CIBB 2019

ON SECONDARY STRUCTURE ANALYSIS BY USING FORMAL GRAMMARS AND ARTIFICIAL NEURAL NETWORKS

Semyon Grigorev, **Polina Lunina**

JetBrains Research, Programming Languages and Tools Lab
Saint Petersburg University

September 6, 2019

Genomic sequences analysis

- Secondary structure handling
  - Covariance models
  - Hidden Markov Models
  - Probabilistic grammars
- Probability estimation for noisy data processing
  - Covariance models
  - Probabilistic grammars
  - Neural networks

Proposed solution: the earlier ideas

Our previous work: *S. Grigorev, P. Lunina* "The Composition of Dense Neural Networks and Formal Grammars for Secondary Structure Analysis"

- Context-free grammar to encode only basic secondary structure features
  - Recursive compositions of stems
  - Conventional base pairing
- Parsing for these features extraction
  - Matrix-based parsing algorithm
  - Parsing result for sequence w and nontermianl N is a boolean matrix M_N, where $M_N[i, j] = 1$, iff $w[i, j-1]$ is derivable from N
- Neural network to solve a given sequences analysis task
  - Input: parsing result linearized and compressed into a byte vector
  - Dense and dropout layers with batch normalization

Proposed solution: the earlier ideas

Our previous work: *S. Grigorev, P. Lunina* "The Composition of Dense Neural Networks and Formal Grammars for Secondary Structure Analysis"

- **Context-free grammar to encode only basic secondary structure features**
  - Recursive compositions of stems
  - Conventional base pairing
- Parsing for these features extraction
  - Matrix-based parsing algorithm
  - Parsing result for sequence w and nontermianl N is a boolean matrix M_N, where $M_N[i, j] = 1$, iff $w[i, j-1]$ is derivable from N
- Neural network to solve a given sequences analysis task
  - Input: parsing result linearized and compressed into a byte vector
  - Dense and dropout layers with batch normalization

Grammar

```
s1: stem<s0>
any_str: any_smb*[2..10]
any_smb: A | U | C | G
stem1<s>:                    \\ stem of height exactly 1
      A s U | U s A | C s G | G s C
stem3<s>:                    \\ stem of height exactly 3
      stem1< stem1< stem1<s> > >
stem<s>:                     \\ stem of height 3 or more
      A stem<s> U
    | U stem<s> A
    | C stem<s> G
    | G stem<s> C
    | stem3<s>
s0: any_str | any_str stem<s0> s0
```

