Hybrid. We dropped the other types during the pre-processing step. The second file is also arranged based on structure ID. This file contains protein meta-data, i.e., resolution, extraction method, experimental technique, etc., with 141401 entries with 14 columns. We can merge both files based on structure ID. The pre-processing step is to drop all the entries with NaN value or if a label or sequence is missing. After removing the missing values, the sequence is checked for the removal of tags and numbers.

Once the dataset is cleaned, the sequence is divided into tri-gram, each sequence now combining three characters strings. (e.g., CGC GAA TTC GCG). The final block may not have all three, and we added 0 to make it an equal slice (padding). The final output contains two columns, one for sequence and the other for the label. As discussed earlier, there are 432474 rows in the processed data with four labels. This is unbalanced data, and we will discuss the incoming augmentation section. We also create a particular case of the dataset to balance and normalize all the classes. We take 424 sequences of each class to create a mini dataset and test the models’ performance. This mini dataset has almost the same results and the whole set with some down/upsampling.

**B. Biological Structures in the Dataset**

DNA makes RNA, RNA makes amino acids, and amino-acid makes a protein which is the central dogma of life. A DNA or RNA is made of Nucleotides, four types (A, T/U, C, G). The nucleotide sequence is the combination of these nucleotides in a row. Three nucleotides combine to form a codon, building a block of Amino Acids. The amino acid then combines to form proteins. To make a protein, at least 20 amino acids are necessary.
To explain, consider this real example. ATT is a codon, which is three nucleotides. This codon represents amino acid (isoleucine) represented by the letter "I." TTT is another codon that means another amino acid (phenylalanine) and is characterized by the letter "F." These "IF" combine along with others to make protein. The letters in the codon represent nucleotides, while the letters in the protein sequence represent amino acids. At least 20 amino-acid must combine to make one functional protein. The maximum number depends on when machinery overcomes a stop codon to stop making one protein. The machinery may crush a finish codon after 20 or overcome after 500. The amino acid sequence determines the type of proteins.

**PROPOSED METHODS**

*Figure 1* shows the block diagram of the proposed system. This system can be broadly divided into three categories. The first is dataset processing, and the second part of this model is embedding. We have different choices in embedding, but word2vec [12], Fast-Text [6], and GloVe [14] are well-known embedding techniques in natural language processing (NLP). These embedding has been tested in different bioinformatics tasks, and the results are promising. We used word2vec in this task. One-hot vector is another famous embedding method for amino acid representation, and we used it compared to other embedding techniques. The final category is CNN, and we will have to determine the number of layers for this task. We also need to find hyper-parameters along with the model's size, height, and width. The output layer is a SoftMax layer used to classify the sequence. This is explained in detail in the next section.

Classifying macromolecule types using sequence can be seen as a sequence classification problem. This is analogous to the sentence classification task in NLP. Thus, we apply a skip-gram analysis from NLP research to model our problem. There are various sequence models in the deep learning domain. We used LSTM, GRU, and 1D Convolution for our problem. Recurrent neural networks, such as LSTM are specifically designed to support input data sequences. *Figure 2* shows the layout of the model. They can learn the complex dynamics within the temporal ordering of input sequences and use internal memory to remember or use information across long input sequences. As it is crucial to any deep learning
task, we applied a preprocessing data task before feeding it to our model. This task includes handling missing values and downsampling the dominant class to balance data distribution.

A. Word Embedding

We used two types of embedding techniques for this work. The first one is word2vec [12]. We need these embedding techniques because deep learning or machine learning model only deals with real numbers. Embedding not only converts these texts or sequences into numbers but also produces relationships between them. We have two algorithms for word2vec, skip-gram and Common Bag of Words (CBOW). After using both of these algorithms in all three models, we find that skip-gram work better in our case. We used the tri-gram data to generate our word2vec model. Figure 3 shows the relationship between sequences. We used different output dimensions, i.e., 100, 150, and 300. We find that the 300-dimension output has better performance.

B. Convolution Neural Network

The convolution neural network (CNN) is the most famous in deep learning. We used the network architecture of VGG [15]-[2]. We are using Convolution1D as the

---

**Figure 2:** LSTM model using embedding at the input.

**Figure 3:** word2vec model relation between sequences
This network has four convolution layers, each layer followed by a max-pooling layer. The network also includes batch normalization and dropout to prevent the model from over-fitting. Two dense layers follow the final max-pooling layer. We set the following hyper-parameters, which give us the best results. Learning rate 0.001, batch size 512, loss function cross-entropy, optimizer Adam, number of epochs 20, dropout rate 0.5, activation ReLU and final layer is SoftMax. Figure 4 shows the model architecture along with some model parameters.

EXPERIMENTS AND RESULTS

We evaluate our models on a protein data bank, a database for the three-dimensional structural data of large biological molecules, such as proteins and nucleic acids. Performance comparison between different models will only make sense if we keep pre-processing and embedding the same. Our different experiments show that word2vec with 300 dimensions performs better, and we will keep this setting unless mentioned.

A. CNN Model Results

Figure 5 shows the validation and training loss over the 50 epochs with an embedding dimension of is 50. We used an early stopping algorithm to save and use our best model. Training loss is 0.028, while validation loss is 0.034. Figure 6 shows the accuracy curve for the same setting. The final training
accuracy is 99.2%, and the test accuracy is 98.8%. The micro-average and macro-average are almost the same. Figure 7 shows the confusion matrix of all four classes. The accuracy of each class is nearly the same, and we can see that from the diagonal color. Figure 8 shows the CNN model's Precision, Recall, and F1-Score.

