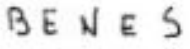


Handwritten Recognition

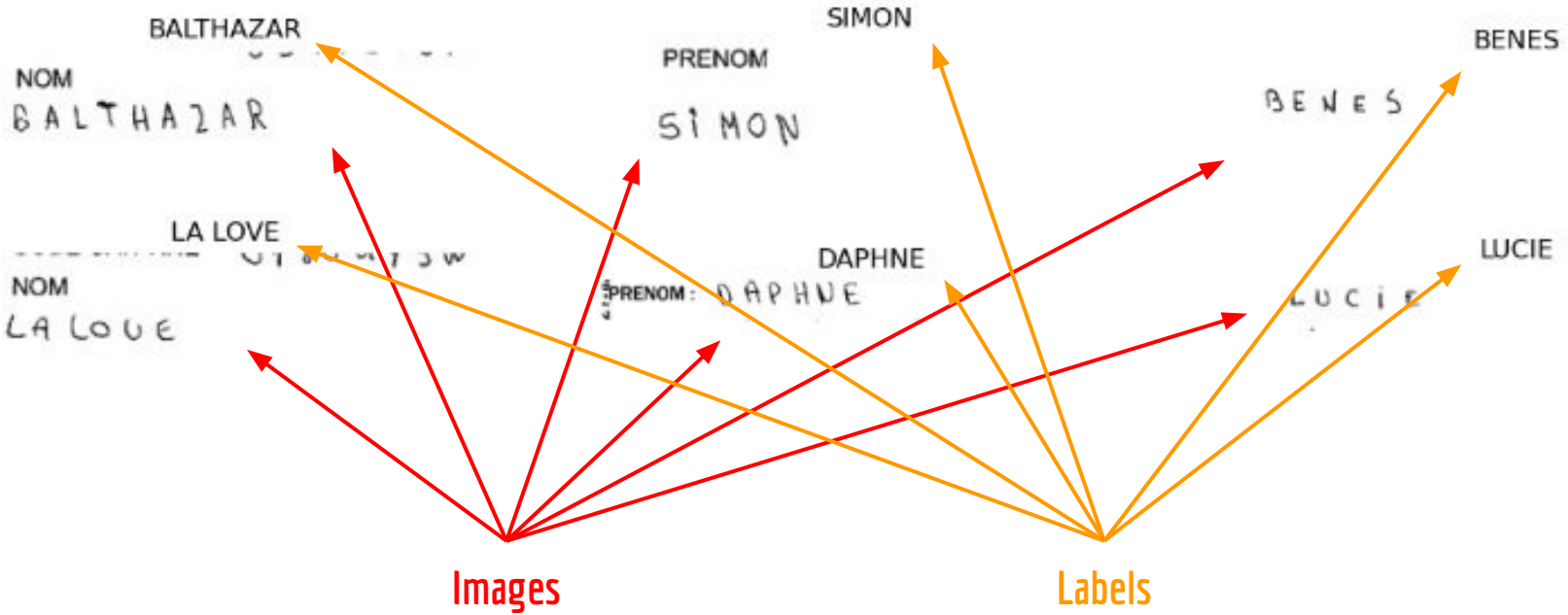
Neural Networks and Deep Learning

Andreu Gascón
Mathias Lommel
Pere Mayol

UAB
Universitat Autònoma de Barcelona

Context

Context


BALTHAZAR
NOM
BALTHAZAR
SIMON
PRENOM
SIMON
BENES
BENES
LA LOVE
NOM
LA LOVE
DAPHNE
PRENOM : DAPHNE
LUCIE
LUCIE
Images
Labels

