Deep learning models for bacteria taxonomic classification of metagenomic data

Fiannaca et al. [1]

Suresh Kumar Choudhary / 04.06.2021

Course: Computational Meta-Omics
Course Instructor: Professor Thilo Muth

Table of Contents

Goal of the study

- Taxonomic classification of metagenomic data using Deep Learning Approaches
- Comparison of Deep Learning Models with Reference Model

Related Work

Paper	Study Focus
Hayssam S et al. [8]	OTU-clustering, binning, taxonomic profling and assignment, comparative metagenomics and gene prediction
Wang Q et al. [9]	Bacterial taxonomy classification (Naive Bayesian classifier)
Kultima JR et al. [10]	MOCAT: combination of genome assembly and gene prediction
Min S et al. [11]	Deep learning in bioinformatics
Lo Bosco G et al. [12]	Deep Learning Architectures for DNA Sequence Classification
Gangi et al. [13]	Prediction of nucleosome positioning from sequences data (CNN, LSTM)

Dataset

- Created two artificial datasets
- In detail, downloaded from the RDP database (release 11, update 5 dated September 30, 2016)
- Filtered with the parameters: Strain: both Type and Non-Type; Source: Isolates; Size: greater than or equal to 1200; Quality: Good
- Obtained 57788 16S gene sequences.

Dataset

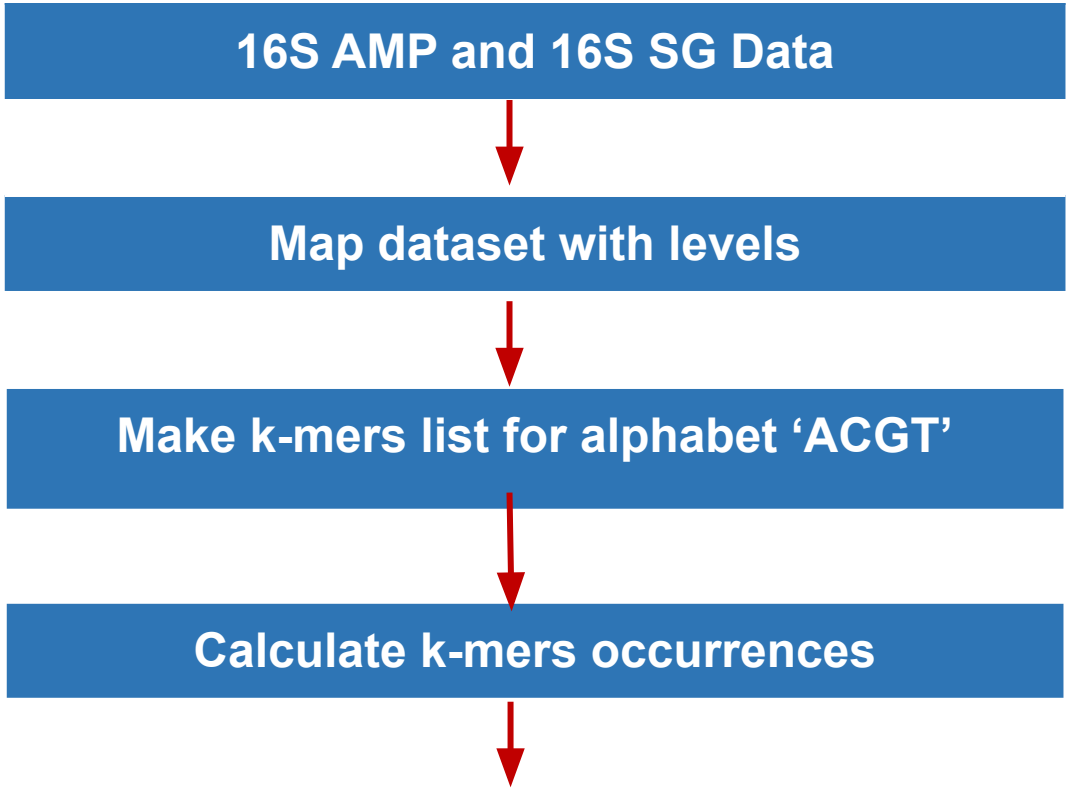
- For each short-read, the taxa is known.
- Only generated short-reads belonging to 16S using the tool REAGO [2]
- REAGO can distinguish reads belonging (or not) to 16S with accuracy near to 99%
- Grinder tool [14] used for simulating shotgun and amplicon metagenomic datasets
- Metagenomic data levels: Class, Order, Family, Genus

Dataset

- SG dataset: 28224 short-reads
- AMP(v3-v4 hypervariable region): 28000 short reads
- MICCA [4] primer trimming tool is used to remove the primer sequences
- Number of different categories for taxa:

Proteobacteria phylum			
# class	# order	# family	# genus
3	20	39	100

Data Preparation



Data Preparation

```
In [3]: df.head()
```

Out[3]:

	seq_id	AAAAA	AAAAC	AAAAG	AAAAT	AAACA	AAACC	AAACG	AAACT	AAAGA	...	TTTCT	TTTGA	TTTGC	TTTGG	TTTGT	TTTTA	TTTTC
0	S003747738	0	0	0	0	0	1	0	0	1	...	0	0	0	0	0	0	0
1	S003747738	1	2	0	1	1	1	0	1	0	...	0	0	0	0	0	0	0
2	S003747738	0	0	0	1	0	1	0	0	1	...	0	0	0	0	0	0	0
3	S003747738	0	0	0	1	0	1	0	1	0	...	0	0	0	0	0	0	0
4	S003747738	0	0	1	0	0	1	0	0	1	...	0	0	0	0	0	0	1

5 rows × 1026 columns

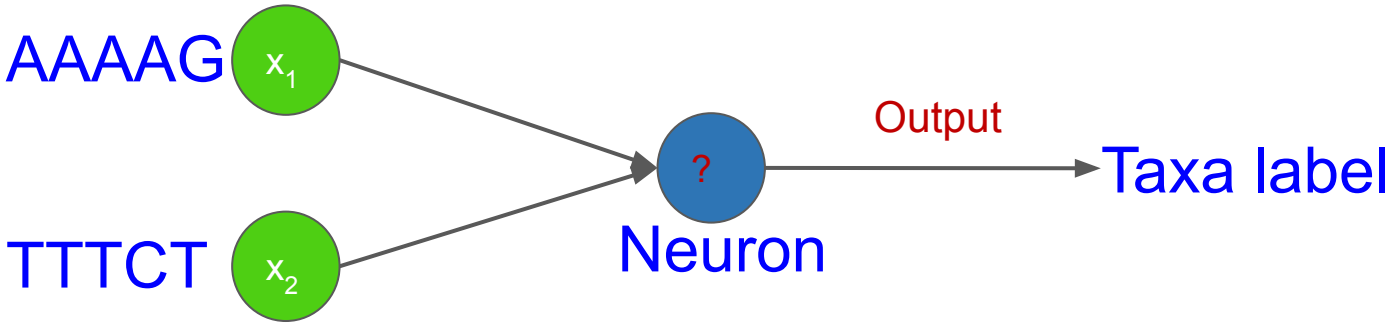
```
In [4]: df['Taxa_label'].value_counts()
```

Out[4]:

```
Nicoletella          344
Histophilus          340
Kingella             340
Phyllobacterium      337
Pseudophaeobacter    336
```

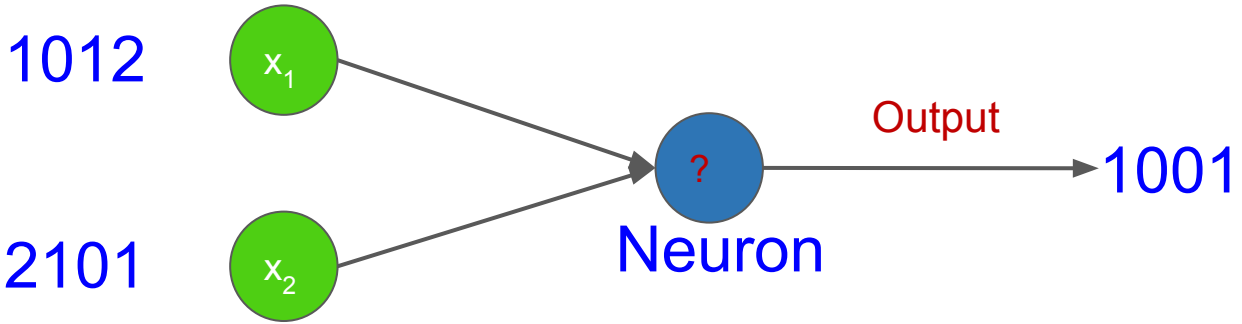
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



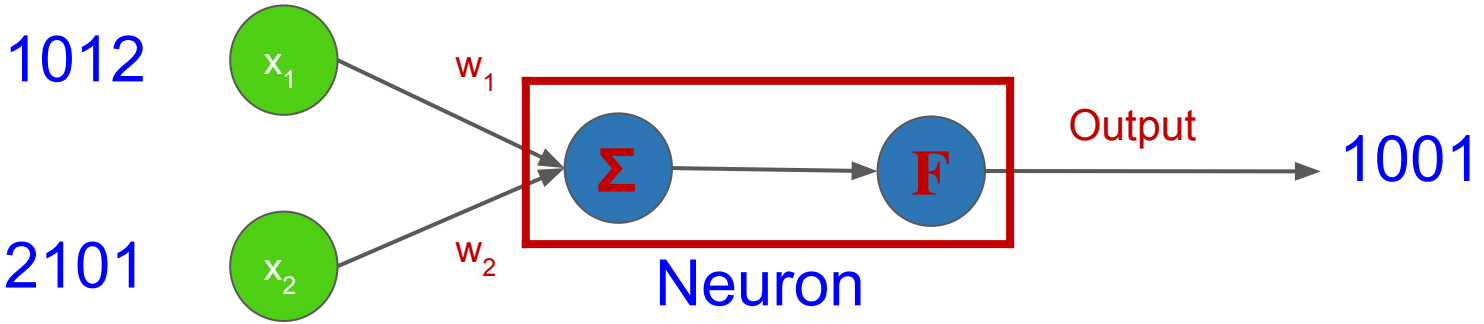
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



