CIBB 2019

On Secondary Structure Analysis by Using Formal Grammars and Artificial Neural Networks

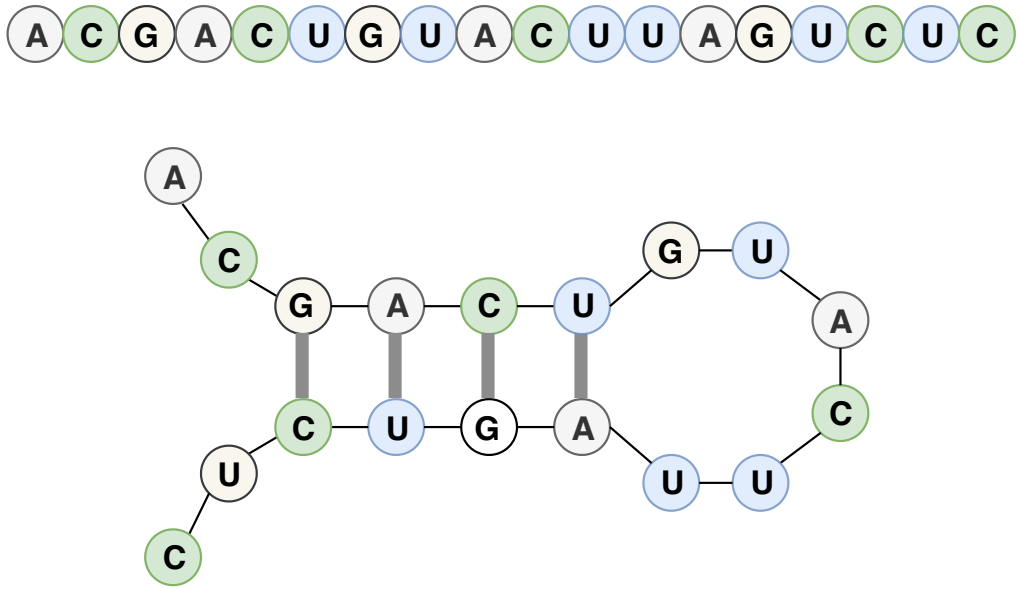
Semyon Grigorev, **Polina Lunina**

JetBrains Research, Programming Languages and Tools Lab
Saint Petersburg University

September 6, 2019

Genomic Sequences Analysis

- Problems
  - Genomic sequences classification
  - Subsequences detection
  - Secondary structure prediction
- Secondary structure handling
  - Covariance models
  - Hidden Markov Models
  - Probabilistic grammars
- Probability estimation for noisy data processing



Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Result of classification

Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Vectors

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

$\Downarrow$

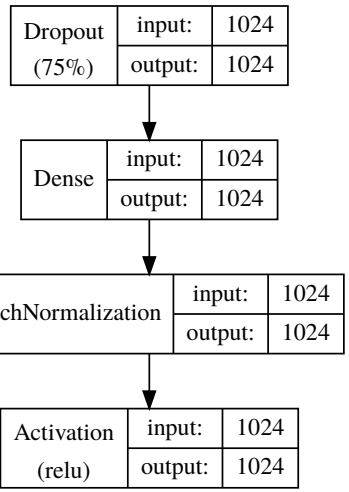
`[0,1,0,1,0,1,0,0,1,0]`

$\Downarrow$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Vectors

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

$\Downarrow$

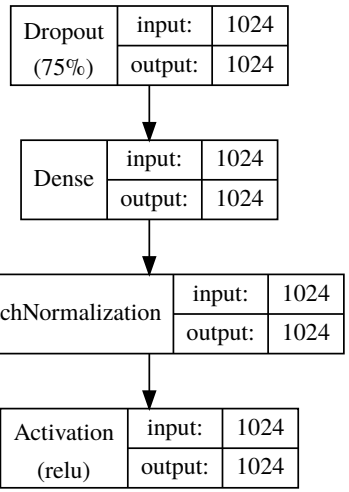
`[0,1,0,1,0,1,0,0,1,0]`

$\Downarrow$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



Result of classification

Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Result of classification

Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Vectors

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

$\Downarrow$

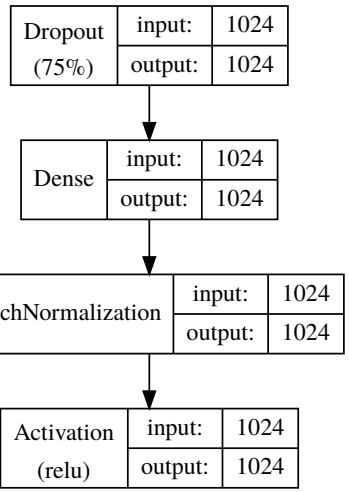
`[0,1,0,1,0,1,0,0,1,0]`

$\Downarrow$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Result of classification