The data

The Data

3 datasets : Training / Validation / Test

For each dataset : 2 types of file

- .zip file : containing the images

  → **330961** training / **41370** test and validation

- .csv file : linking each **image** with its **label**

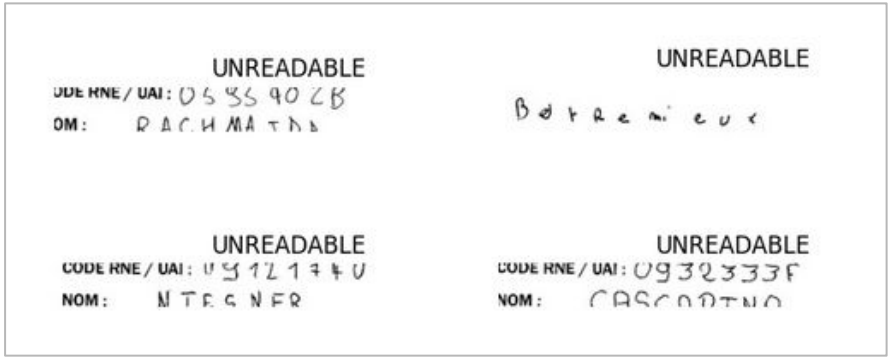
TRAIN	FILENAME	IDENTITY
0	TRAIN_00001.jpg	BALTHAZAR
1	TRAIN_00002.jpg	SIMON
2	TRAIN_00003.jpg	BENES
3	TRAIN_00004.jpg	LA LOVE
4	TRAIN_00005.jpg	DAPHNE

The Data

For the labels :

→ Delete existing **NaN values**

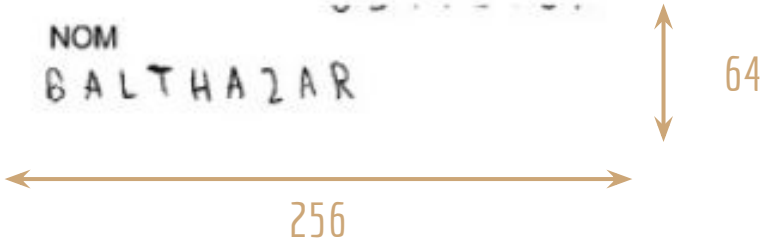
→ Delete images labelled **'UNREADABLE'**



For the images :

→ Cropping the image to a shape **256x64**

→ Change type to **float**



The Data

Finally...

⟶ Code the labels into vectors

⟶ Using a certain alphabet : " ABCDEFGHIJKLMNOPQRSTUVWXYZ-' "

⟶ Padded into a length 24, with -1 values

WORD  [23, 16, 19, 5, -1, -1, -1, ..., -1]

Longest word found : 34 characters

Second longest : 24 characters

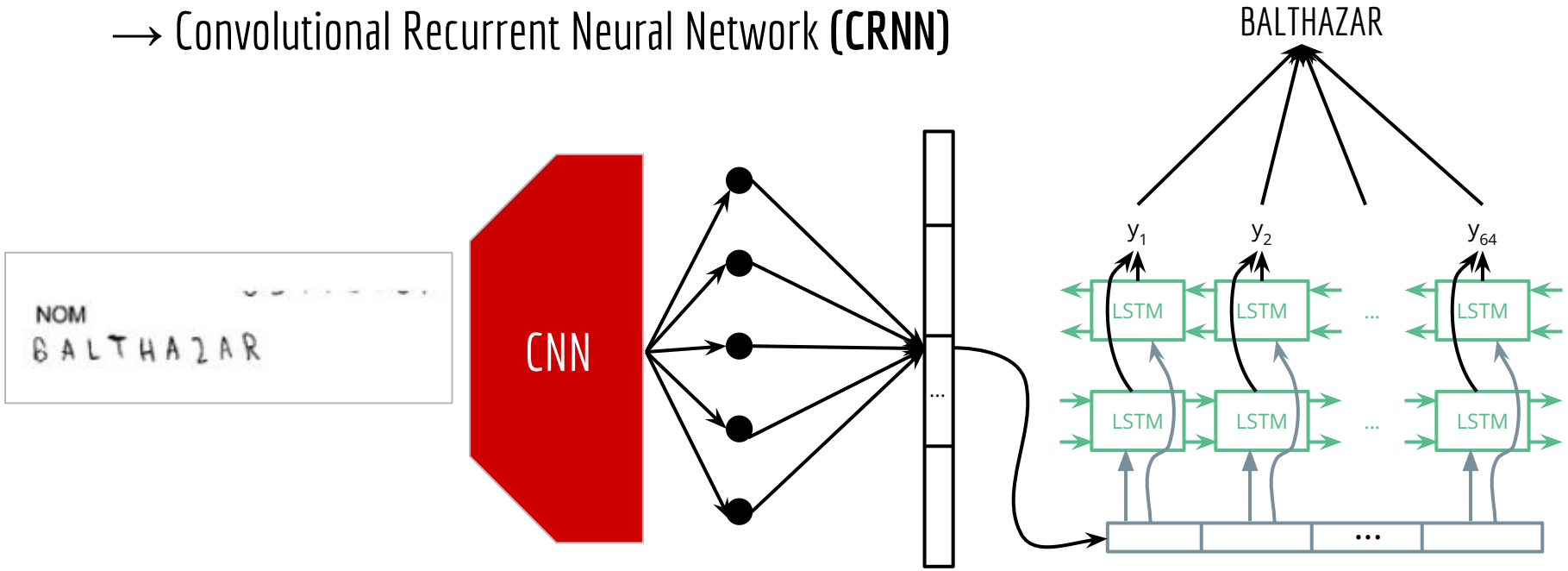
⟶ We deleted the 34 character image from the training set

