Metagenomic Classification with Deep Learning: Experiments #1

Daniele Bellani

23 • 03 • 2018

Goal

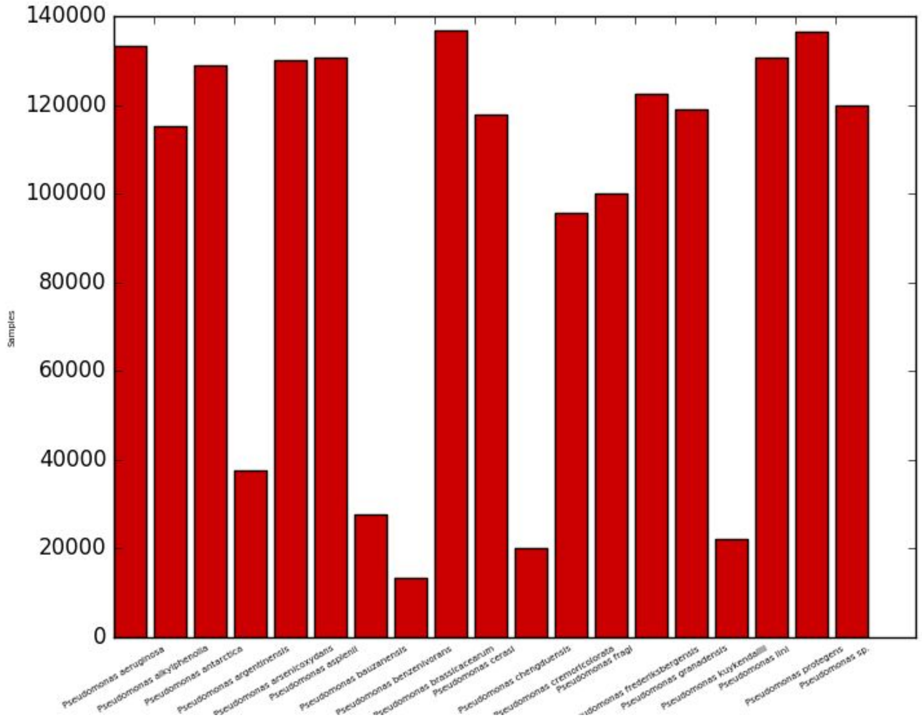
Recognize a specie among others:

one-against-all binary classification

Dataset

- 19 genomes belonging to species of the genus *Pseudomonas*
- Minimum of 90% similarity among them
- Splitting of each genome in 250 base pair **reads**
    - Shifting a cropping window by 50 bp
- Total number of reads per genome = ((GenomeLength-250)/50)+1
- Max number of reads: *Pseudomonas brassicacearum*, 136860
- Min number of reads: *Pseudomonas benzenivorans*, 13459

Dataset reads distribution



Test #1: *Pseudomonas benzenivorans* against all *(least-number-of-reads specie)*

Test 1.1 - Goal

- Study the performances of a chosen network on different training sets
- The training sets differ by composition, i.e. positive/negative examples proportion

Dataset preparation

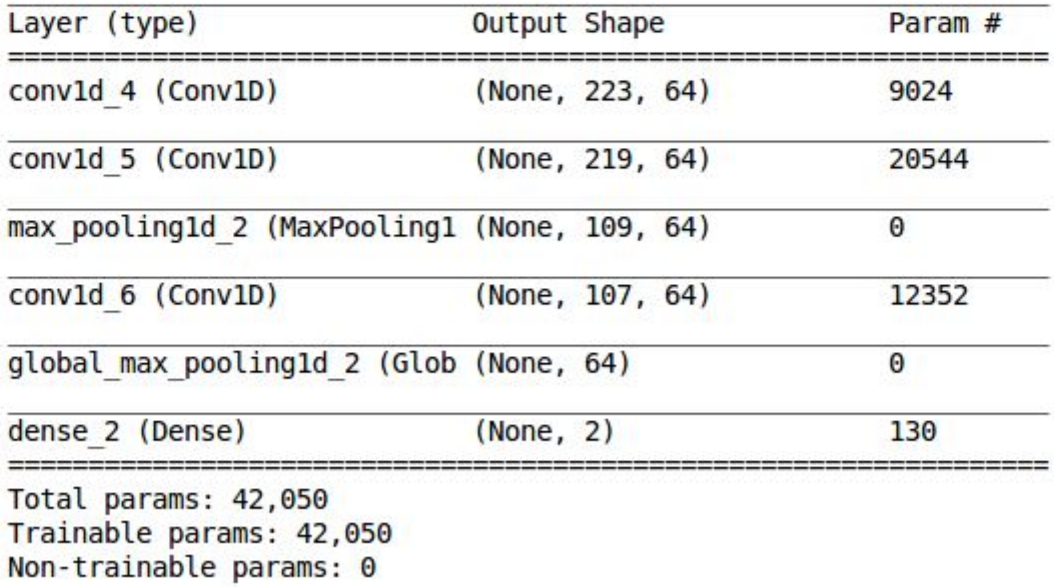
- **Training set**:
  - 80% of positive class reads (Pos = 10767)
  - A number of negative examples proportional to Pos:
    - Neg = Pos*8, Pos*12, Pos*16, Pos*18, Pos*20, Pos*24
- **Test set**:
  - 20% of positive class reads (Pos = 2692)
  - Same amount from all other classes (Neg = Pos*18 = 51130)

