16s rRNA Detection by Using Context-Free Grammars and Neural Networks

Semyon Grigorev[1], Polina Lunina[1]

[1] *Saint Petersburg State University, JetBrains, St. Petersburg, Russia*

E-mail: *semen.grigorev@jetbrains.com*

Motivation

Algorithms that can efficiently and accurately identify and classify bacterial taxonomic hierarchy have become a focus in computational genetics. The idea that secondary structure of genomic sequences is sufficient for solving the detection and classification problems lies at the heart of many tools.The secondary structure can be specified in terms of formal grammars. The sequences obtained from the real bacteria usually contain a huge number of mutations and "noise" which renders precise methods impractical. Probabilistic grammars and covariance models (CMs) are a way to take the noise into account [1]. For example, CMs are successfully used in the Infernal tool. Neural networks is another way to deal with "noisy" data. The works [2, 3] utilize neural networks for 16s rRNA processing and demonstrate promising results. In our work we want to combine ordinary context-free grammars nad neural networks and utilize this composition for 16s rRNA detecting and classification.

Results

- We combine neural networks and ordinary context-free grammars to classify genomic sequences by using its secondary structure information. We extract features by using the ordinary (not probabilistic) context-free grammar and use the dense neural network for features processing.
- We evaluate the proposed approach for 16s rRNA detection. Current accuracy is 90% for validation set (up to 81000 sequences from NCBI and Green Genes databases), thus we conclude that our approach is applicable.
- Our solution provides an ability to use GPGPU and multi-core systems for sequences processing which can be useful for large biological data analysis.

Solution Overview

Grammar

```
s1: stem<s0> any
a_0_7 : any*[2..10]
s0: a_0_7 | a_0_7 stem<s0> s0
any: A | U | C | G
stem1<s>: A s U | G s C | U s A | C s G
stem2<s>: stem1< stem1<s> >
stem<s>:
    A stem<s> U
    | U stem<s> A
    | C stem<s> G
    | G stem<s> C
    | stem1< stem2<s> >
```

Fixed context-free grammar describes features of secondary structure and can be tuned in order to increase result quality.

Sequences

Genoms parts of fixed length. Current length is 512. Length is variable parameter and can be changed in order to increase quality of solution. Each sequence is treated as a text in $\{A, U, G, C\}$ alphabet.

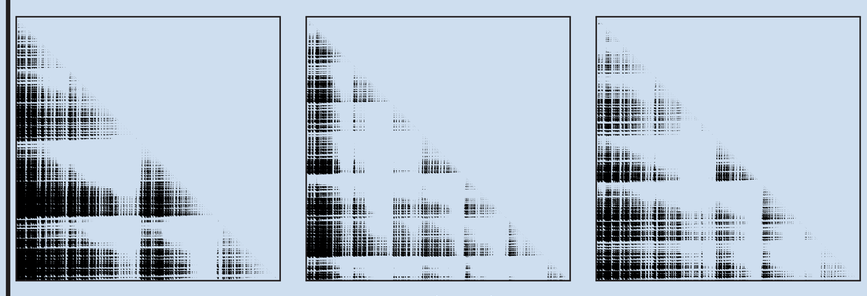
Result of classification

Currently we implement only simple binary classifier that separates 16s and non-16s sequences. One of interesting question is may this classifier be used for chimeric sequences filtering. We hope that it is possible because "global" secondary structure of chimeric sequences should be broken.

Parser

Parser is features extractor: it extracts features of given squence secondary structure. Parsing algorithm is based on the algorithm [4], so the grammar can be extended with conjunctive rules [5] for pseudoknots description.Parser performance is a bottleneck of our solution. Current implementation utilizes GPGPU and should be optimized in future.

Matrices



Parsing result is boolean (0-1) matrix which represents secondary structure features for sequence ω: cell $[i, j]$ contains 1 iff $\omega.[j, i]$ is derivable from `s1` and 0 in other case. These matrices can be handled as images or binary vectors by using respective types of neural networks (convolutional or binary) and one of topics for future research is to select better approach. Currently we compress each matrix to integer vector and use dense newral network.

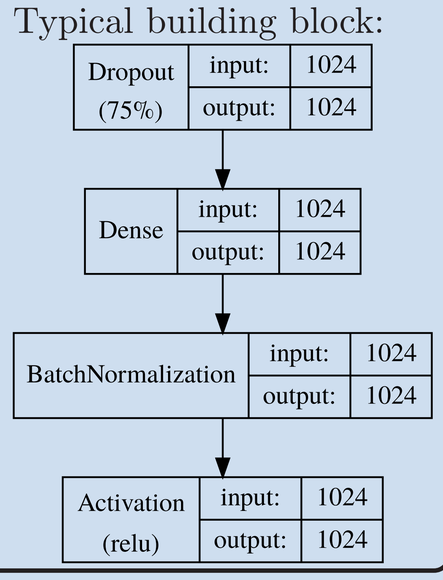
Vectors

We use line-by-line compressed matrix representation: sequence of 32 cells (bits) is compressed to unsigned integer. Top right triangle of matrix is always empty, so can be ignored. We hope that compression to 16 bit integer or byte may decrease complexity of neural network and improve result quality, but it requires significantly more memory on GPGPU which can be serious technical problem.

Neural Network

We use dense neural network with 14 dense layers. Almost all of them are wrapped with dropout (up to 75%) and batch normalization layers for learning stabilization. Our network is trained on up to 310000 sequences of length 512: positive (16s rRNA) from NCBI database, negative (non-16s) from Green Genes database. Current accuracy for validation set (up to 81000 sequences) is 90%.

Typical building block:



Future Research

The presented is a work in progress. The ongoing experiment is finding all instances of 16s rRNA in full genomes. Also we plan to use the proposed approach for the filtration of chimeric sequences and the classification. Composition of our approach with other methods and tools as well as grammar tuning and detailed performance evaluation may improve the applicability for the real data processing.

Acknowledgments

The research was supported by the Russian Science Foundation grant 18-11-00100 and a grant from JetBrains Research.

Information

All materials available on GitHub: https://github.com/YaccConstructor

References

[1] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[2] Douglas Sherman. Humidor: Microbial community classification of the 16s gene by training cigar strings with convolutional neural networks. 2017.

[3] Susan Higashi, Mariangela Hungria, and MADC 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*, pages 86–91, 2009.

[4] Rustam Azimov and Semyon 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)*, page 5. ACM, 2018.

[5] Alexander Okhotin. Conjunctive grammars. *Journal of Automata, Languages and Combinatorics*, 6(4):519–535, 2001.