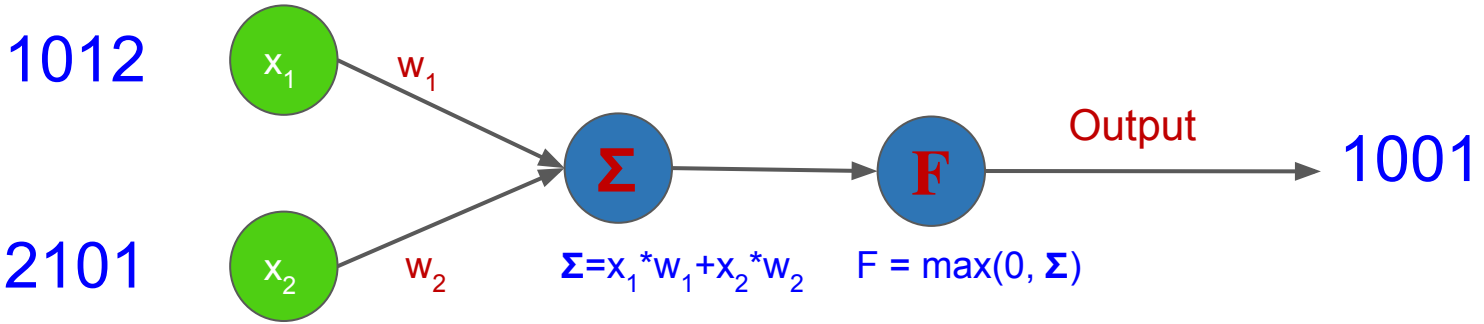
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



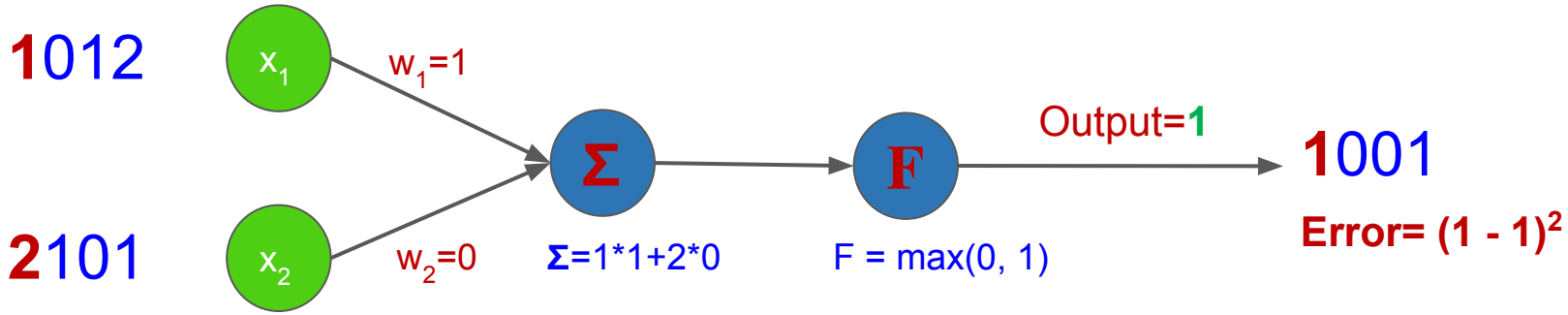
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



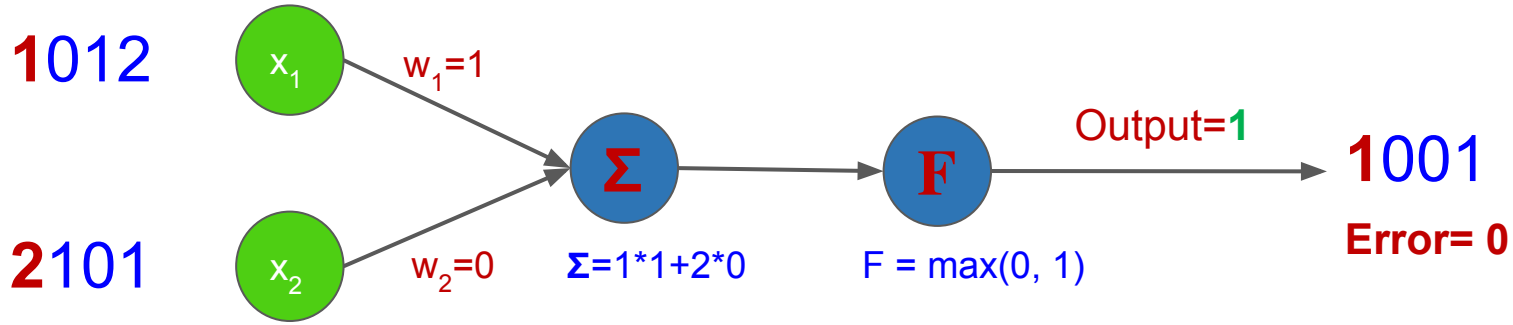
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



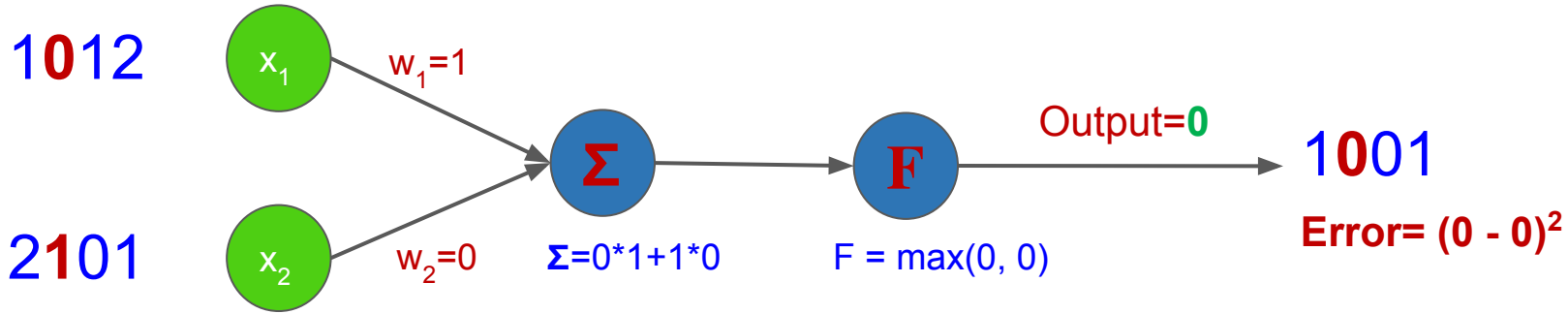
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



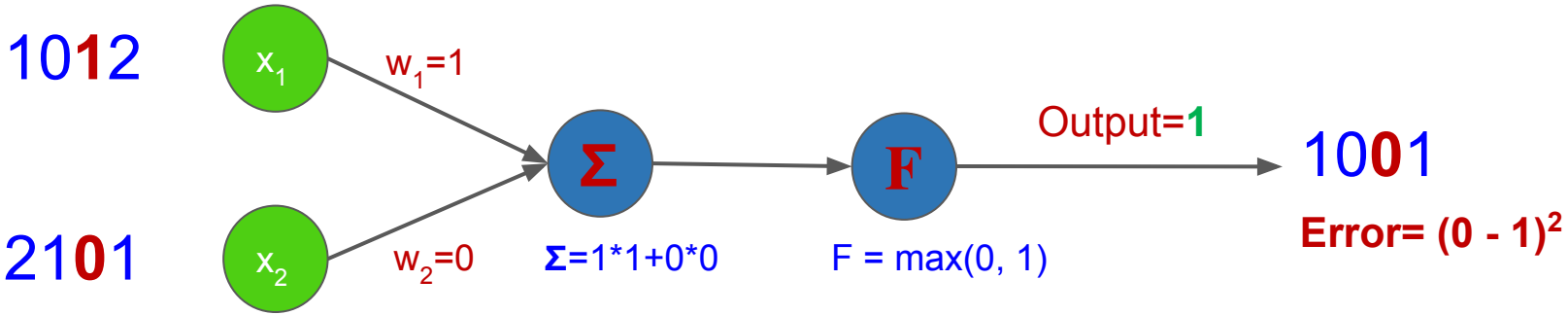
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



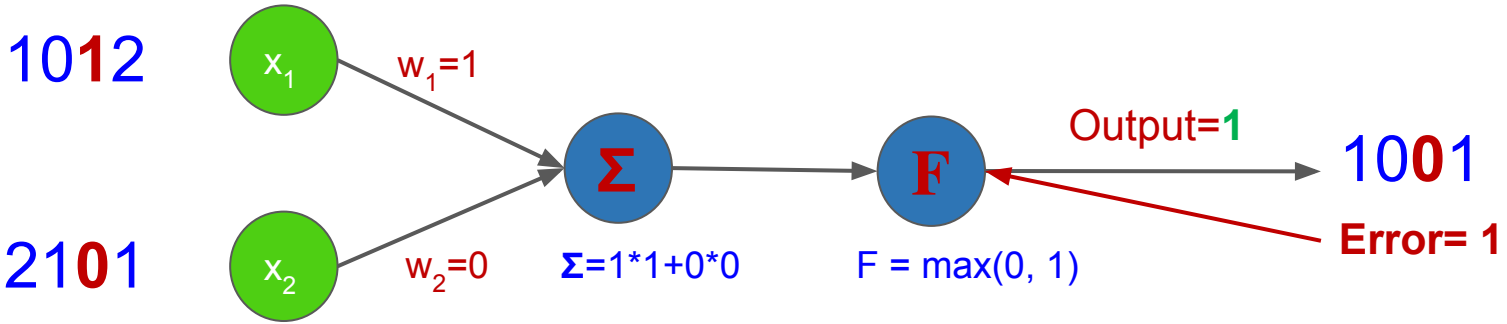
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