Neural network

1. Convolutional(#kernels = 64, kernel_size = 28, activation = ReLU)
2. Convolutional(#kernels = 64, kernel_size = 5, activation = ReLU)
3. MaxPooling(kernel_size = 2)
4. Convolutional(#kernels = 64, kernel_size = 3, activation = ReLU)
5. GlobalMaxPooling(kernel_size = 2)
6. Dense(#neurons = 2, activation = Softmax)

- Learning rate: 0.0005
- Optimizer: Adam
- Epochs: 21
- Batch size: 250

Neural network



```
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_4 (Conv1D)            (None, 223, 64)           9024
_________________________________________________________________
conv1d_5 (Conv1D)            (None, 219, 64)           20544
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 109, 64)           0
_________________________________________________________________
conv1d_6 (Conv1D)            (None, 107, 64)           12352
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 130
=================================================================
Total params: 42,050
Trainable params: 42,050
Non-trainable params: 0
```

Training set Neg = Pos*8

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2112	580
	Neg	1039	47399

- Precision: 0.670
- Recall: 0.784
- F1: 0.722

Training set Neg = Pos*12

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2392	300
	Neg	1042	47396

- Precision: 0.696
- Recall: 0.888
- F1: 0.780

Training set Neg = Pos*16

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	1939	753
	Neg	632	47806

- Precision: 0.762
- Recall: 0.845
- F1: 0.801

Training set Neg = Pos*18

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	1712	980
	Neg	182	48256

- Precision: 0.903
- Recall: 0.635
- F1: 0.746

Training set Neg = Pos*20

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2140	552
	Neg	557	47881

- Precision: 0.793
- Recall: 0.794
- F1: 0.794

Training set Neg = Pos*24

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2373	319
	Neg	714	47724

- Precision: 0.768
- Recall: 0.881
- F1: 0.821

Test 1.2 - Goal

- Changing some of the parameters of the network, in order to improve performances.
- Training set chosen as the one which brought the worst recall (Neg = 18*Pos).

Neural network

1. Convolutional(#kernels = 64, kernel_size = 28, activation = ReLU)
2. Convolutional(#kernels = 96, kernel_size = 5, activation = ReLU)
3. MaxPooling(kernel_size = 2)
4. Convolutional(#kernels = 96, kernel_size = 3, activation = ReLU)
5. GlobalMaxPooling(kernel_size = 2)
6. Dense(#neurons = 2, activation = Softmax)

- Learning rate: 0.0005
- Optimizer: Adam
- Epochs: 21
- Batch size: 250

Kernels: 64-96-96

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2466	226
	Neg	398	48040

- Precision: 0.861
- Recall: 0.916
- F1: 0.887

Neural network

1. Convolutional(#kernels = 96, kernel_size = 28, activation = ReLU)
2. Convolutional(#kernels = 96, kernel_size = 5, activation = ReLU)
3. MaxPooling(kernel_size = 2)
4. Convolutional(#kernels = 96, kernel_size = 3, activation = ReLU)
5. GlobalMaxPooling(kernel_size = 2)
6. Dense(#neurons = 2, activation = Softmax)

- Learning rate: 0.0005
- Optimizer: Adam
- Epochs: 21
- Batch size: 250

Kernels: 96-96-96

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2442	250
	Neg	269	48169

- Precision: 0.900
- Recall: 0.907
- F1: 0.903

Neural network

1. Convolutional(#kernels = 96, kernel_size = 21, activation = ReLU)
2. Convolutional(#kernels = 96, kernel_size = 5, activation = ReLU)
3. MaxPooling(kernel_size = 2)
4. Convolutional(#kernels = 96, kernel_size = 3, activation = ReLU)
5. GlobalMaxPooling(kernel_size = 2)
6. Dense(#neurons = 2, activation = Softmax)

- Learning rate: 0.0005
- Optimizer: Adam
- Epochs: 21
- Batch size: 250

Kernels: 96-96-96, Conv1 kernel_size 21

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2511	181
	Neg	290	48148

- Precision: 0.896
- Recall: 0.932
- F1: 0.914

Neural network

1. Convolutional(#kernels = 96, kernel_size = 35, activation = ReLU)
2. Convolutional(#kernels = 96, kernel_size = 5, activation = ReLU)
3. MaxPooling(kernel_size = 2)
4. Convolutional(#kernels = 96, kernel_size = 3, activation = ReLU)
5. GlobalMaxPooling(kernel_size = 2)
6. Dense(#neurons = 2, activation = Softmax)

- Learning rate: 0.0005
- Optimizer: Adam
- Epochs: 21
- Batch size: 250

Kernels: 96-96-96, Conv1 kernel_size 35

- Confusion matrix:

		Predicted	
		Pos	Neg
Real	Pos	2421	271
	Neg	179	48259

- Precision: 0.931
- Recall: 0.899
- F1: 0.914