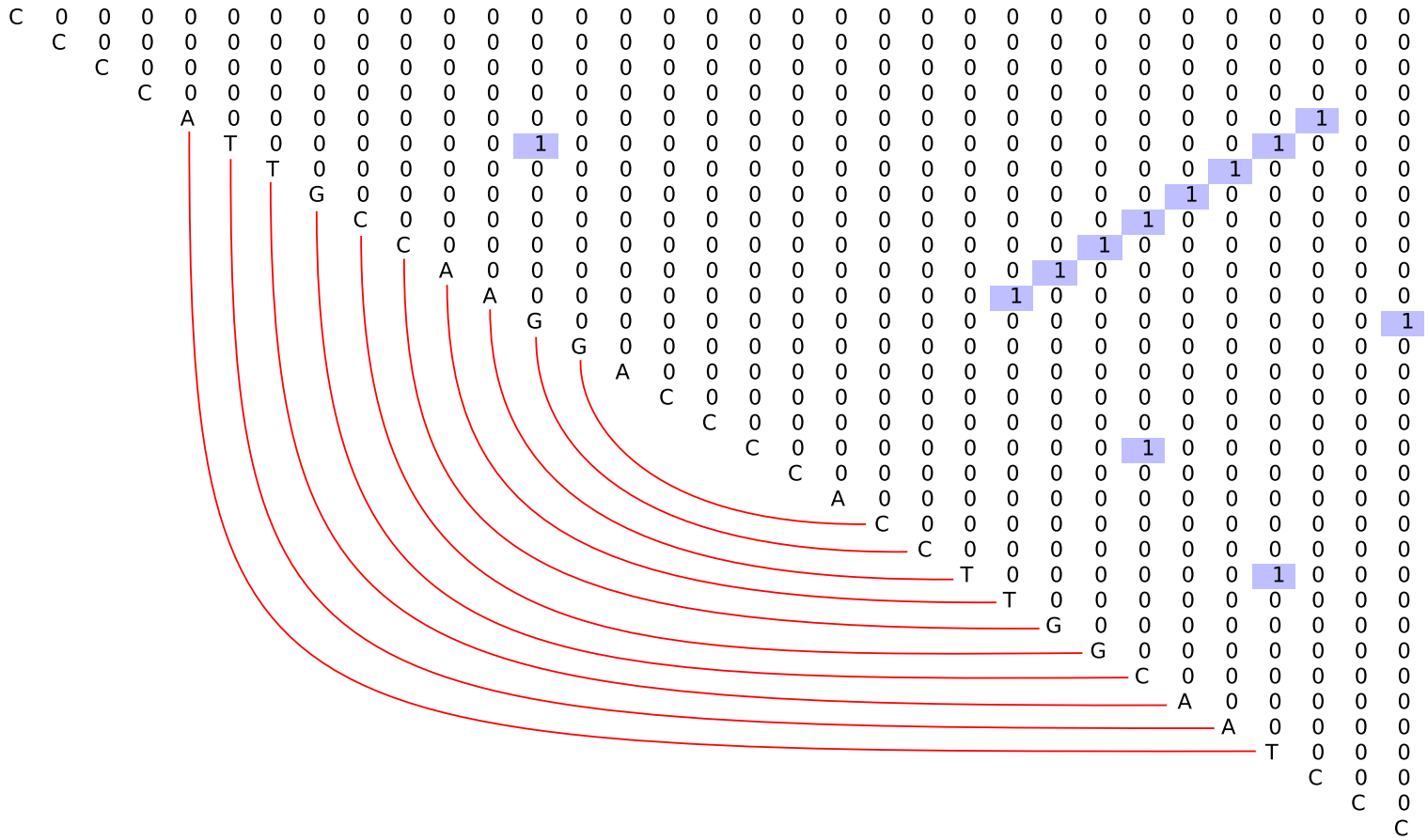
Proposed solution: the earlier ideas

Our previous work: *S. Grigorev, P. Lunina* "The Composition of Dense Neural Networks and Formal Grammars for Secondary Structure Analysis"

- Context-free grammar to encode only basic secondary structure features
  - Recursive compositions of stems
  - Conventional base pairing
- **Parsing for these features extraction**
  - Matrix-based parsing algorithm
  - Parsing result for sequence w and nontermianl N is a boolean matrix M_N, where $M_N[i, j] = 1$, iff $w[i, j-1]$ is derivable from N
- Neural network to solve a given sequences analysis task
  - Input: parsing result linearized and compressed into a byte vector
  - Dense and dropout layers with batch normalization