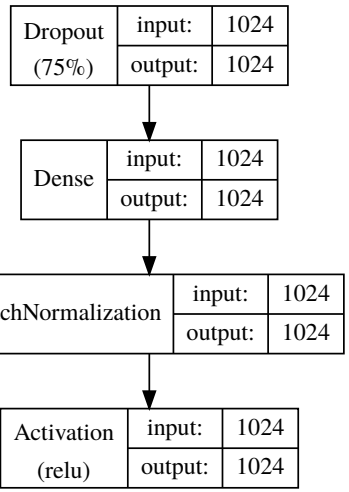
Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



Vectors

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

$\Downarrow$

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

$\Downarrow$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Result of classification

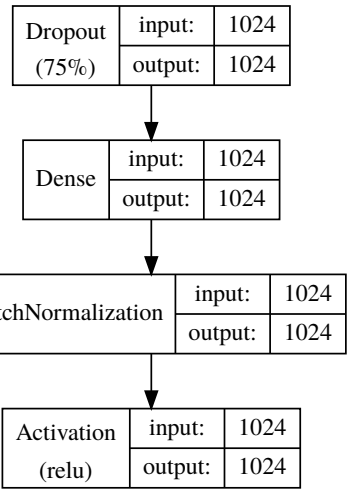
Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



Vectors

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

$\Downarrow$

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

$\Downarrow$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Result of classification

Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Vectors

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

$\Downarrow$

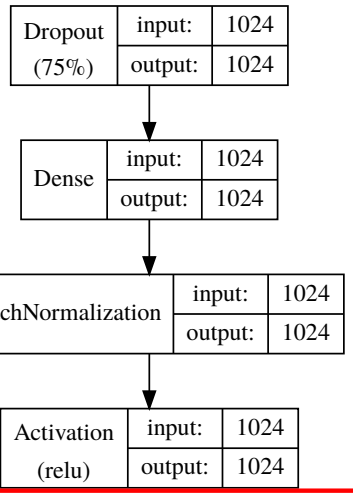
`[0,1,0,1,0,1,0,0,1,0]`

$\Downarrow$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



Solution Structure

Grammar
Formal grammar describes the features of secondary structure.

Sequences
Each sequence is treated as a text in $\{A, C, G, U\}$ alphabet.

Result of classification

Parser
Parser extracts features of the secondary structure. Implementation of parsing algorithm is based on matrix multiplications and utilizes GPGPU.