B. Additional Simulations

One hundred estimators initialized random forest, showing 96.83% test accuracy for more than 90 thousand test sequences. We tested the random forest for balance and unbalanced data, and the results were almost identical. Table 2 shows each class's Precision, Recall, and F1-Score. The results are not good enough compared to CNN, but Random forest is much faster and less expensive than CNN. This comparison may be fair, but we are reporting these results to show that traditional machine learning algorithms also work better for some visual question answering times. The next step is to compare the different state-of-the-art algorithms like LSTM, Random Forest, and GRU with CNN [2].

B. Additional Simulations

One hundred estimators initialized random forest, showing 96.83% test accuracy for more than 90 thousand test sequences. We tested the random forest for balance and unbalanced data, and the results were almost identical. Table 2 shows each class's Precision, Recall, and F1-Score. The results are not good enough compared to CNN, but Random forest is much faster and less expensive than CNN. This comparison may be fair, but we are reporting these results to show
that traditional machine learning algorithms also work better for some visual question answering times.

The next step is to compare the different state-of-the-art algorithms like LSTM, Random Forest, and GRU with CNN [2]. We are using TensorFlow embedding in this case with 50-dimension vectors. In this case, the LSTM and GRU have the same network structure and use the same number of cells, 512. Table 3 shows the comparison of accuracy and loss across the different networks. The table shows that VGG-16 [15] performs better with word2vec [12] as embedding. Seven-layer CNN also performs better compared to LSTM and GRU models. All the convolution neural networks are one dimension. Figure 9 shows the complete dataset's precision, Recall, F1-Score, and support.

**CONCLUSION**

We used NLP techniques here to classify the protein sequences. We achieved much better results based on accuracy and precision. The CNN and CNN-FRU model has better performance than other models.

| precision | recall | f1-score | support |
|-----------|--------|----------|---------|
| DNA       | 0.86   | 0.79     | 0.82    | 773     |
| Hybrid    | 0.94   | 0.95     | 0.95    | 16316   |
| Protein   | 0.99   | 0.99     | 0.99    | 68916   |
| RNA       | 0.88   | 0.77     | 0.82    | 490     |
| micro avg | 0.98   | 0.98     | 0.98    | 86495   |
| macro avg | 0.92   | 0.88     | 0.99    | 86495   |
| weighted avg | 0.98 | 0.98 | 0.98 | 86495 |

Figure 10. Precision and Recall for the complete dataset.
Although we didn't train the models for long epochs because of time constraints, even training for limited epochs, we obtained high accuracy with CNN. Moreover, the data is highly non-normalized. The models perform much better when the data is normalized. The future work includes using the original dataset and classifying the data into the actual 13 classes of data.

**REFERENCES**

[1] Leo Breiman. Random Forests. journal of Machine Learning, 2001. [2] Geoff Pleiss Zhuang Liu John Hopcroft Gao Huang, Yixuan Li, and Kilian Weinberger. Snapshot ensembles: Train 1, get m for free. In International Conference on Learning Representations, 2017.

[3] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[5] Jie Hou, Badri Adhikari, and Jianlin Cheng. Deepsf: deep convolutional neural network for mapping protein sequences to folds. Bioinformatics, 34(8):1295–1303, 2017.

[6] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomas Mikolov. Fasttext: Compressing text classification models. arXiv preprint arXiv:1612.03651, 2016.

**Table 2:** Random forest results in terms of Precision and Recall

| Label  | Precision | Recall | F1-Score |
|--------|-----------|--------|----------|
| Protein | 0.96      | 0.99   | 0.97     |
| DNA    | 0.90      | 0.80   | 0.85     |
| RNA    | 0.93      | 0.69   | 0.79     |
| Hybrid |           | 0.88   | 0.91     |
| Macro-avg. | 0.93 | .84   | 0.88     |
| Weighted-avg | 0.96 | 0.96 | 0.96     |

**Table 3:** Performance of different networks

| Model     | Train-Acc. | Train-Loss | Val-Acc. | Val-Loss. |
|-----------|------------|------------|----------|-----------|
| CNN       | 98.19%     | 0.0486     | 97.74%   | 0.0819    |
| GRU       | 90.79%     | 0.2691     | 89.70%   | 0.2715    |
| LSTM      | 95.12%     | 0.3982     | 95.14%   | 0.1962    |
| CNN-GRU   | 94.85%     | 0.1509     | 92.74%   | 0.1962    |
| RF [1]    | 95.17%     | 0.4019     | 94.87%   | 0.4906    |
| VGG16 [15]| **99.11 %** | **0.0288** | **98.1 %** | **0.0297** |
[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, Nov 1998.

[9] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, 2015.

[10] Timothy K Lee and Tuan Nguyen. Protein family classification with neural networks, 2016.

[11] Chien-Liang Liu, Wen-Hoar Hsiao, and Yao-Chung Tu. Time series classification with multivariate convolutional neural network. IEEE Transactions on Industrial Electronics, 66(6):4788–4797, 2019.

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In Advances in Neural Information Processing Systems 26, pages 3111–3119. Curran Associates, Inc., 2013.

[13] Ngoc Giang Nguyen, Vu Anh Tran, Duc Luu Ngo, Dau Phan, Fa- vorisen Rosyking Lumbanraja, Mohammad Reza Faisal, Bahrriddin Abapihi, Mamoru Kubo, and Kenji Satou. DNA sequence classification by convolutional neural network. Journal of Biomedical Science and Engineering, 9(05):280, 2016.

[14] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In In EMNLP, 2014.

[15] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations, 2015.

[16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826, 2016.