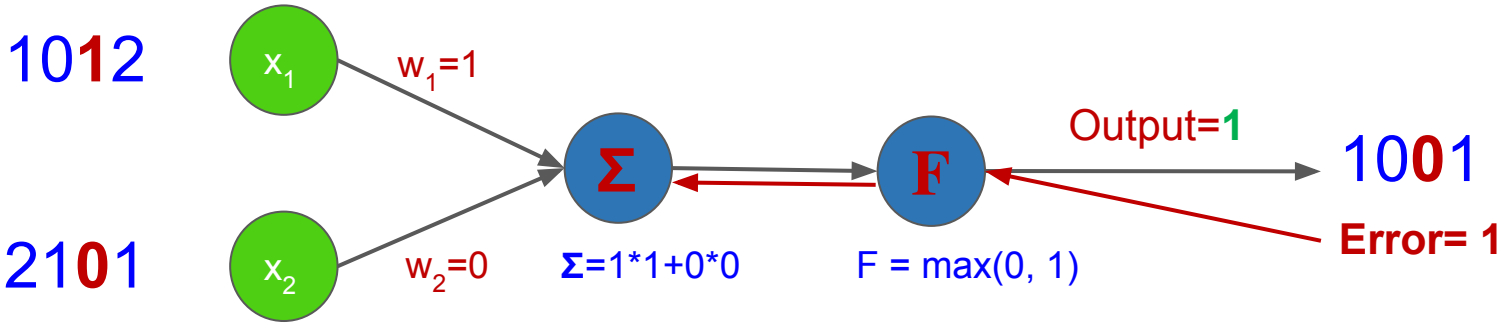
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



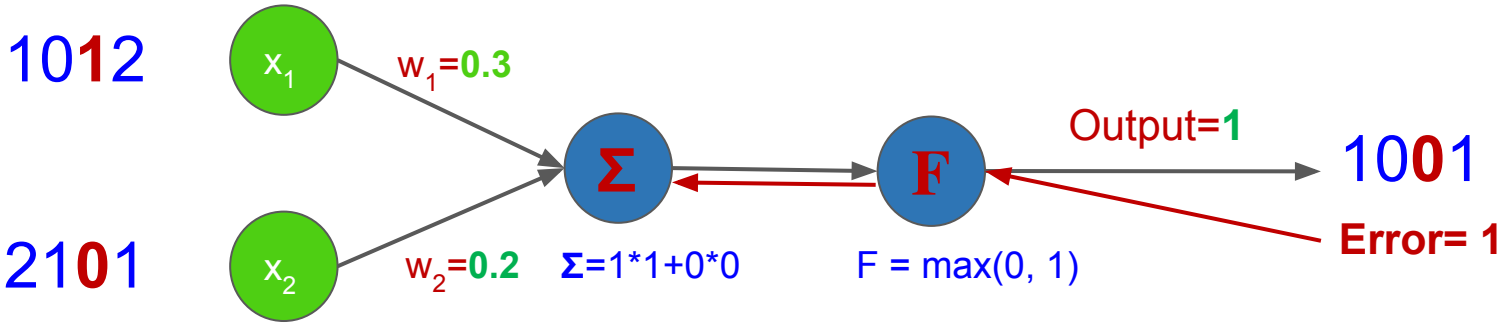
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



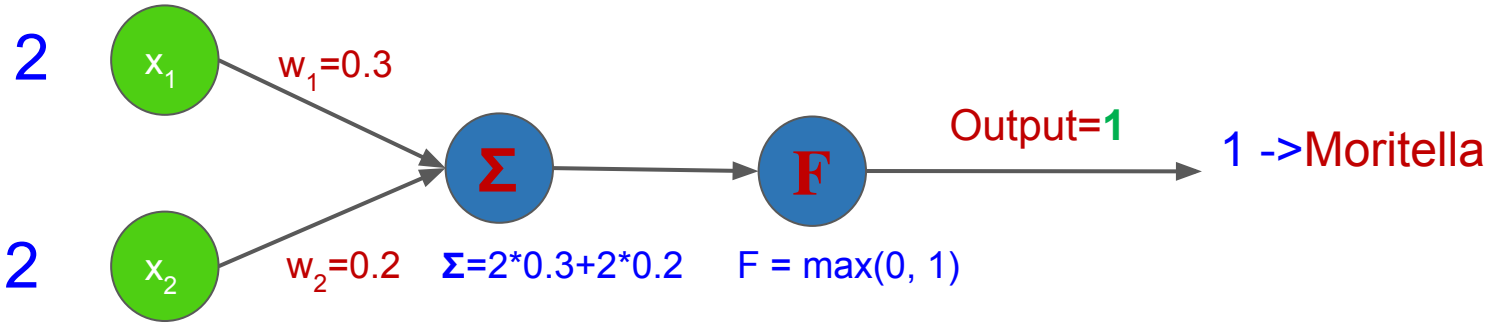
Single Neuron

AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????

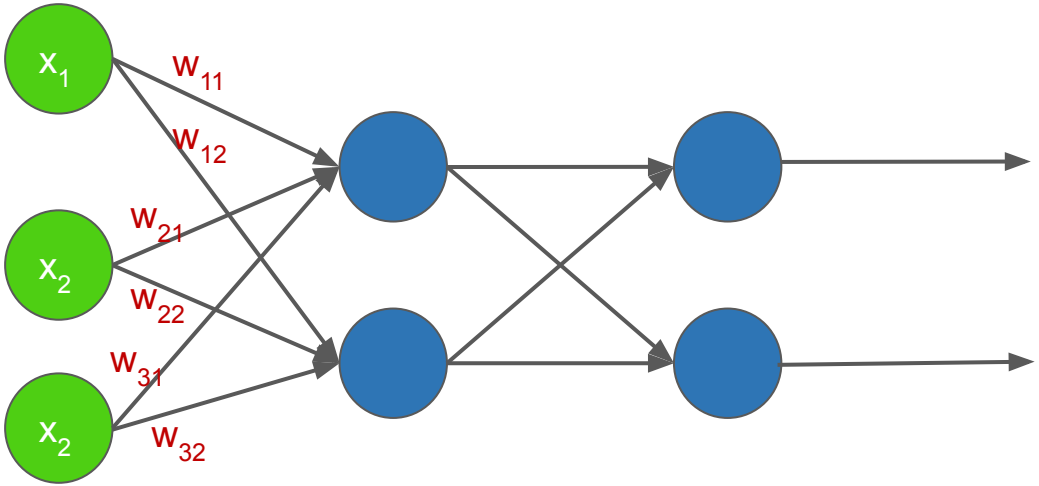


Single Neuron

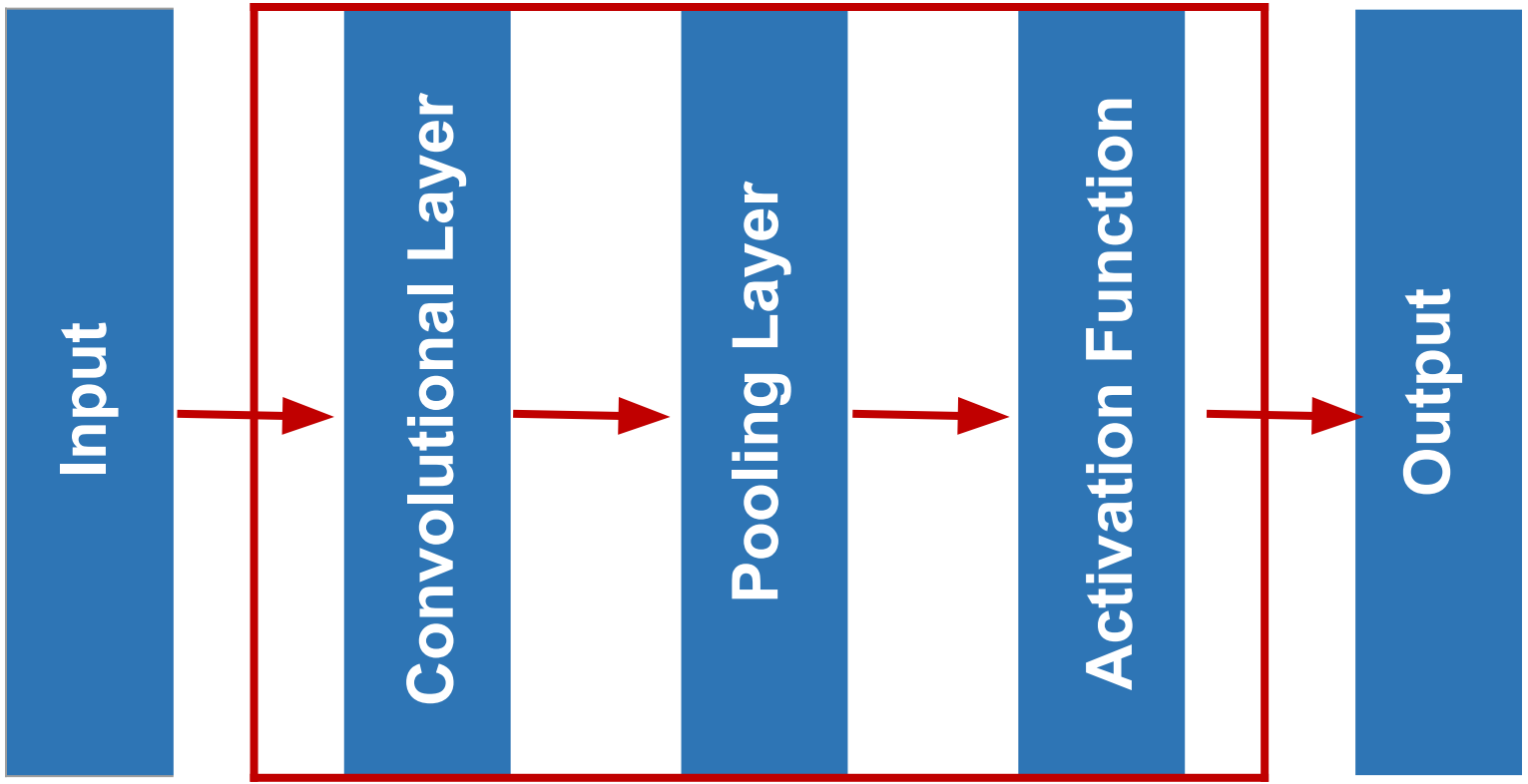
AAAAG	TTTCT	Taxa label
1	2	Moritella -> 1
0	1	Kingella -> 0
1	0	Kingella -> 0
2	1	Moritella -> 1
2	2	??????????