Matrices

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

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

Vectors

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

$\Downarrow$

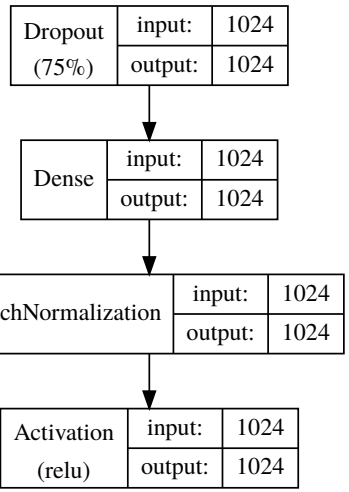
`[0,1,0,1,0,1,0,0,1,0]`

$\Downarrow$

`[84,128]`

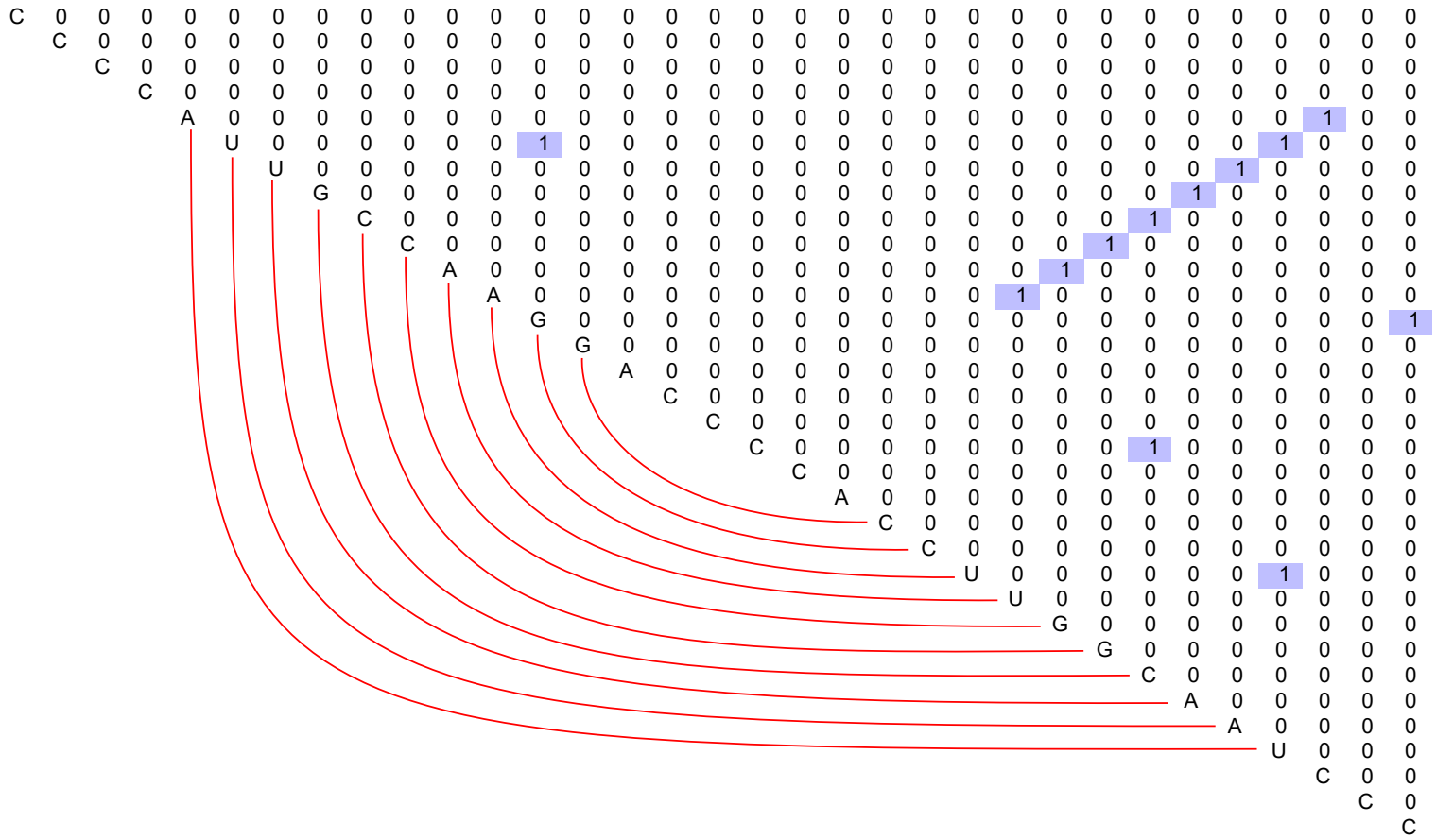
Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and can be ignored.

Neural Network
Dense neural network with more than 10 dense layers. Aggressive dropout and batch normalization for learning process stabilization. Typical building block:



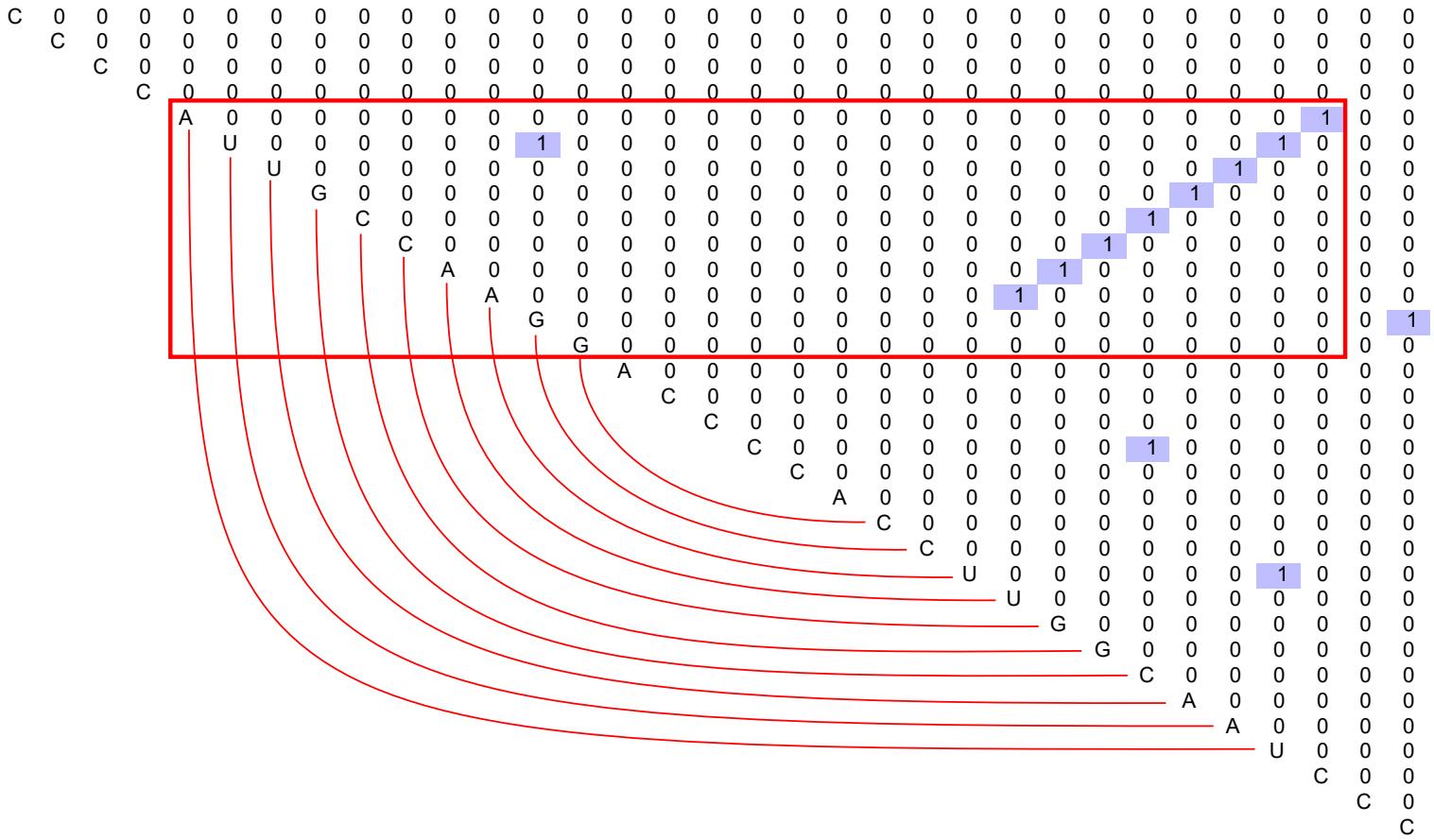
Example

CCCCAUUGCCAAGGACCCCACCUUGGCAAUCCC



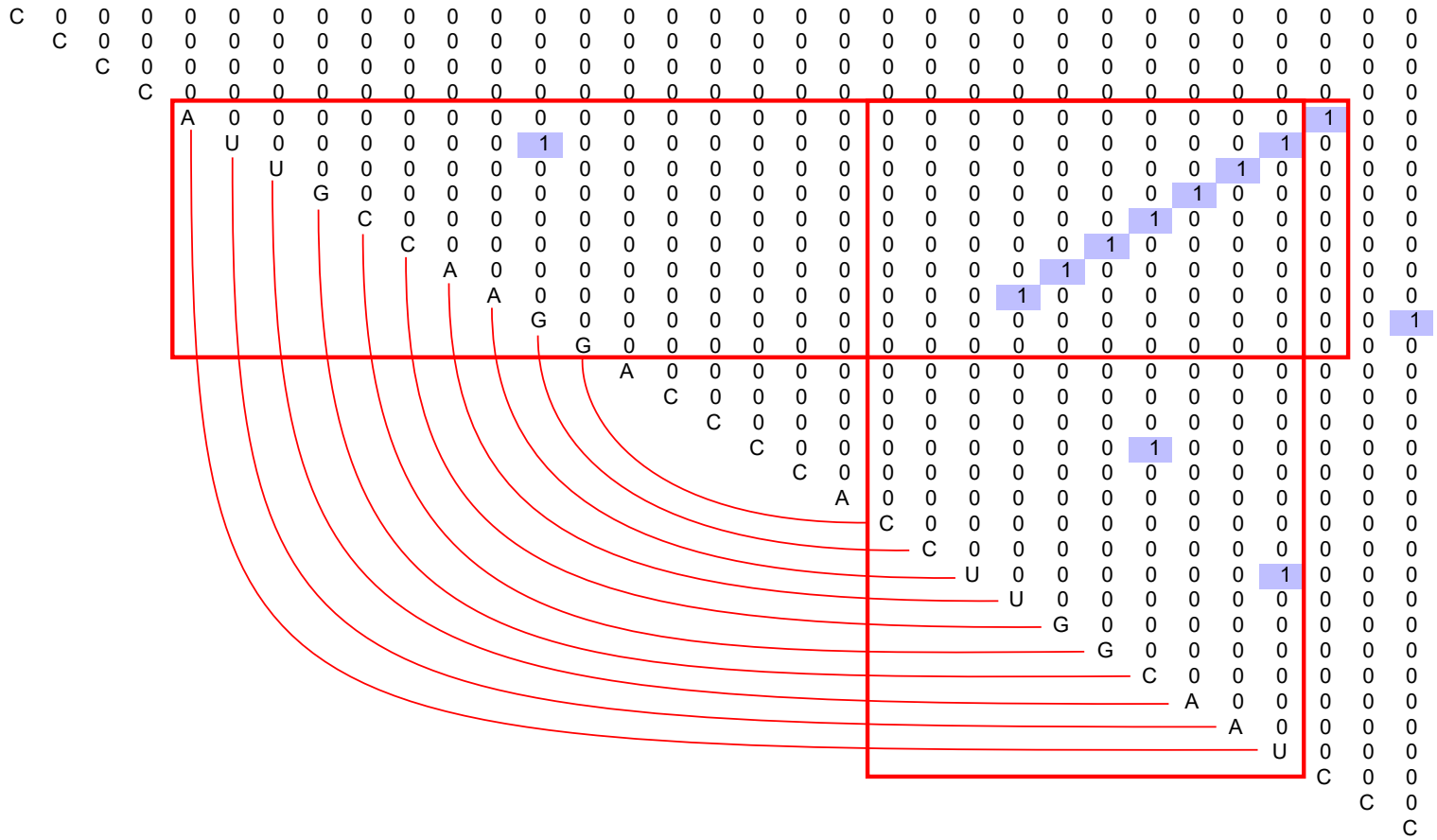
Example

CCCCAUUGCCAAGGACCCCACCUUGGCAAUCCC



Example

CCCCAUUGCCAAGGACCCCACCUUGGCAAUCCC



Data Locality Preservation

Problem: data locality is broken during vectorization

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

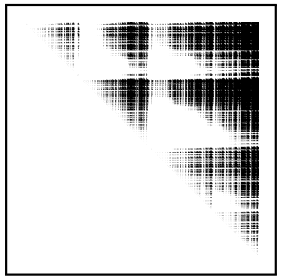
`[1,0,...,1,1,...,0,...,1,...]`

⇓

`[84,128,...]`

Line-by-line compressed matrix representation. Bottom left triangle of the matrix is always empty and 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. It is possible to process sequences with different lengths since the images can be transformed to a specified size. Data locality is preserved.