Models

Models

⟶ 1 model - 2 versions

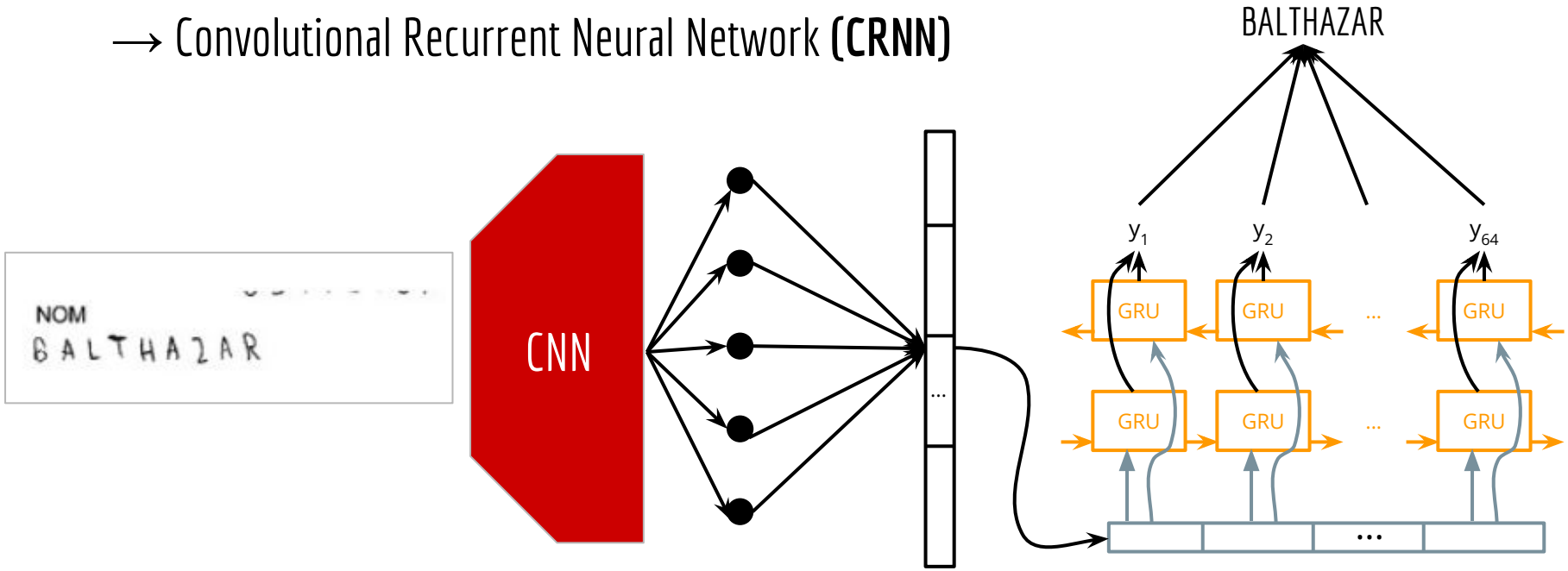
⟶ Convolutional Recurrent Neural Network **(CRNN)**