Multi Neuron with Multilayer



Convolutional Neural Network(CNN)

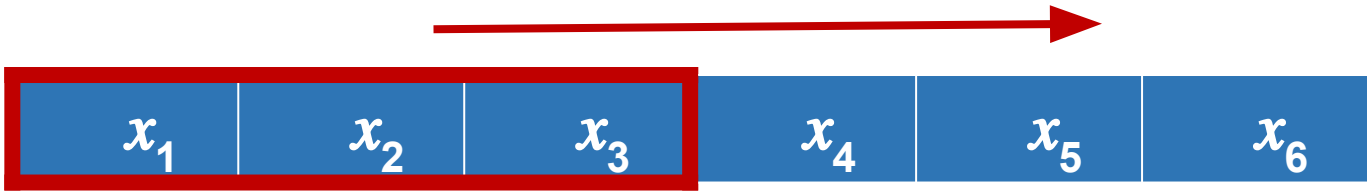


Convolutional Neural Network

- 1 D Convolution
- It is a linear operation
- Mathematical Definition:

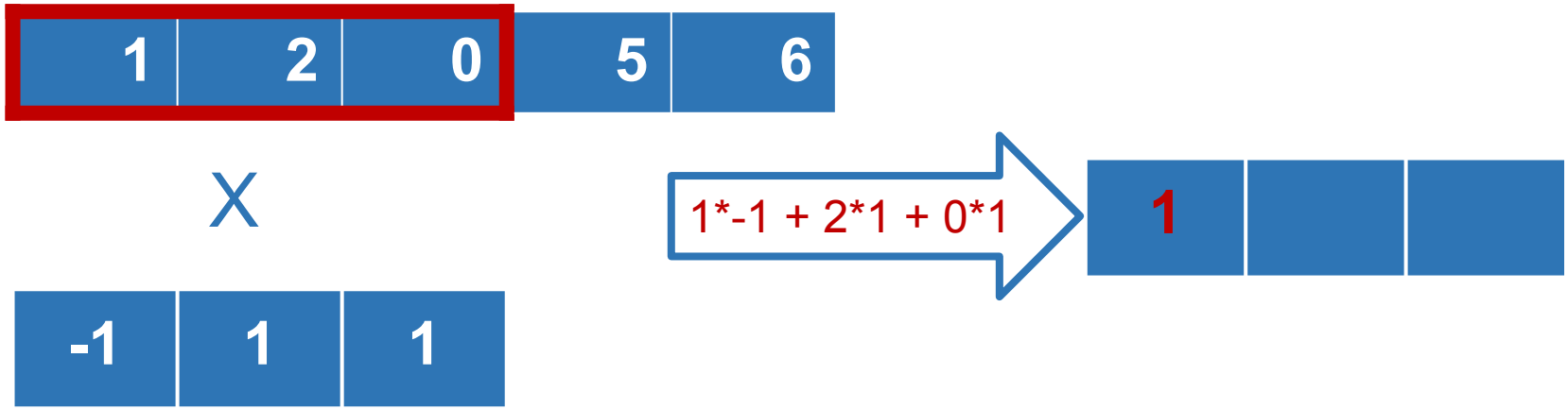
For $x \in \mathbb{R}^n$, $K \in \mathbb{R}^m$ and $m < n$

