??THE COMPOSITION OF NEURAL NETWORKS AND FORMAL GRAMMARS FOR TRNA CLASSIFICATION PROBLEMS??

Polina Lunina[(1)], Semyon Grigorev[(2)]

(1) First Institute
Affiliation, address, email

(2) Second Institute
Affiliation, address, email

Keywords: DNN, CNN, Machine Learning, Secondary Structure, Genomic Sequences, Formal Grammars, Parsing.

Abstract. In the wok [1] we proposed a way to combine formal grammars and artificial neural networks for biological sequences processing. We use grammar to encode the primitive features of RNA secondary structure (in contrast to the classical way, when probabilistic grammars are used for modeling the secondary structure of the whole sequence). After that we extract these features by parsing algorithm and process the obtained data by neural network to perform some sort of classification. It was shown that this approach is applicable for real-world data processing and some questions were formulated for future research. In this work we provide further development of the proposed approach. We explore different ways of parsing result representation and appropriate neural network architectures. Also we show that it is possible to create a model that handles original sequences and does not require parsing in practical usage. The idea here is to perform two-staged learning: firstly, we train neural network that classifies parsed data and after that we extend it with a number of input layers that transform the nucleotide sequence into parsing result. We evaluate the proposed improvements on some tRNA classification problems.

1 Introduction

Developing the effective computational methods for genomic sequences analysis is the open problem in bioinformatics. The existing algorithms for sequences classification and subsequences detection adopt different concepts and approaches but the fundamental idea here is that secondary structure of genomic sequences contains the important information about the organisms biological functions. There are different ways of secondary structure formal description such as probabilistic grammars, covariance models and Hidden Markovs Models [2][more refs].

The common problem while dealing with genetic data is the possible presence of different mutations, noises and random surges which requires some sort of probability estimation while modeling the secondary structure. Probabilistic grammars and covariance models provide such functionality along with good expressive possibilities and long-distance connections handling and are successfully used in some tools [3], but building and training accurate grammar or model for predicting the whole secondary structure involves some theoretical and practical difficulties.

In the work [1] we proposed to use the combination of formal grammars and neural networks for biological sequences processing. The key idea is that we use an ordinary context-free grammar to describe only the key secondary structure features and leave the probabilistic analysis to neural network which takes parsing-provided data as an input and performs some sort of classification. Neural networks are a common way to process noisy data and find complex structural patterns, moreover, the performance of neural networks in genetic data processing have already been shown in some works [4, 5]. We demonstrated the applicability of the proposed approach on some real-world problems and in this work we provide some improvements and modifications with evaluation on tRNA classification problems.

2 Proposed solution

In this section we partly repeat the key ideas from our previous work and describe some new modifications of the proposed approach. The basic scenario for its usage is to perform some sort of classification on a genetic sequences dataset.

The first step is to describe the main structural elements of RNA secondary structure (stems and loops) by context-free grammar and extract them by means of parsing. Our solution is not dependent on a specific parsing algorithm, but in evaluation we use a matrix-based version [6] due to its high practical performance and the possibilities in use of parallel computing. The result of a parsing algorithm for the input string w and fixed grammar non-terminal N can be presented as an upper-triangular boolean matrix M_N, where $M_N[i, j] = 1$ if the substring $w[i, j-1]$ is derivable from N. We use such matrices as an input to neural network that is supposed to perform classification of some kind by detecting sufficient features and finding patterns in their appearance. Therefore, we need to transform these boolean matrices to some data structures accepted by neural network. Presently, we came up with two possible ways. The first one is to drop out the bottom left triangle, vectorize it row by row and transform it to the byte vector. This approach reduces the input size, but it requires the equal length of the input sequences, therefore we propose to either cut sequences or add some special symbol for each of them till the definite length. The second way is to represent the matrix as a image: the false bits of matrix as white pixels and the true bits as black ones. This approach makes it possible to process sequences with different length since the images could be easily transformed to a constant size.

The final step is to process the parsing-provided data by an artificial neural network constructed and trained for a specific task. The neural network architecture is unique for each problem, but during the experimental research we worked out some common concepts. For vectorized data we use dense layers because data locality is broken during vectorization and dropout layers with batch normalization to stabilize learning. For image data we use a small number of convolutional layers, then linearization and go to the same architecture as for vectorized data, because convolution layers are suited for features extraction, but in our case it is already done by parsing. In the previous work we performed experiments only on vectorized data and in this work we provide an evaluation on both data formats and compare the results.

The bottleneck of our solution is parsing and the main problem here is following: to use trained model in practice we need to parse the input sequence which is quite time-consuming while working with huge biological databases. To solve this problem we propose to use two-staged learning. Firstly, we train a neural network on the parsed data that performs classification according to a given problem. After that, we extend this neural network by a number of input layers that take the initial nucleotide sequence as an inputs and convert it to the parsing result. So, we build a model that handles sequences and requires parsing only for training the model it is based on. This way we can remove the parsing step from the practical usage of trained model and also improve its accuracy without the additional data generation. In the next section we provide the results of this modification usage.

