CIBB 2019

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

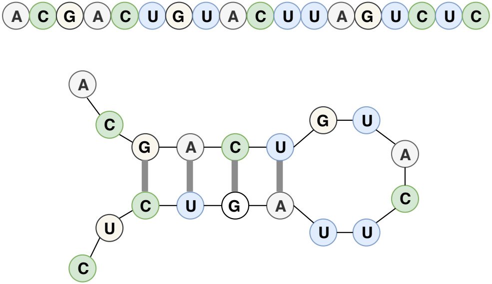
Polina Lunina, Semyon Grigorev

JetBrains Research, Programming Languages and Tools Lab
Saint Petersburg University

September 6, 2019

Genomic Sequences Analysis

- Problems
  - Genomic sequences classification
  - Subsequences detection
- Secondary structure handling
- Probability estimation for noisy data processing



Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

Matrices

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

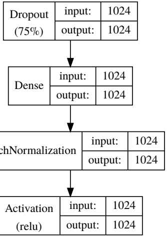
Parsing result for a sequence ω is a boolean matrix M, where $M[i,j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i,j]$.

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 is empty, so, can be ignored.

Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.



Result

Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

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 $M[i,j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i,j]$.

Vectors

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

$$\Downarrow$$

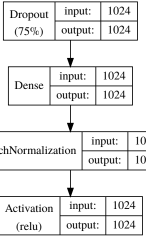
`[0,1,0,1,0,1,0,0,1,0]`

$$\Downarrow$$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle is empty, so, can be ignored.

Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.



Result

Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

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 $M[i,j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i,j]$.

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 is empty, so, can be ignored.

Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.

Dropout (75%)	input:	1024
	output:	1024
Dense	input:	1024
	output:	1024
BatchNormalization	input:	1024
	output:	1024
Activation (relu)	input:	1024
	output:	1024

Result

Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

Matrices

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

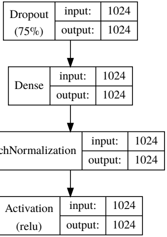
Parsing result for a sequence ω is a boolean matrix M, where $M[i,j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i,j]$.

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 is empty, so, can be ignored.

Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.



Result

Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

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 $M[i,j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i,j]$.

Vectors

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

$$\Downarrow$$

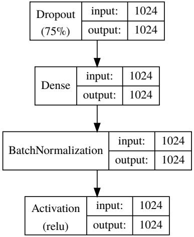
`[0,1,0,1,0,1,0,0,1,0]`

$$\Downarrow$$

`[84,128]`

Line-by-line compressed matrix representation. Bottom left triangle is empty, so, can be ignored.

Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.



Result

Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

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 $M[i, j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i, j]$.

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 is empty, so, can be ignored.

Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.

Dropout (75%)	input:	1024
	output:	1024

Dense	input:	1024
	output:	1024

BatchNormalization	input:	1024
	output:	1024

Activation (relu)	input:	1024
	output:	1024

Result

Solution Structure

Ordinary Grammar
Describes the features of secondary structure.

Sequences
Text in the $\{A, C, G, U\}$ alphabet.

Parser
Extracts the features from sequence.

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 $M[i,j] = 1 \iff \mathtt{s} \xrightarrow{*} \omega[i,j]$.

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 is empty, so, can be ignored.

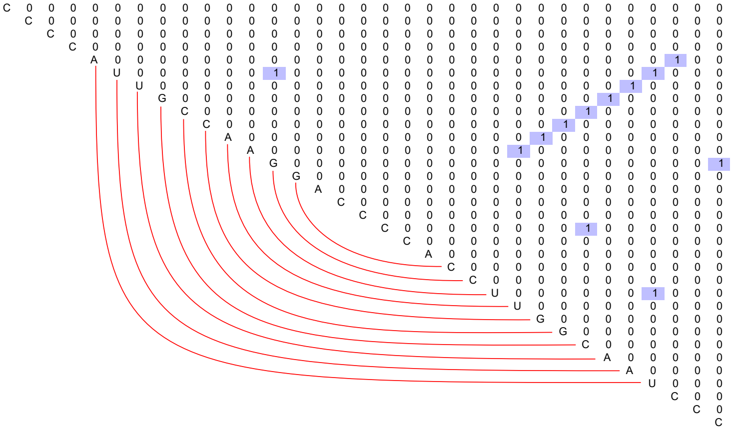
Neural Network
Dense neural network with aggressive dropout and batch normalization for learning process stabilization.