$$(x * K)_i = \sum_{j=1}^{m} x_{i+j-1} K_j$$



1 D Convolution

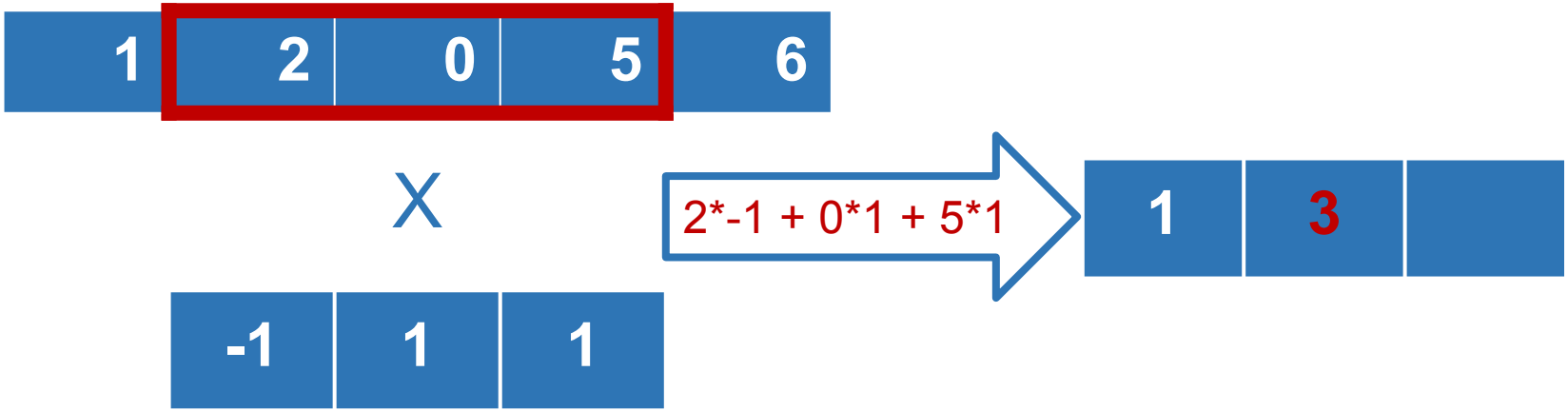
K-mers occurrences



Kernel Size 3 x 1

1 D Convolution

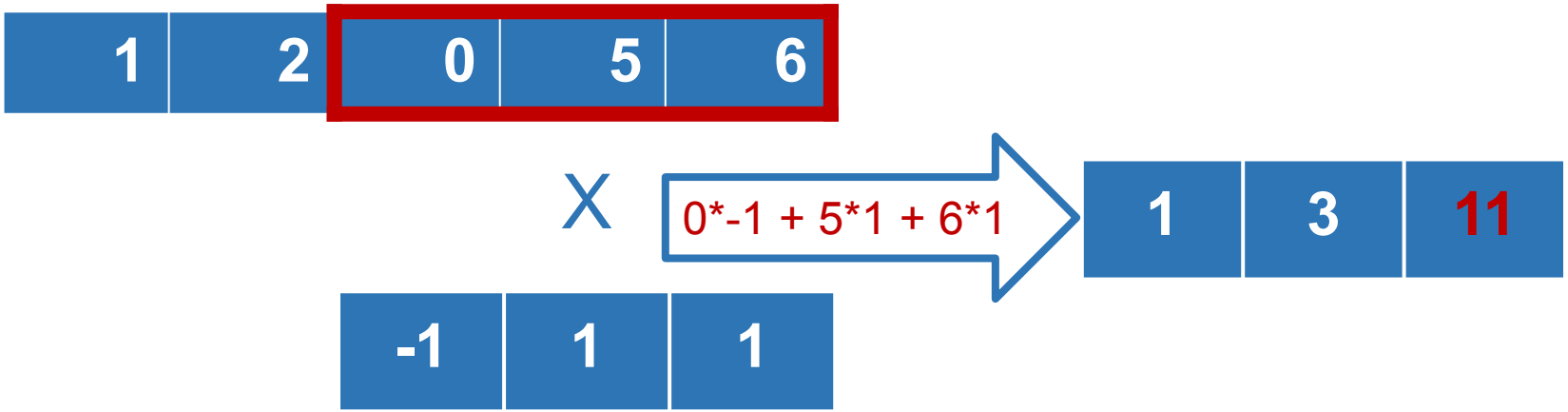
K-mers occurrences



Kernel Size 3 x 1

1 D Convolution

K-mers occurrences



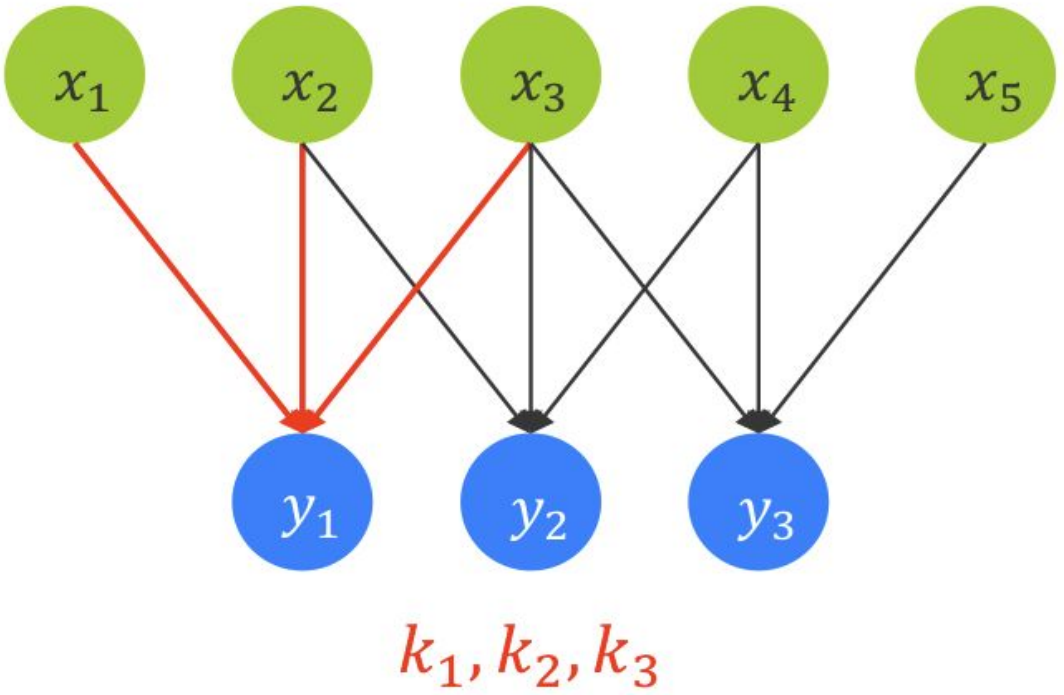
Kernel Size 3 x 1

Neural Network Representation

- $y_i = x * w$

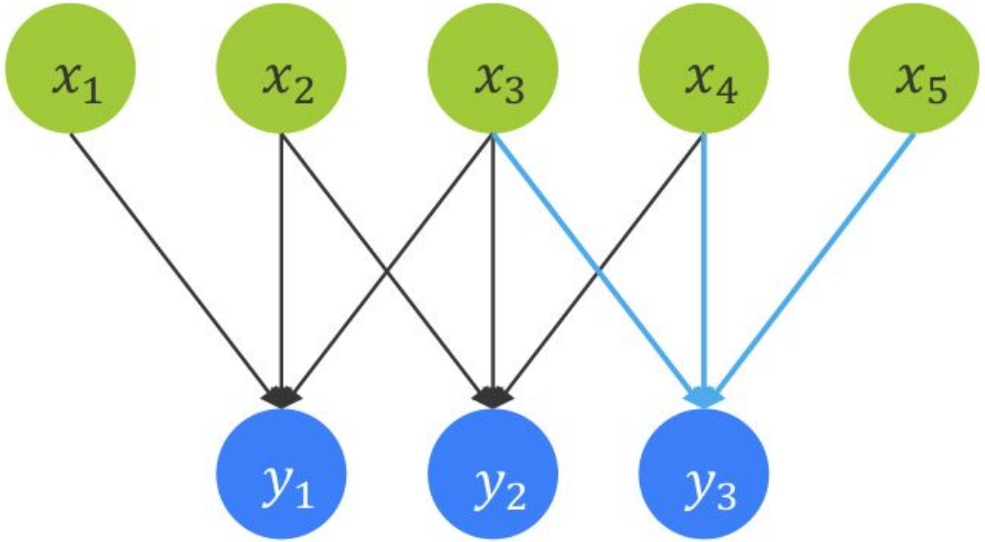
  Or

- $y_i = x * k$



Neural Network Representation

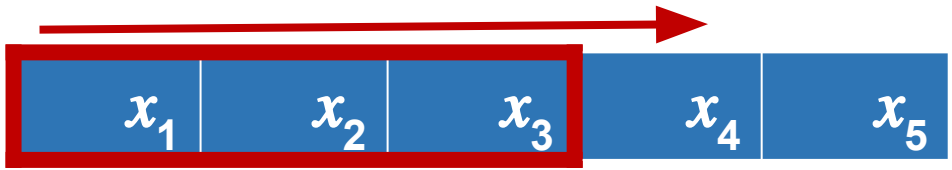
- $y_i = x * k$



Pooling

- Pooling is similar to convolution layer:
  - Slides window over image and
  - aggregates neighbouring pixels
- But: Aggregation without kernel.
- Learnable parameters: 0
- Common aggregations:
  - Maximum (max pooling)
  - Average (average pooling)

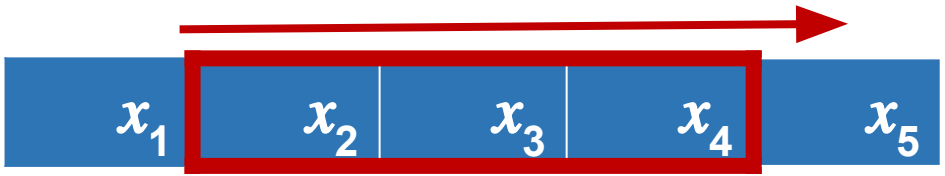
1 D Max Pooling



$$= max(x_i)$$

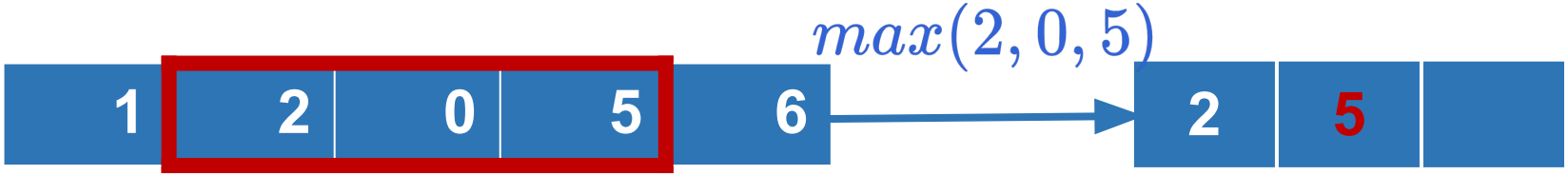
K-mers occurrences

1 D Max Pooling

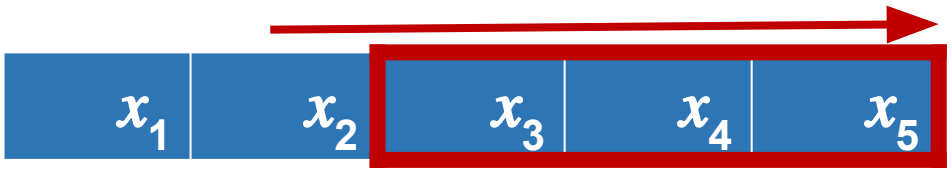


$$= max(x_i)$$

K-mers occurrences



1 D Max Pooling



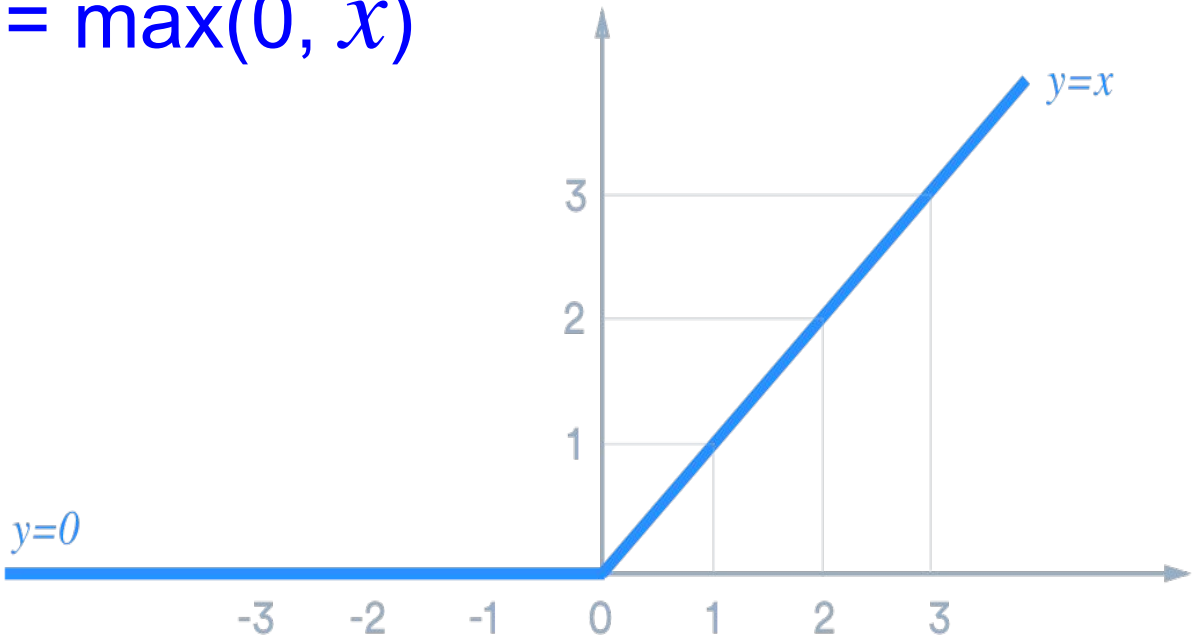
$$= max(x_i)$$

K-mers occurrences

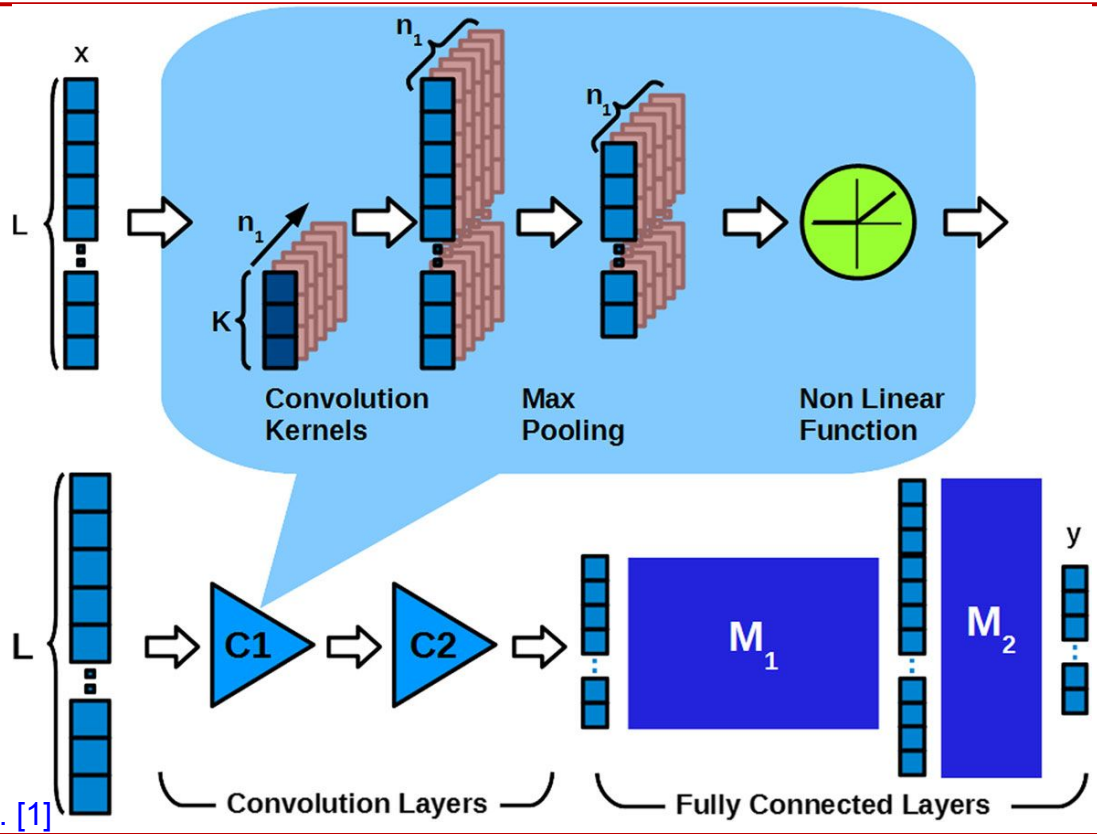
$$max(0, 5, 6)$$

Activation Function

- ReLU (Rectified Linear Units):
  - $f(x) = \max(0, x)$



Convolutional Neural Network Architecture



Source: Fiannaca et al. [1]