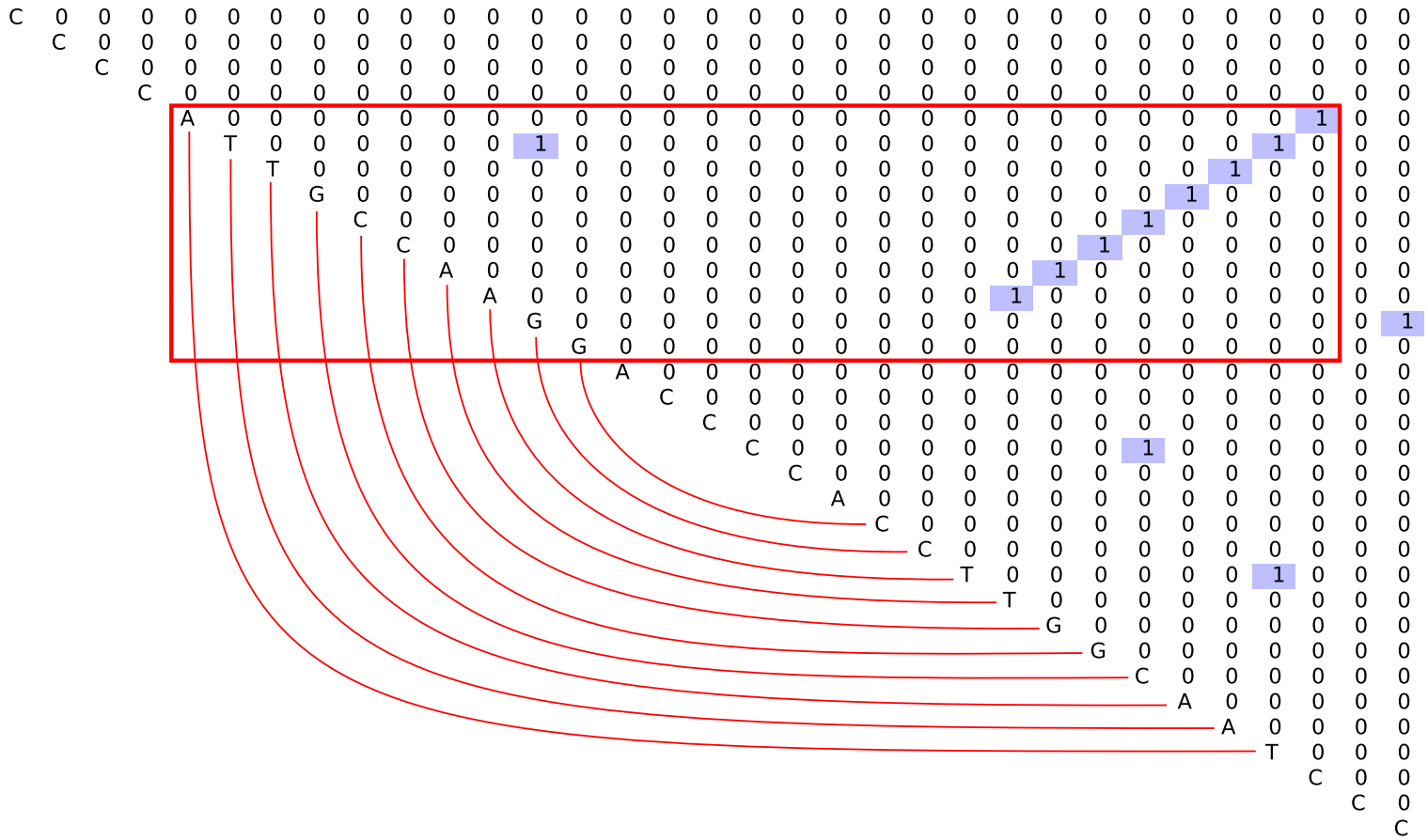
Example

CCCCATTGCCAAGGACCCCACCTTGGCAATCCC



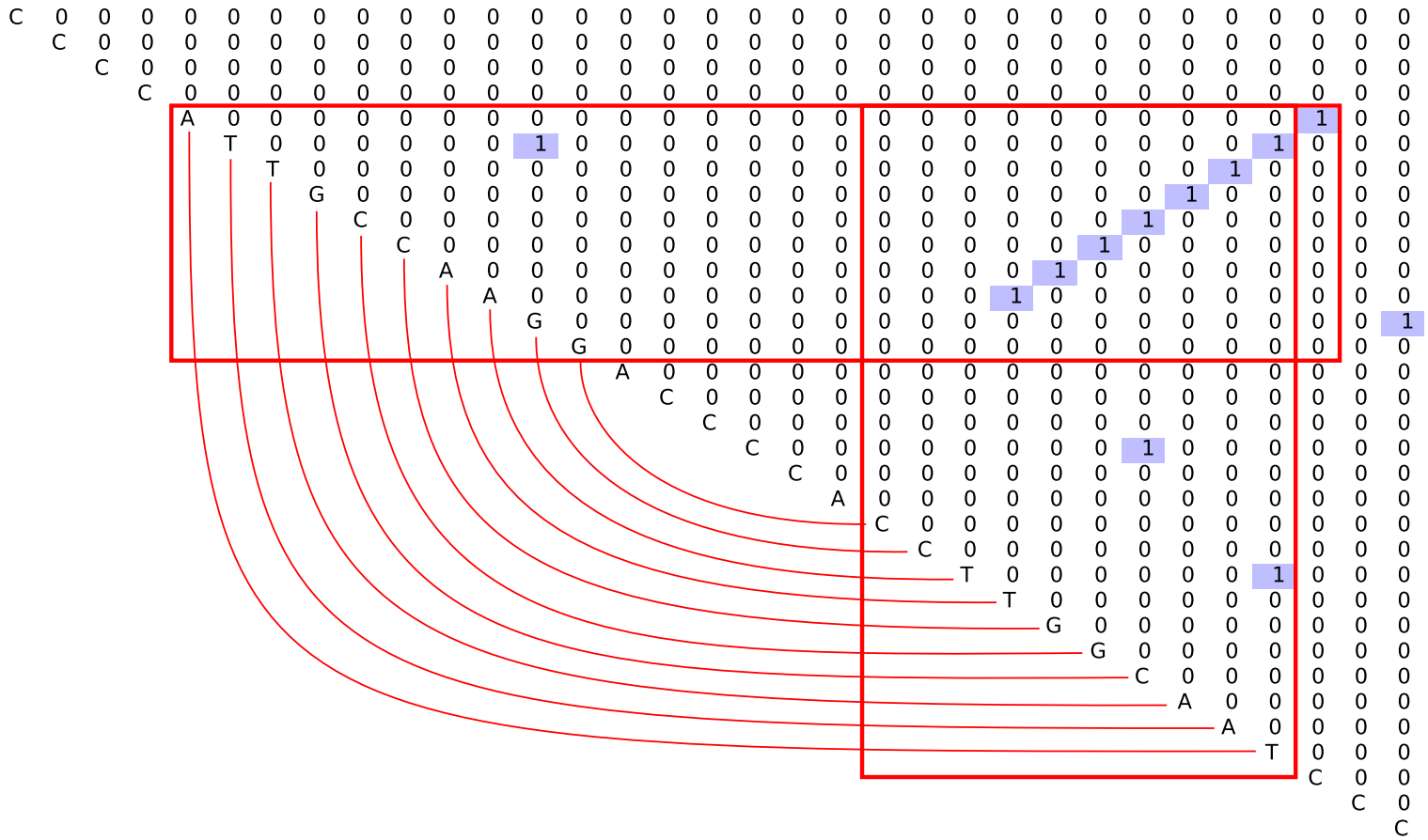
Example

CCCCATTGCCAAGGACCCCACCTTGGCAATCCC



Example

CCCCATTGCCAAGGACCCCACCTTGGCAATCCC



Proposed solution: the earlier ideas

Our previous work: *S. Grigorev, P. Lunina* "The Composition of Dense Neural Networks and Formal Grammars for Secondary Structure Analysis"

- Context-free grammar to encode only basic secondary structure features
  - Recursive compositions of stems
  - Conventional base pairing
- Parsing for these features extraction
  - Matrix-based parsing algorithm
  - Parsing result for sequence w and nontermianl N is a boolean matrix M_N, where $M_N[i, j] = 1$, iff $w[i, j-1]$ is derivable from N
- **Neural network to solve a given sequences analysis task**
  - Input: parsing result linearized and compressed into a byte vector
  - Dense and dropout layers with batch normalization