Dropout (75%)	input:	1024
	output:	1024
Dense	input:	1024
	output:	1024
BatchNormalization	input:	1024
	output:	1024
Activation (relu)	input:	1024
	output:	1024

Result

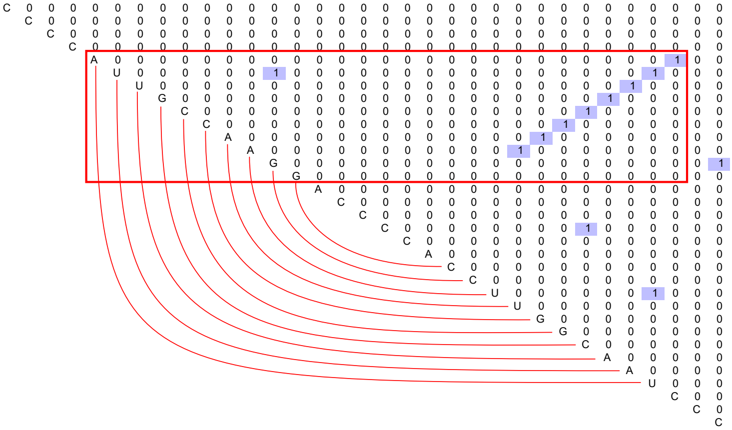
Example

CCCCAUUGCCAAGGACCCCACCUUGGCAAUCCC



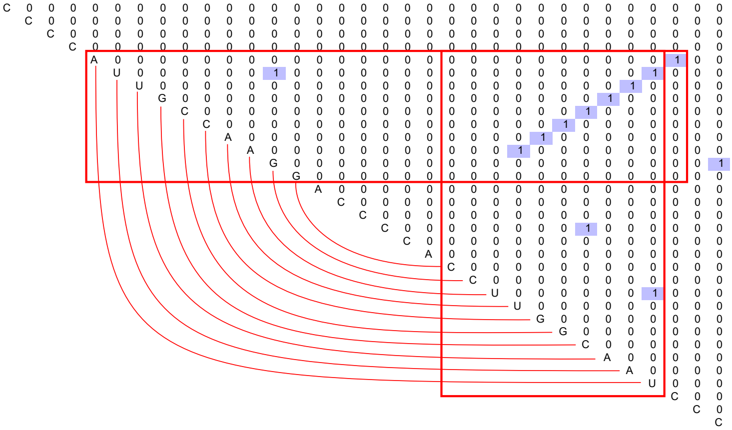
Example

CCCCAUUGCCAAGGACCCCACCUUGGCAAUCCC



Example

CCCCAUUGCCAAGGACCCCACCUUGGCAAUCCC



Data Locality Preservation

Problem: data locality is broken during vectorization

Solution:

- Represent parsing result as an image
- Use convolutional layers for such images processing

Parsing Results Representation

Matrices

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

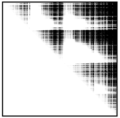
Parsing result for sequence ω is a boolean matrix M, where $M[i,j] = 1 \iff \mathrm{s} \xrightarrow{*} \omega[i,j]$.

Vectors

$$\texttt{[0,1,0,1,0,1,0,0,1,0]}$$
$$\Downarrow$$
$$\texttt{[84,128]}$$

Line-by-line compressed matrix representation. Bottom left triangle is empty, so, can be ignored. It requires the equal length of the sequences and breaks 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 and data locality is preserved.