CNN Architecture (continue)

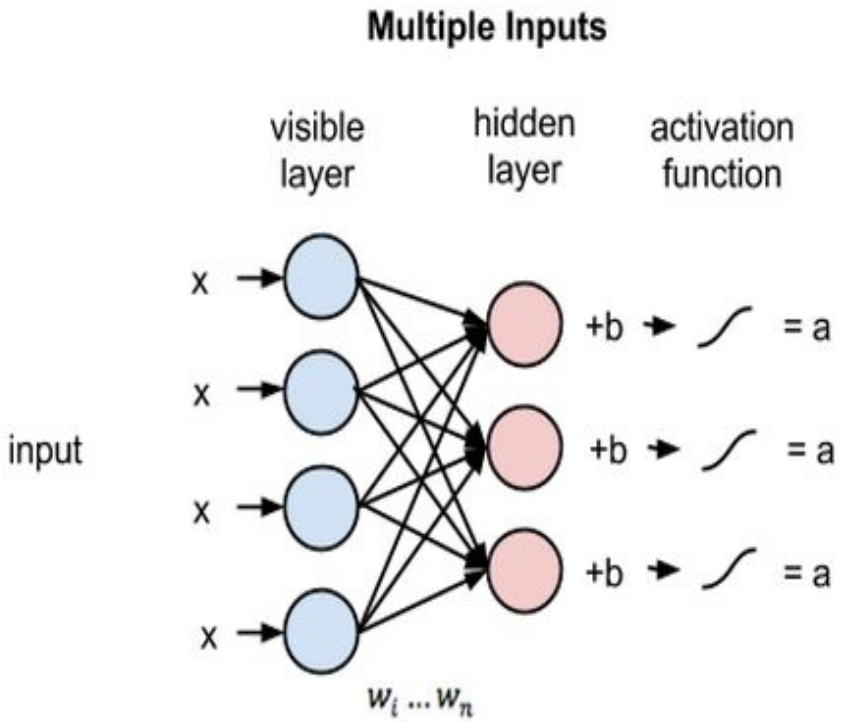
CNN Parameters						
Layer1			Layer2			Dense Layer
Kernel Size(k)	No of Kernels(n1)	Pooling size	Kernel Size(k)	No of Kernels(n2)	Pooling size	Hidden units
5	5	2	5	10	2	500

Deep Belief Network

- A probabilistic generative model
- Used to extract a hierarchical representation of input data
- Composed of Restricted Boltzmann Machines (RBM)
- Stack of at least two RBM layers
- DBN has two phases:-
    - Pre-train Phase
    - Fine-tune Phase

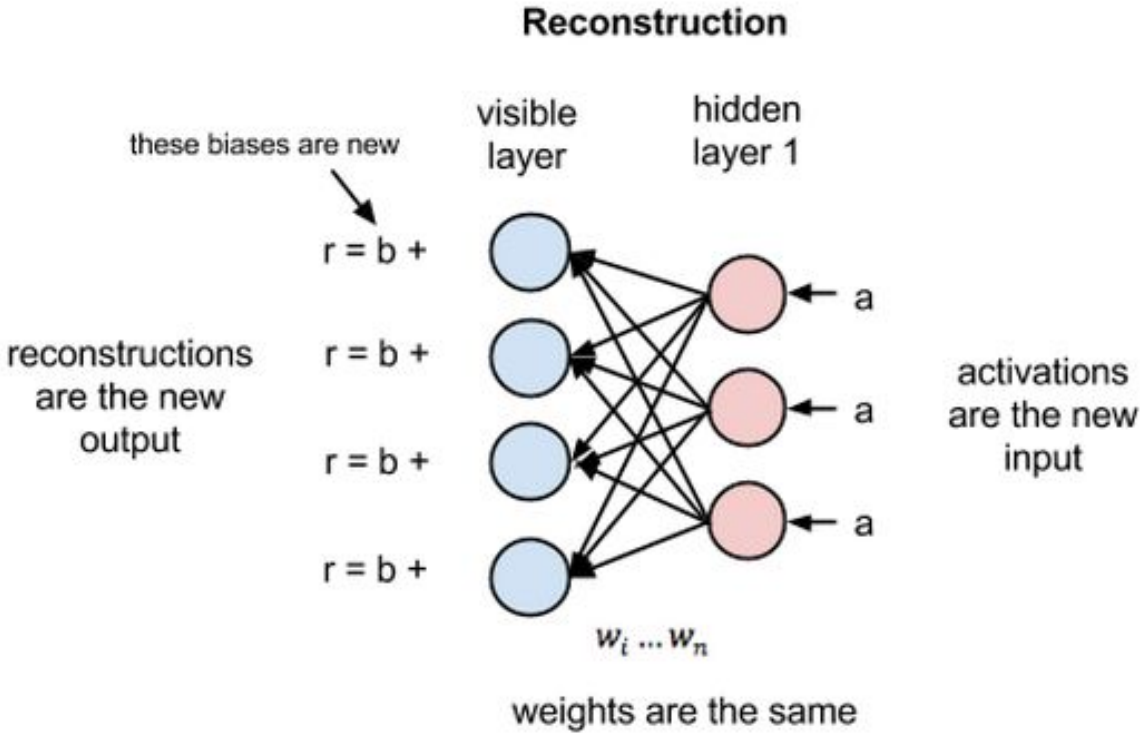
Restricted Boltzmann Machine(RBM)

- Bipartite Graph
- Learns probability distribution over its set of inputs.
- Visible units are our inputs



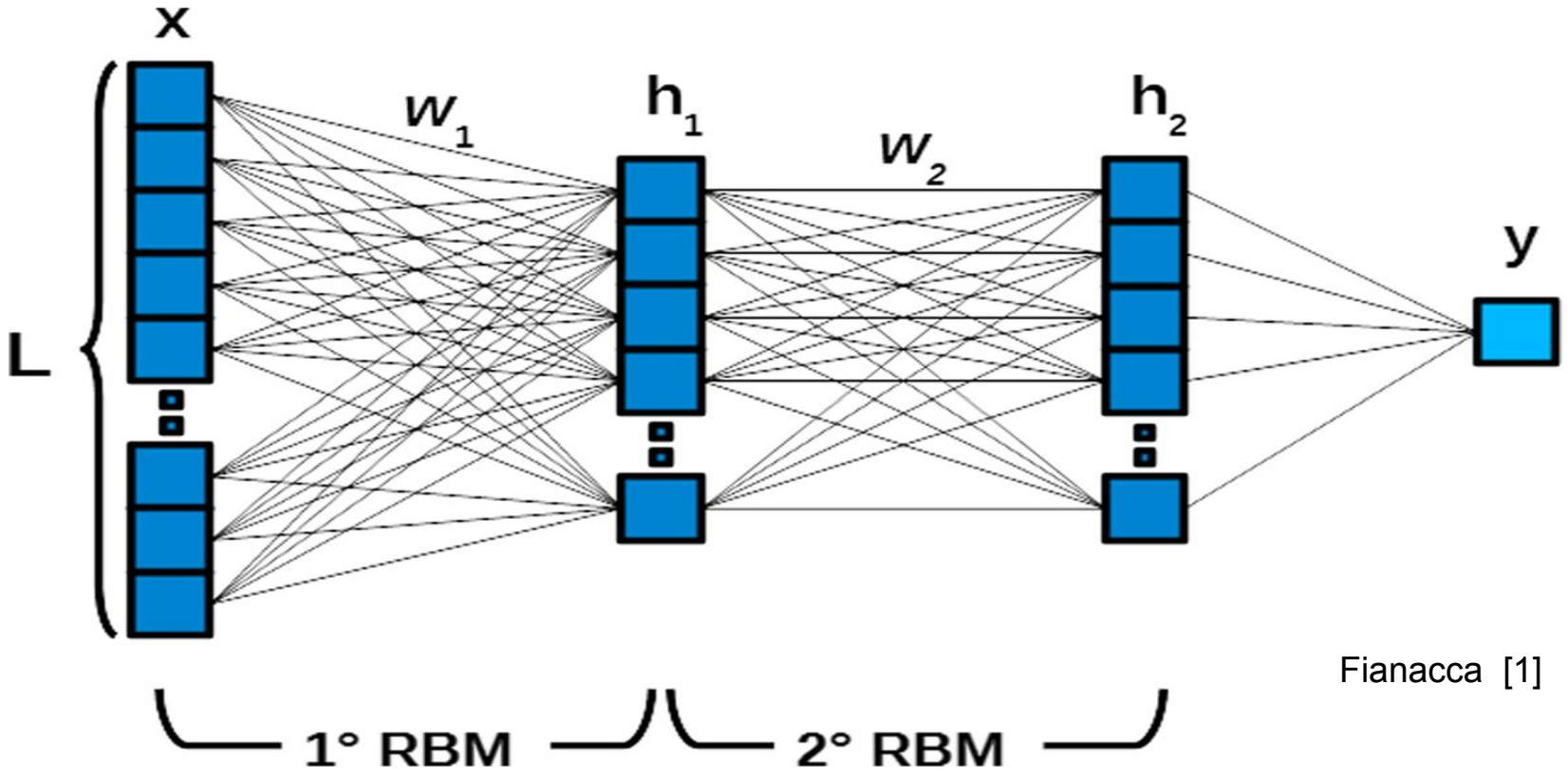
Source: Pathmind [7]

RBM



Source: Pathmind [7]