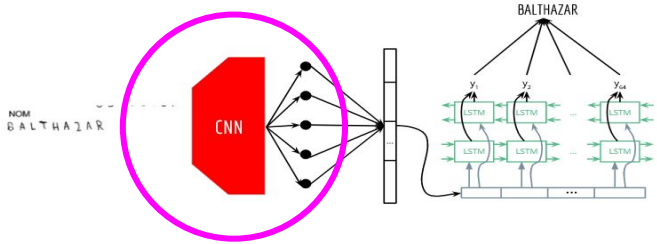
Models

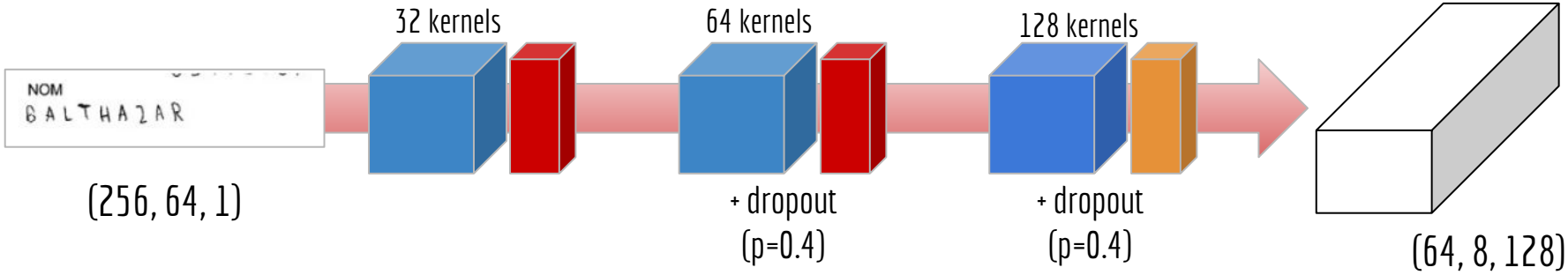
⟶ 1 model - 2 versions

⟶ Convolutional Recurrent Neural Network **(CRNN)**



Models





Filter 3x3 / ReLU

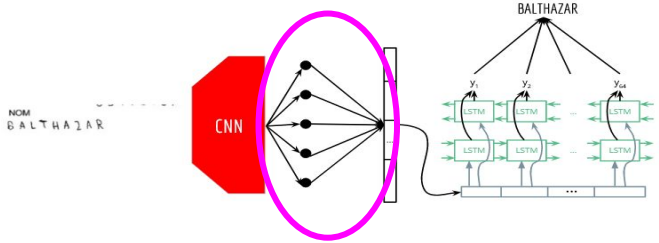
Maxpool (2,2)

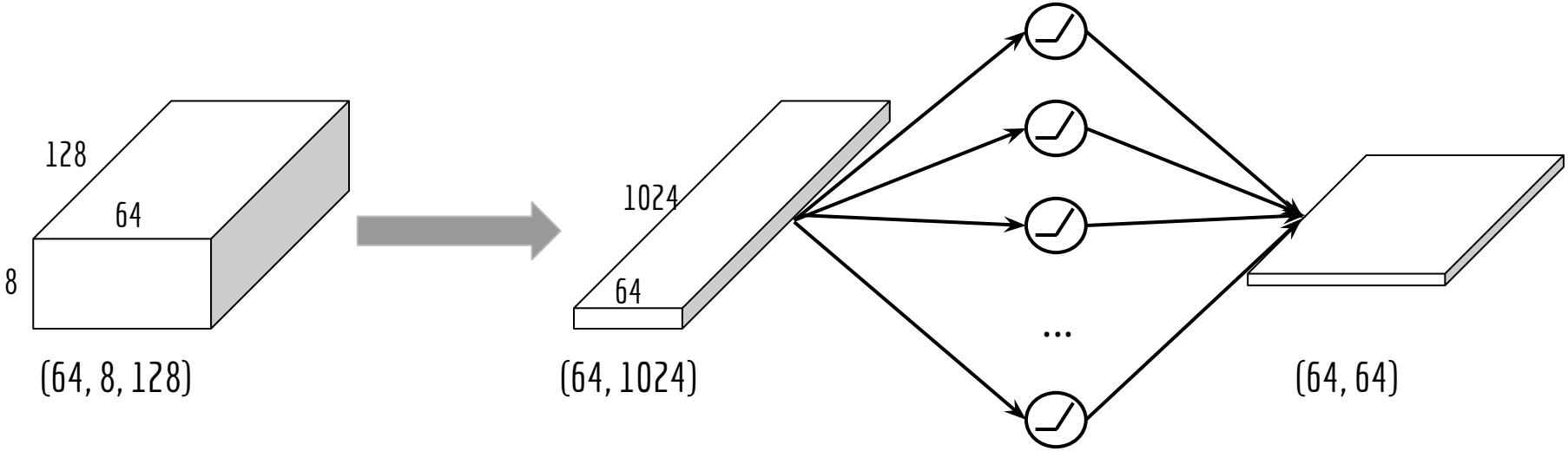
Maxpool (1,2)