Improvements (1)

Problem: vectorization breaks data locality which increases network learning time

Solution:

- Represent parsing result as an image
- Use convolutional layers for these images processing
- Compare image- and vector-based networks on the same data

Parsing results representation

Matrices

$$\begin{pmatrix} 0 & 1 & 0 & \dots & 1 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \dots & 1 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & 0 \end{pmatrix}$$

Parsing result is a boolean matrix M which represents secondary structure features for sequence ω: $M[i,j] = 1 \iff \mathtt{s1} \xrightarrow{*} \omega[i,j]$, and 0 otherwise.

Vectors

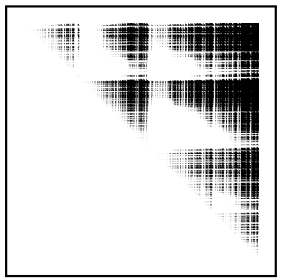
`[1,0,...,1,1,...,0,...,1,...]`

⇓

`[84,128,...]`

Line-by-line compressed matrix representation: sequence of 8 cells (bits) is compressed into a byte. Bottom left triangle of the matrix is always empty, so can be ignored. Requires the equal length of the input sequences and breaks the data locality.

Images



The false bits of the matrix are represented as white pixels and the true bits as black ones. This approach makes it possible to process sequences of different lengths since the images can be transformed to a specified size. Data locality is preserved.

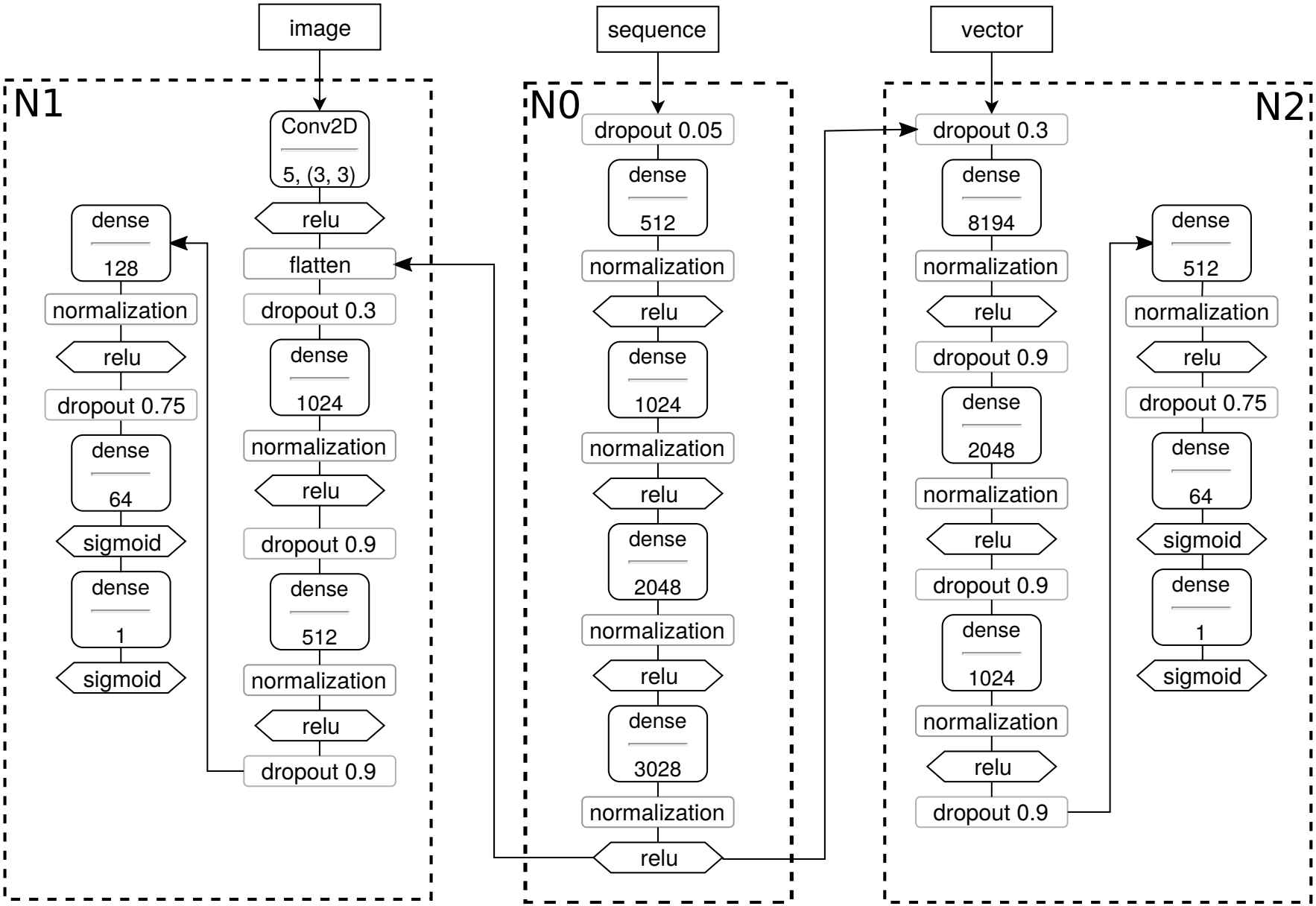
Improvements (2)