3 Experiments

We evaluated the proposed approach with described above modifications on tRNA sequences analysis problems: classification of tRNA into 2 classes: eukaryotes and prokaryotes and 4 classes: archaea, bacteria, plants and fungi. For these experiments we used sequences from databases [7, 8]. For both classification problems we took the equal amount of samples for each class and generated vectors and images datasets by parsing toolon the same data with the use of grammar presented in our previous work. After that, we trained neural networks that take parsed data (vectors or images) and perform classification for our problems. Then, for both vector- and image-based models we created the extended neural network which contains two blocks: the first one takes initial tRNA sequence as an input and transforms it to the parsing result by a number of dense layers with batch normalization and the second one copies the sequence of layers from the base model and uses its weights while training. The example of such neural network architecture for 2 classes classification based on vectorized data is presented in figure 1, where the right rectangle is the original model that classifies vectors and the left rectangle is the extension that transforms sequence to vector. For image-based approach we used the same architecture except in that case we removed the convolutional layer from the extended model, thus, at the junction of the blocks the first layer corresponds to linearized image.

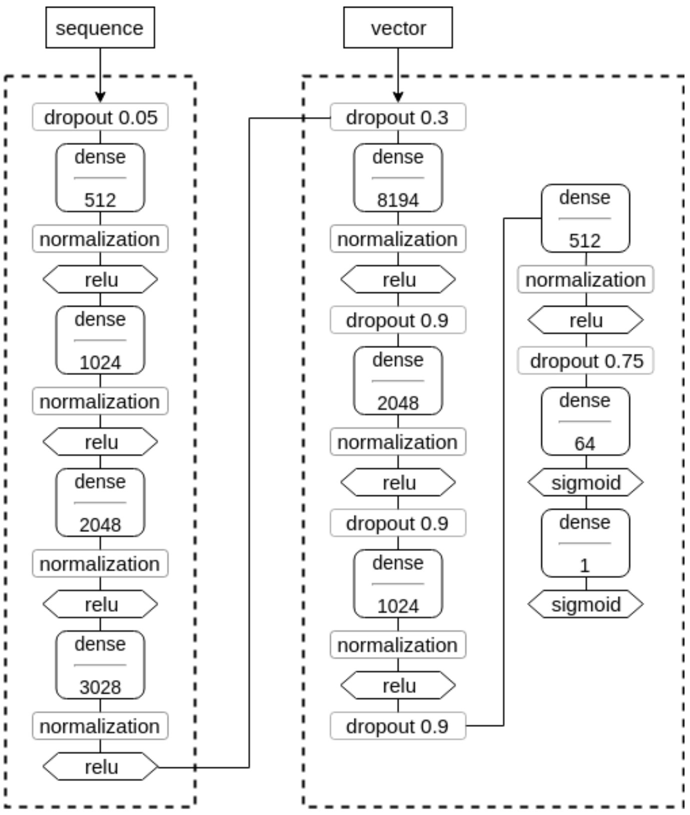

Figure 1: Neural network architecture

Test results for 2 classes (EP) and 4 classes (ABFP) classification problems are presented in the table 1, where base model is a model which handles parsing result (image or vector) and extended model handles tRNA sequences and extends the corresponding base model.

The results show that our approach is applicable to RNA classification problems and both vector- and image-based models can be used along with dense and convolutional layers in neural networks architectures. Moreover, the idea of extended model that handles sequences is proved to be applicable in practice and it demonstrates even higher

Table 1: Test results

Classifier	EP		ABFP	
Approach	Vector-based	Image-based	Vector-based	Image-based
Base model accuracy	94.1%	96.2%	86.7%	93.3%
Extended model accuracy	97.5%	97.8%	96.2%	95.7%
Total samples (train:valid:test)	20000:5000:10000		8000:1000:3000	

performance than the original parsing-based model.

4 Conclusion

We described some new modifications of the proposed approach for biological sequences analysis using the combination of formal grammars and neural networks. We showed that it is possible to represent parsing result as an image and use convolutional layers while processing it with neural network. Also we developed a solution that removes the parsing step from the trained model use and allows to test models on the original RNA sequences. We demonstrated the applicability of the proposed modifications on real-world problems.

Acknowledgments

The research was supported by Russian Science Foundation grant... and a grant from JetBrains Research. ???

References

[1] S. Grigorev and P. Lunina, “The composition of dense neural networks and formal grammars for secondary structure analysis,”

[2] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids.* Cambridge university press, 1998.

[3] E. P. Nawrocki and S. R. Eddy, “Infernal 1.1: 100-fold faster RNA homology searches,” *Bioinformatics*, vol. 29, pp. 2933–2935, Nov 2013.

[4] D. Sherman, “Humidor: Microbial community classification of the 16s gene by training cigar strings with convolutional neural networks,” 2017.

[5] S. Higashi, M. Hungria, and M. Brunetto, “Bacteria classification based on 16s ribosomal gene using artificial neural networks,” in *Proceedings of the 8th WSEAS International Conference on Computational intelligence, man-machine systems and cybernetics*, pp. 86–91, 2009.

[6] R. Azimov and S. Grigorev, “Context-free path querying by matrix multiplication,” in *Proceedings of the 1st ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*, GRADES-NDA ’18, (New York, NY, USA), pp. 5:1–5:10, ACM, 2018.

[7] “Genomic trna database [ ].”

[8] “trnadb-ce [ ].”