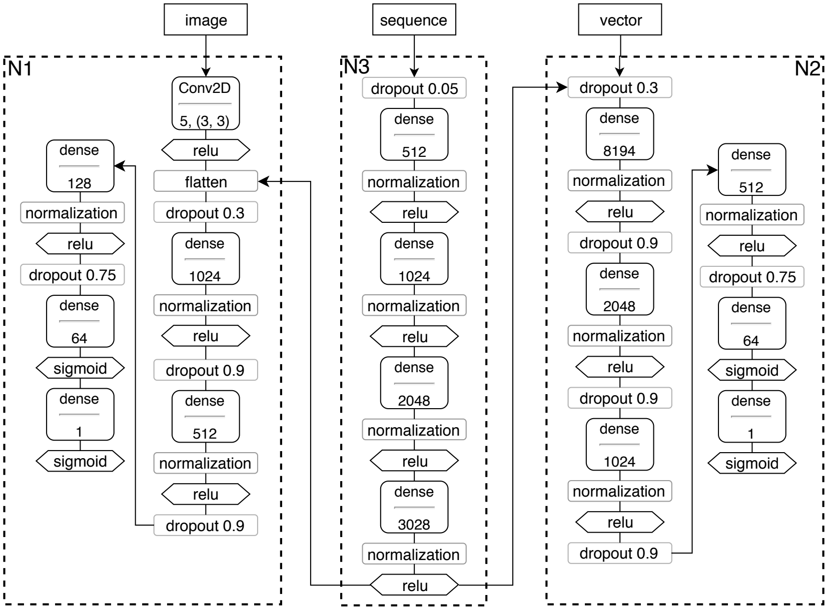
Parsing Elimination

Problem: parsing is a time-consuming operation

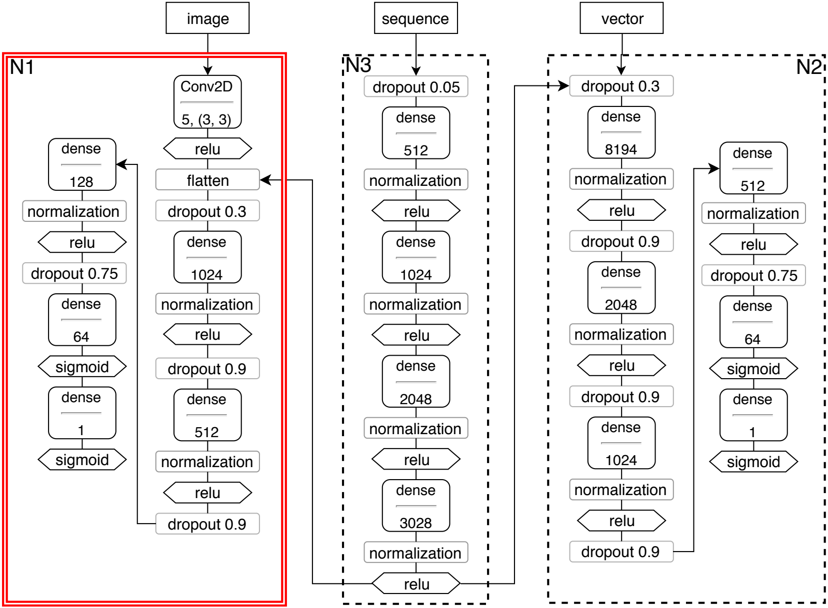
Solution:

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

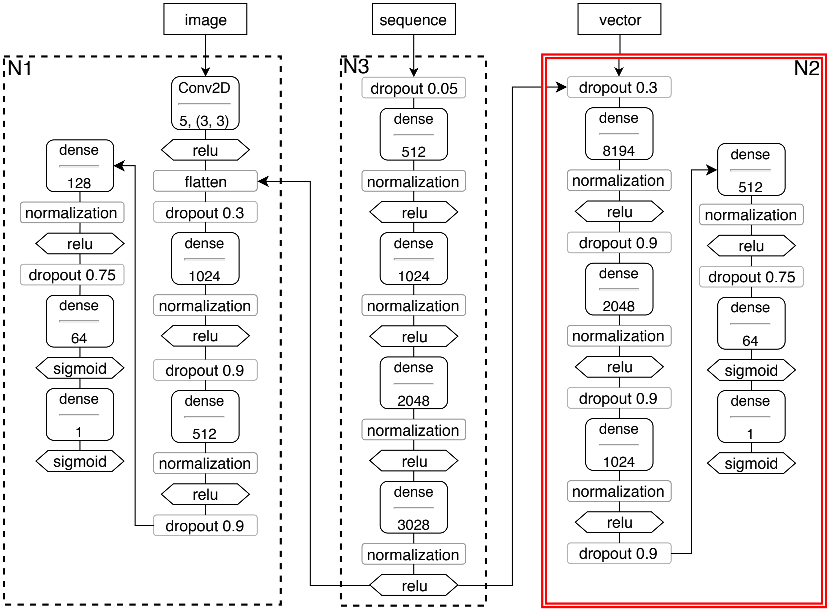
Neural Networks



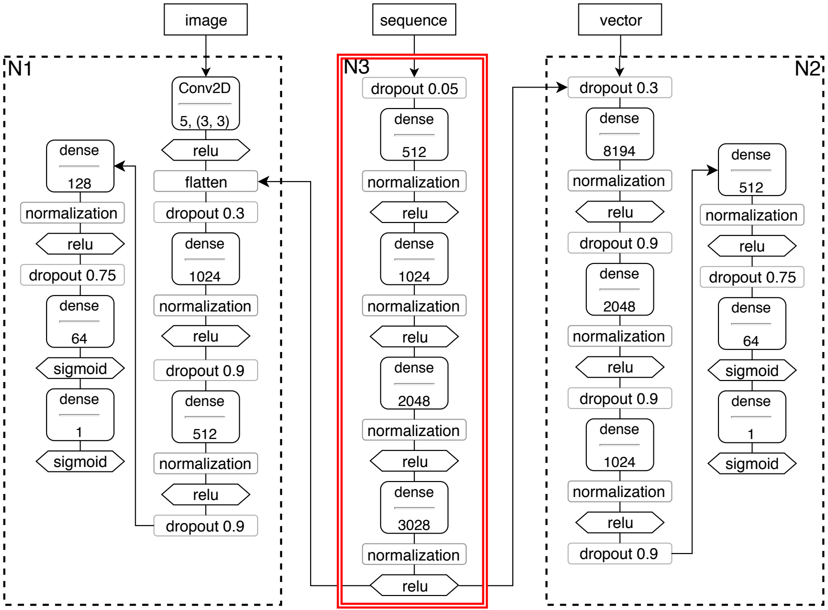
Neural Networks



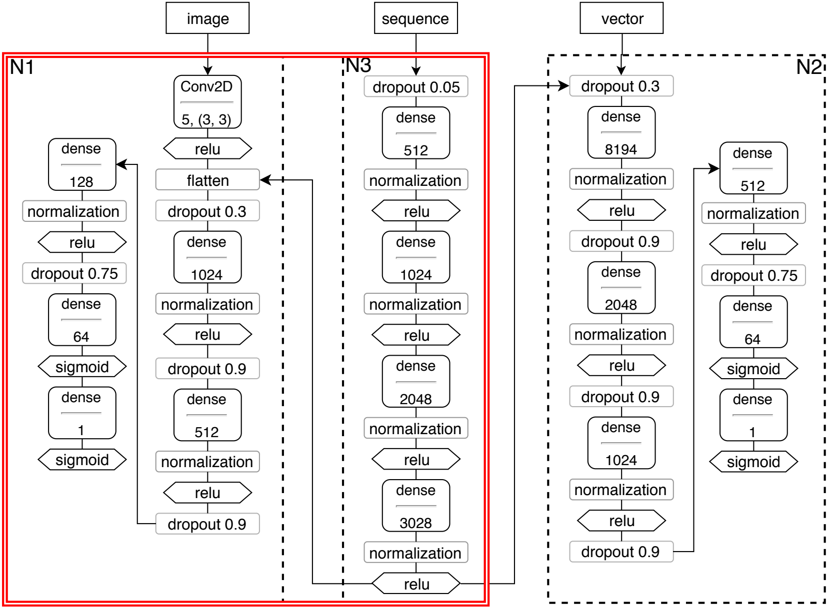
Neural Networks



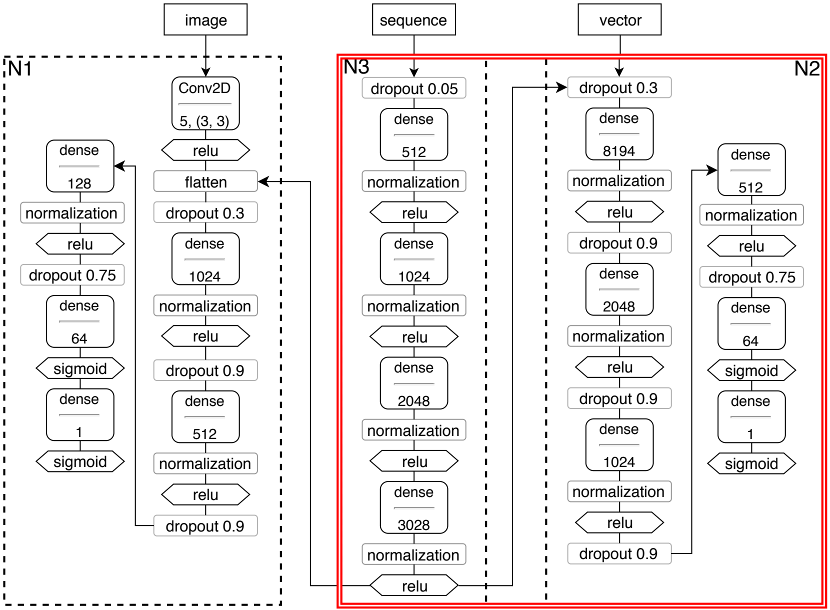
Neural Networks



Neural Networks



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

- We improved the quality of secondary structure analysis by combination of formal grammars and neural networks
  - Parsing result in a form of image can be handled by convolutional layers and it preserves data locality
  - The parsing step can be removed from the final pipeline which allows to run models on the original RNA sequences
- The improved version is applicable for real-world problems

Future Work

- Deep convolutional networks for secondary structure analysis
- Other RNA sequences analysis tasks
  - 16s rRNA classification
  - Chimeric sequences filtration
- Secondary structure prediction by generative networks

Contact Information

- Semyon Grigorev:
    - s.v.grigoriev@spbu.ru
    - semyon.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!