Problem: parsing is a time-consuming operation which complicates the use of trained models

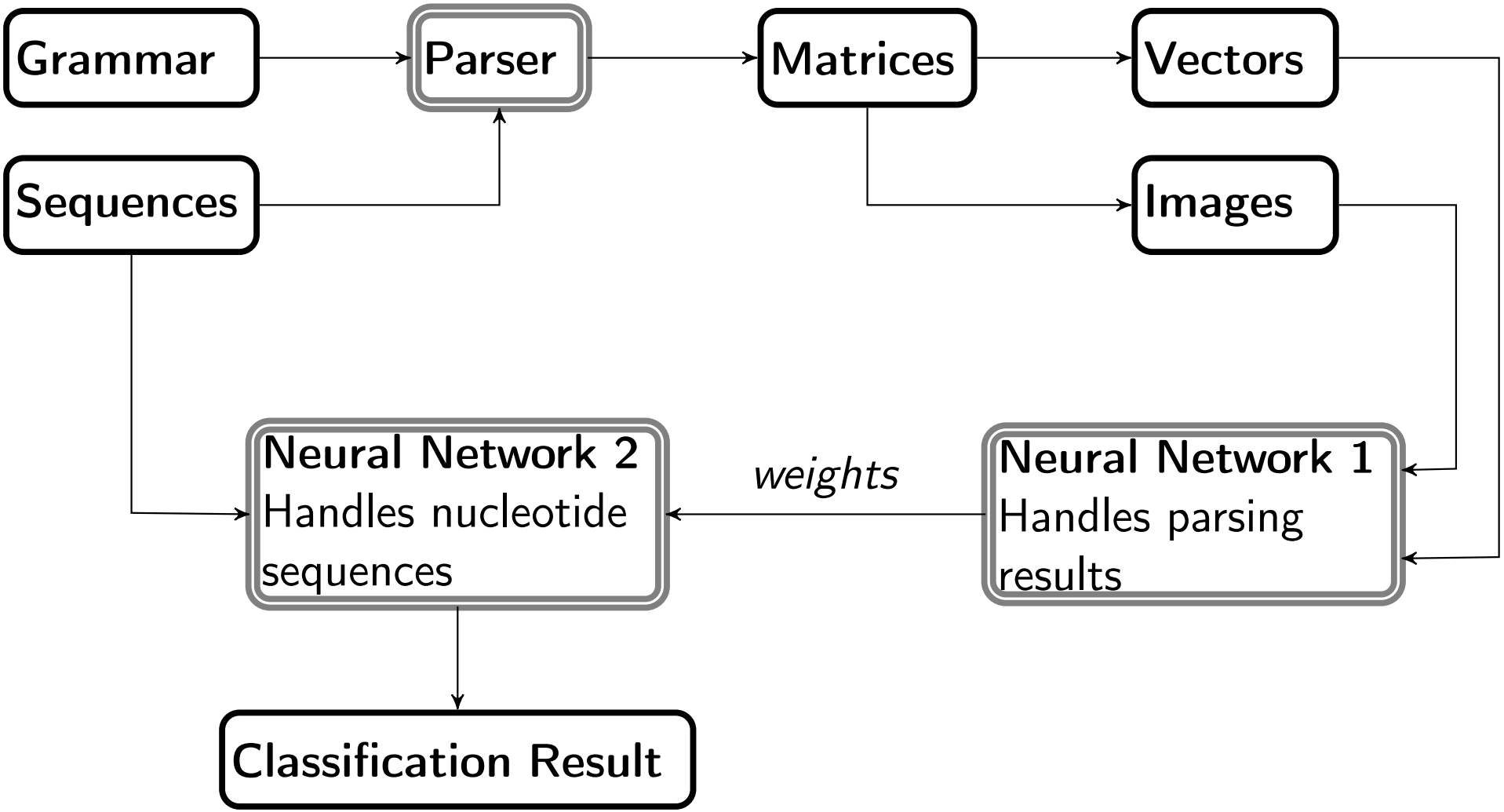
Solution:

- Create a network which handles initial sequences
- Use two-staged learning
  - Train network that takes images or vectors as an input and performs classification according to a given problem
  - Extend it by a number of input layers that take the initial nucleotide sequence as an input and convert it to the parsing result which is handled by the pretrained layers

Neural networks



Final solution structure



Evaluation

- tRNA sequences analysis tasks
  - Classification into two classes: eukaryotes and prokaryotes
  - Classification into four classes: archaea, bacteria, plants and fungi
- Tools
  - Parsing: YaccConstructor platform
  - Neural networks: Keras, Tensorflow
- Databases
  - tRNADB-CE
  - Genomic tRNA database

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%
Samples for train:valid:test	20000:5000:10000 (57%:14%:29%)		8000:1000:3000 (67%:8%:25%)	

Classifier	Class	Vector-based approach		Image-based approach	
		precision	recall	precision	recall
EP	prokaryotic	95.8%	99.4%	96.2%	99.4%
	eukaryotic	99.4%	95.6%	99.4%	99.5%
ABFP	archaeal	91.1%	99.2%	91.6%	98.5%
	bacterial	96.6%	95.1%	95.2%	95.5%
	fungi	98.5%	94.9%	97.5%	94.3%
	plant	99.4%	95.7%	99.2%	94.7%

Conclusion

- We propose some modifications of our approach for biological sequences analysis
  - Parsing result in a form of image can be handled by convolutional layers and it decreases training time
  - It is possible to remove the parsing step from the trained model use which allows to run models on the original RNA sequences
- These modification improve the performance of the solution and are applicable for real-world problems

Future work

- Other RNA sequences analysis tasks
  - 16s rRNA classification
  - Chimeric sequences filtration
- Secondary structure prediction by using generative networks
- Proteomic sequences processing

Contact Information

- Semyon Grigorev:
  - s.v.grigoriev@spbu.ru
  - Semen.Grigorev@jetbrains.com
- Polina Lunina: lunina_polina@mail.ru

- Implementation:
  https://github.com/LuninaPolina/SecondaryStructureAnalyzer

Thanks!