Deep Belief Network



Fianacca [1]

Deep Belief Network

Fianacca [1]

K-mer size (k)	RBM layer 1 hidden units	RBM layer 2 hidden units
3	32	32
4	128	128
5	256	256
6	256	256
7	256	256

RDP Classifier

- Feature Space: k-mers substring of length 8
- Given unknown query sequence, s, belongs to a genus g_i is modeled according to the Bayes rule:

$$P(g_i|s) = P(s|g_i) * P(g_i)/P(s)$$

- Where $P(s|g_i)$ is the joint probability of observing a sequence s from a genus g_i ,
- $P(g_i)$ is the prior probability
- P(s) is the overall probability

RDP Classifier

- The prior estimate of the likelihood of observing a single k-mer r_i in an rRNA sequence:

$$P_i = (n(r_i) + 0.5)/(N - 1)$$

- Where $n(r_i)$ is the number of sequences in the corpus containing k-mer r_i
- N is the total number of sequences

RDP Classifier

- The joint probability Calculated as:

$$P(s|g_i) = \prod_{r_j \epsilon V_i} \frac{m(r_j)+P_j}{M_i+1}$$

- Where M_i is the total number of sequences in the training set T_i of genus g_i , $m(r_j)$ the number of sequences in T_i containing k-mer r_j and V_i is the subset of k-mers that are substrings of at least one sequence in T_i

RDP Classifier

- Assuming all genera are equally probable (equal priors)
- The constant terms $P(g_i)$ and $P(s)$ can be ignored
- The rule to assign a sequence s to a genus g_i is:

$$i = argmax_z P(s|g_z)$$

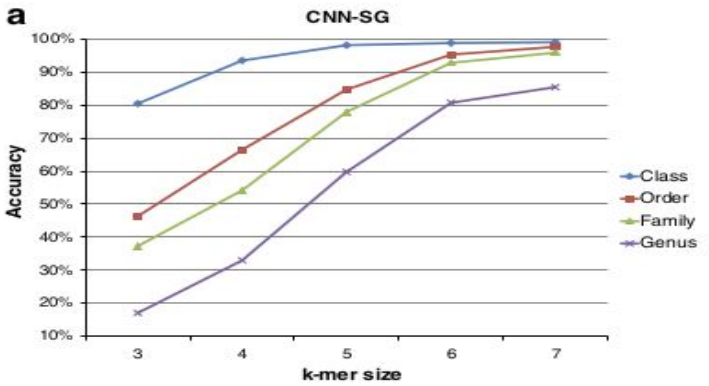
Comparison of Models

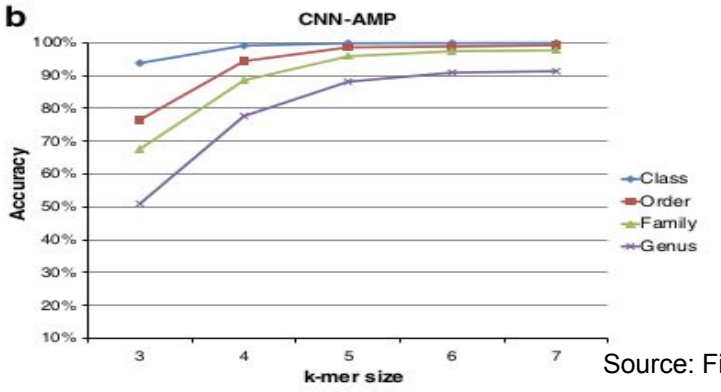
CNN	DBN	RDP
Discriminative Model	Generative Model	Generative Model
Weight shares	No Weight Share	No weight share concept
Supervised Algorithm	Supervised and Unsupervised both	Supervised algorithm
High computation time	Less than the CNN	Lowest computation time
Uses less parameters compare to DBN	Highest	No parameters
Suitable to capture local spatial information	Suitable for non spatial data	Suitable for non spatial data

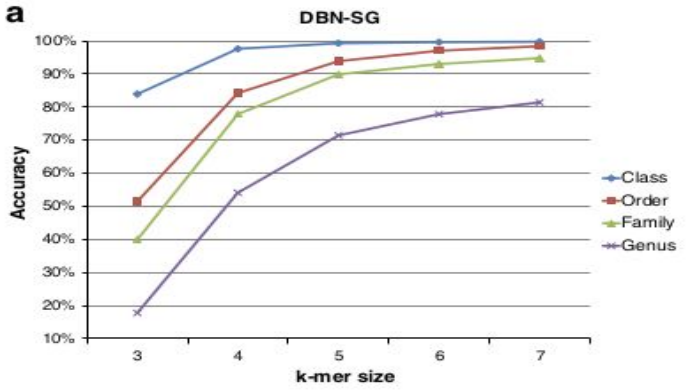
Results

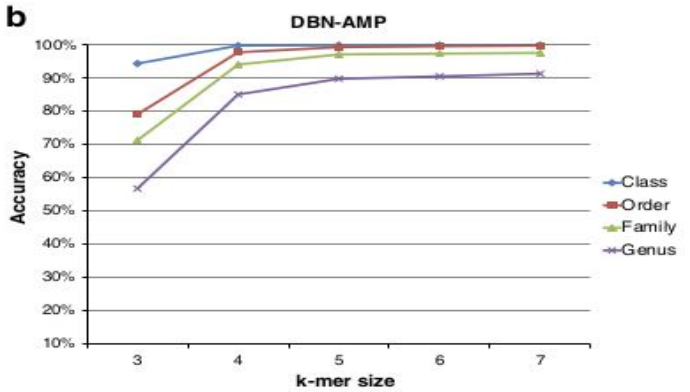
- Tenfold cross-validation scheme is used
- Experimented with k-mers of length from 3 to 7
- Accuracy, Precision, Recall and F1-Score metric has been used

Results



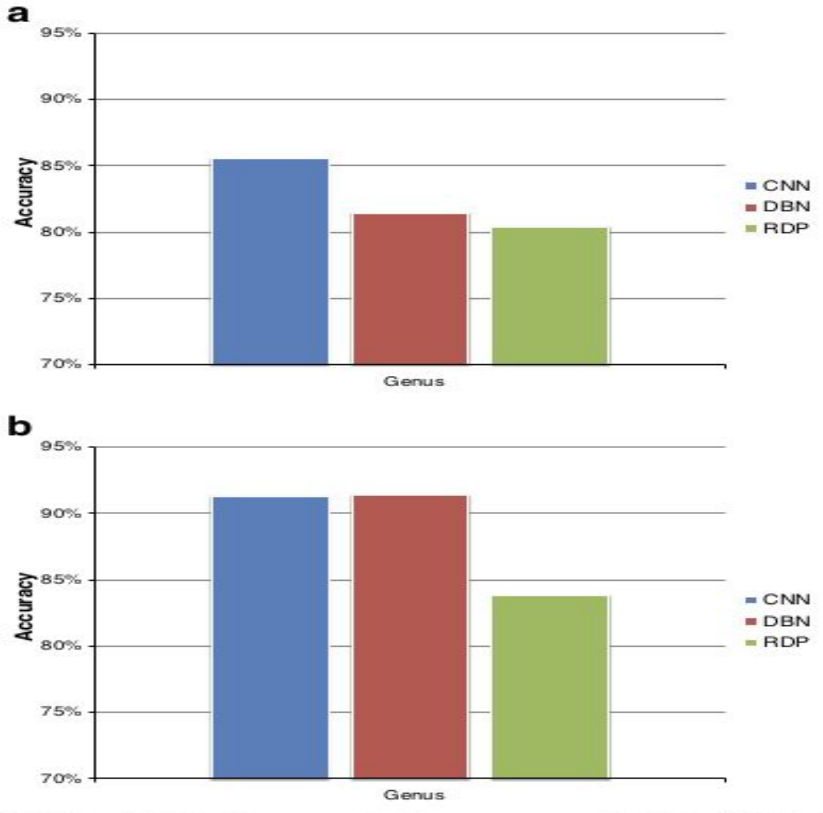






Source: Fianacca [1]

Results



Source: Fianacca [1]

Conclusion & Discussion

- Increasing the k size, accuracy also increases (for higher K- values it is not tested)
- With k=7, got better results
- CNN and DBN performed better compared to RDP classifier
- DBN performed well with k<7 as well