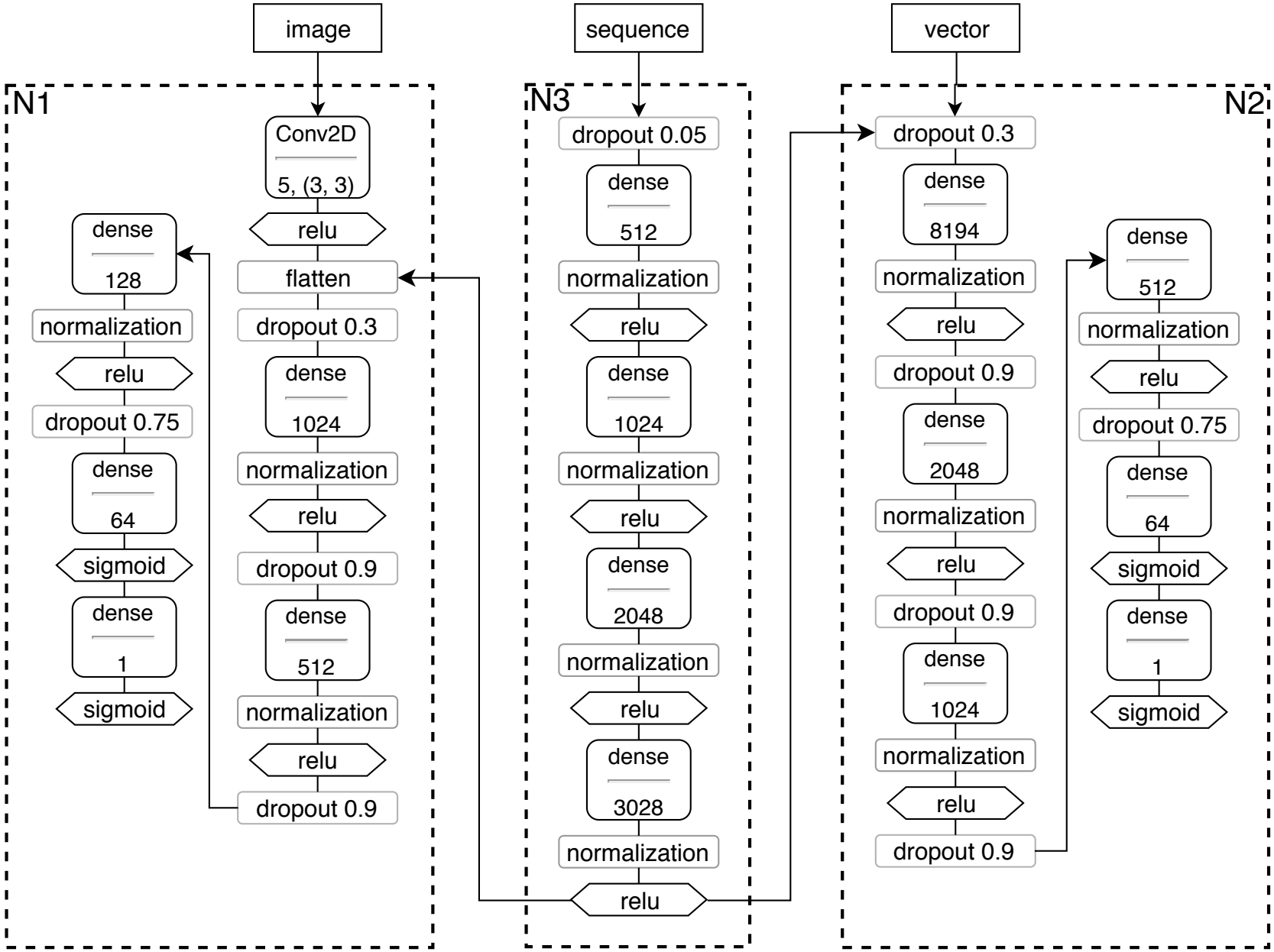
Parsing Elimination

Problem: parsing is the bottleneck of the solution

Solution:

- Create a network which handles initial sequences
- Use two-staged learning
  - Train network on images or vectors for a given problem
  - Extend it by several input layers that take the initial nucleotide sequence as an input and convert it to the parsing result

Neural Networks



Evaluation

- tRNA sequences analysis tasks
  - Classification into two classes: eukaryotes and prokaryotes
  - Classification into four classes: archaea, bacteria, plants and fungi
- Databases
  - tRNADB-CE
  - Genomic tRNA database

Results

EP — eukaryotes/prokaryotes task
ABFP — archaea/bacteria/plants/fungi task

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%)	

Results

EP — eukaryotes/prokaryotes task
ABFP — archaea/bacteria/plants/fungi task

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

- The modifications of our approach for biological sequences analysis were implemented
  - Parsing result in a form of image can be handled by convolutional layers
  - The parsing step can be removed from the trained model use which allows to run models on the original RNA sequences
- These modification improve the quality of the solution
- The improved version is 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
- The use of deep convolutional networks for secondary structure analysis

Contact Information

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

- Secondary structure analyzer project: https://research.jetbrains.org/groups/plt_lab/projects?project_id=43

Thanks!