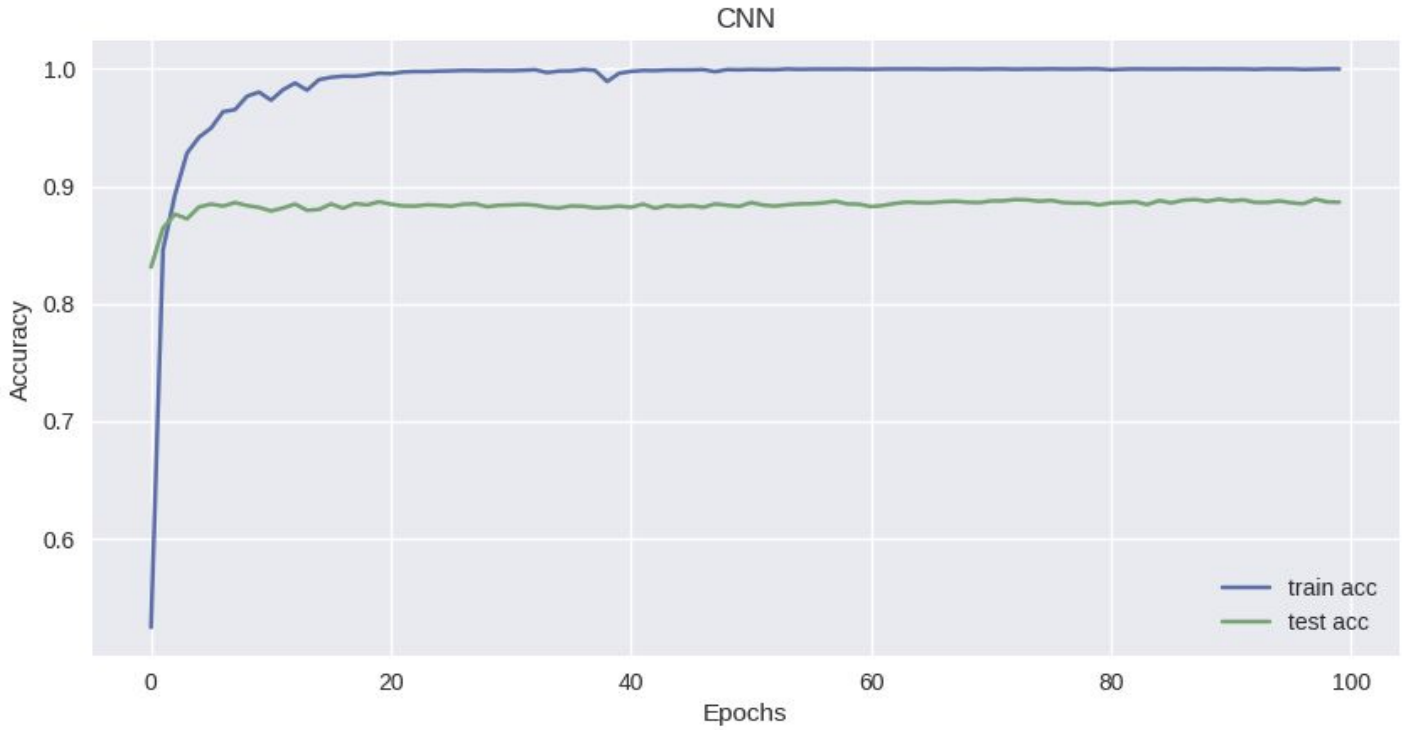
Implementation

In [29]:

```
start = time.time()
model = CNNModel()
if cuda_available:
    model = model.cuda()
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(model.parameters(), lr=1e-2, momentum=0.9, weight_decay=5e-4)
scheduler = torch.optim.lr_scheduler.MultiStepLR(optimizer, milestones=[150, 200], gamma=0.1)
#optimizer = optim.Adam(model.parameters(), lr=1e-3)
losses,test_losses, train_acc,test_acc, best_model = train(model, optimizer,scheduler, (train_loader, test_loader),
end = time.time()
print(f'\nTraining took {end-start}s!')
```

```
Epoch: 99/100  Loss: 0.004548 Acc: 100.00% Test loss: 0.454715 Test acc: 88.64%
Training took 46.607646465301514s!
```

Implementation



Implementation

- My Implementation of this project can be found at my github repository
  - https://github.com/sureshkuc/Freie-Universitat-Berlin/tree/main/Metagenomics/
- Technologies used: Python(Pytorch, Keras, Scikit-learn), Jupyter Notebook

References

1. Fiannaca, Antonino, et al. "Deep learning models for bacteria taxonomic classification of metagenomic data." *BMC bioinformatics* 19.7 (2018): 61-76.
2. Yuan C, Lei J, Cole J, Sun Y. Reconstructing 16S rRNA genes in metagenomic data. Bioinformatics. 2015;31(12):i35.
3. Angly FE, Willner D, Rohwer F, Hugenholtz P, Tyson GW. Grinder: a versatile amplicon and shotgun sequence simulator. Nucleic Acids Res. 2012;40(12):e94

References (continued)

4. Albanese D, Fontana P, De Filippo C, Cavalieri D, Donati C. MICCA: a complete and accurate software for taxonomic profiling of metagenomic data. Sci Rep. 2015;5:9743.

5. Chor B, Horn D, Goldman N, Levy Y, Massingham T. Genomic DNA k-mer spectra: models and modalities. Genome Biol. 2009;10(10):R108.

6. Kuksa P, Pavlovic V. Efficient alignment-free DNA barcode analytics. BMC Bioinformatics. 2009;10(14):S9.

7. https://wiki.pathmind.com/restricted-boltzmann-machine

References (continued)

8. Hayssam S, Macha N. Machine learning for metagenomics: methods and tools. Metagenomics. 2016;1:1–19

9. Wang Q, Garrity GM, Tiedje JM, Cole JR. Naive Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy. ApplEnviron Microbiol. 2007;73(16):5261–7.

10. Kultima JR, Sunagawa S, Li J, Chen W, Chen H, Mende DR, et al. MOCAT:A Metagenomics Assembly and Gene Prediction Toolkit. Plos ONE. 2012;7(10):e4765.

References (continued)

11. Min S, Lee B, Yoon S. Deep learning in bioinformatics. Brief Bioinform. 2017;18(5):851–69.

12. Lo Bosco G, Di Gangi MA. In: Petrosino A, Loia V, Pedrycz W, editors. Deep Learning Architectures for DNA Sequence Classification. Cham:Springer International Publishing; 2017, pp. 162–71.

13. Di Gangi MA, Gaglio S, La Bua C, Lo Bosco G, Rizzo R. In: Rojas I, Ortuño F, editors. A Deep Learning Network for Exploiting Positional Information in Nucleosome Related Sequences. Cham: Springer International Publishing; 2017, pp. 524–33.

References (continued)

14. Angly FE, Willner D, Rohwer F, Hugenholtz P, Tyson GW. Grinder: a versatile amplicon and shotgun sequence simulator. Nucleic Acids Res. 2012;40(12):e94.

Questions?

Thank You

Supplementary Slides

Results

Dataset	Algorithm	k	Evaluation of short-reads classification at genus level							
			Accuracy		Precision		Recall		F1	
			mean %	std	mean %	std	mean %	std	mean %	std
AMP		3	51.01	0.005	51.40	0.005	50.90	0.005	50.84	0.015
		4	77.69	0.004	77.91	0.005	77.69	0.005	77.57	0.014
	CNN	5	88.13	0.005	88.38	0.005	88.07	0.006	88.98	0.014
		6	90.92	0.005	91.14	0.005	90.91	0.005	90.82	0.009
		7	91.33	0.004	91.57	0.004	91.32	0.004	91.18	0.015
		3	56.69	0.013	57.88	0.011	56.62	0.013	55.56	0.013
		4	85.10	0.004	85.47	0.005	85.08	0.004	84.53	0.008
	DBN	5	89.82	0.003	90.12	0.004	89.82	0.003	89.63	0.004
		6	90.55	0.005	90.73	0.005	90.53	0.005	90.45	0.005
		7	91.37	0.005	91.62	0.005	91.37	0.005	91.26	0.005
	RDP	-	83.84	0.007	84.42	0.007	83.57	0.007	83.65	0.007
SG		3	17.02	0.018	17.32	0.013	16.53	0.015	16.69	0.006
		4	32.98	0.015	33.42	0.012	32.59	0.013	32.65	0.005
	CNN	5	59.80	0.015	60.34	0.014	59.41	0.015	59.31	0.005
		6	80.77	0.009	81.10	0.010	80.41	0.009	80.33	0.005
		7	85.50	0.014	85.70	0.014	85.20	0.014	85.11	0.005
		3	17.75	0.009	19.80	0.010	17.50	0.009	16.32	0.010
		4	54.11	0.007	55.62	0.007	53.67	0.007	53.17	0.007
	DBN	5	71.44	0.007	72.45	0.009	71.07	0.007	70.99	0.008
		6	77.85	0.007	78.36	0.008	77.53	0.008	77.47	0.008
		7	81.27	0.002	81.87	0.004	80.92	0.003	80.94	0.002
	RDP	-	80.38	0.009	80.83	0.008	80.18	0.008	80.09	0.009

Source: Fianacca [1]

Training Time

Table 5 Average execution time in seconds for a single fold, obtained for both training and testing models at varying of k value. Although models training require several seconds, the testing phase is quite fast, even for $k = 7$

Execution times for training and testing models

k	DBN		CNN	
	Train (s)	Test (s)	Train (s)	Test (s)
3	7288.913	0.111	686.403	0.240
4	8170.077	0.122	1256.652	0.375
5	11875.716	0.060	3091.721	0.719
6	20346.112	0.053	8021.737	1.506
7	37161.237	0.128	24204.754	3.986

Source: Fianacca [1]

Example of RDP Classifier

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

N = 4, $n(r_j = \text{'TTTCT'}) = 2$, $P_i = (n(r_i) + 0.5)/(N - 1)$

So $P_j = (2 + 0.5)/(4 - 1) = 2.5 / 3 = 0.8333$

Example

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

N = 4, n(r_j = 'AAAT') = 2, $P_i = (n(r_i) + 0.5)/(N - 1)$

So P_j = (2 + 0.5)/(4 - 1) = 2.5 / 3 = 0.8333

Example

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

$$P(s|g_i) = \prod_{r_j \epsilon V_i} \frac{m(r_j)+P_j}{M_i+1}$$

$$P(s = \text{“ } TTAAATGGGG \text{ ”} \; | g_i = \text{“ } Moritella \text{ ”}) = \frac{2+0.8888}{2+1}$$

Example

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

$$P(s|g_i) = \prod_{r_j \epsilon V_i} \frac{m(r_j)+P_j}{M_i+1}$$

$$P(s = \text{“}\ TTAAATGGGG\ \text{”}\ |g_i = \text{“}\ Moritella\ \text{”}) = 0.96$$

Example

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

$$P(s|g_i) = \prod_{r_j \epsilon V_i} \frac{m(r_j)+P_j}{M_i+1}$$

$$P(s = \text{"}\ TTAAATGGGG\ \text{"}\ |g_i = \text{"}\ Kingella\ \text{"}) = \frac{0+0.8888}{2+1}$$

Example

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

$$P(s|g_i) = \prod_{r_j \epsilon V_i} \frac{m(r_j)+P_j}{M_i+1}$$

$$P(s = \text{“}\ TTAAATGGGG\ \text{”}\ |g_i = \text{“}\ Kingella\ \text{”}) = 0.2962$$

Example

Sequence	TTTCT	AAAT	Taxa label
AAATCTTTTTT	0	1	Moritella
TTTCTAAAAAA	1	0	Kingella
GGGTTTCTAAA	1	0	Kingella
TTAAATCTTTT	0	1	Moritella
TTAAATGGGG	0	1	??????????

$$i = argmax_z P(s|g_z)$$
$$i = argmax_z(0.96, 0.2962)$$
$$i = \text{" } Moritella \text{ "}$$