*For each **convolutional** operation :*

- *Batch normalisation*
- *He normal initialisation*

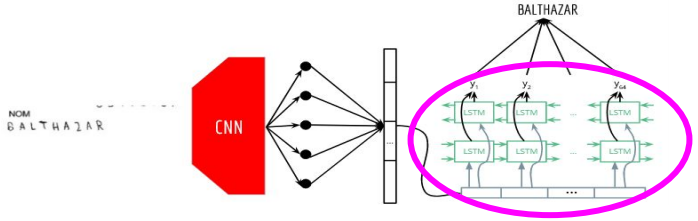
Models

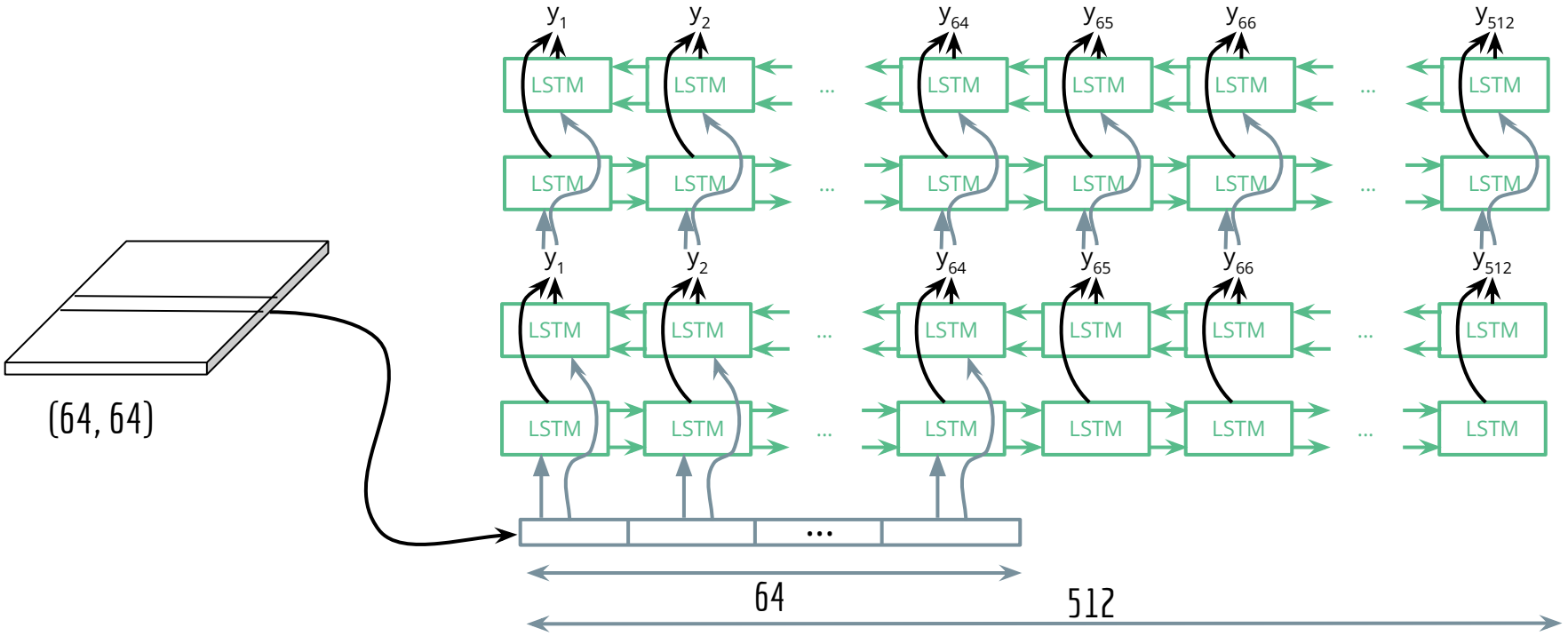




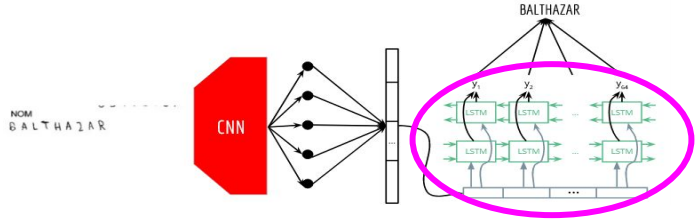
64 neurons - ReLU
He normal initialisation

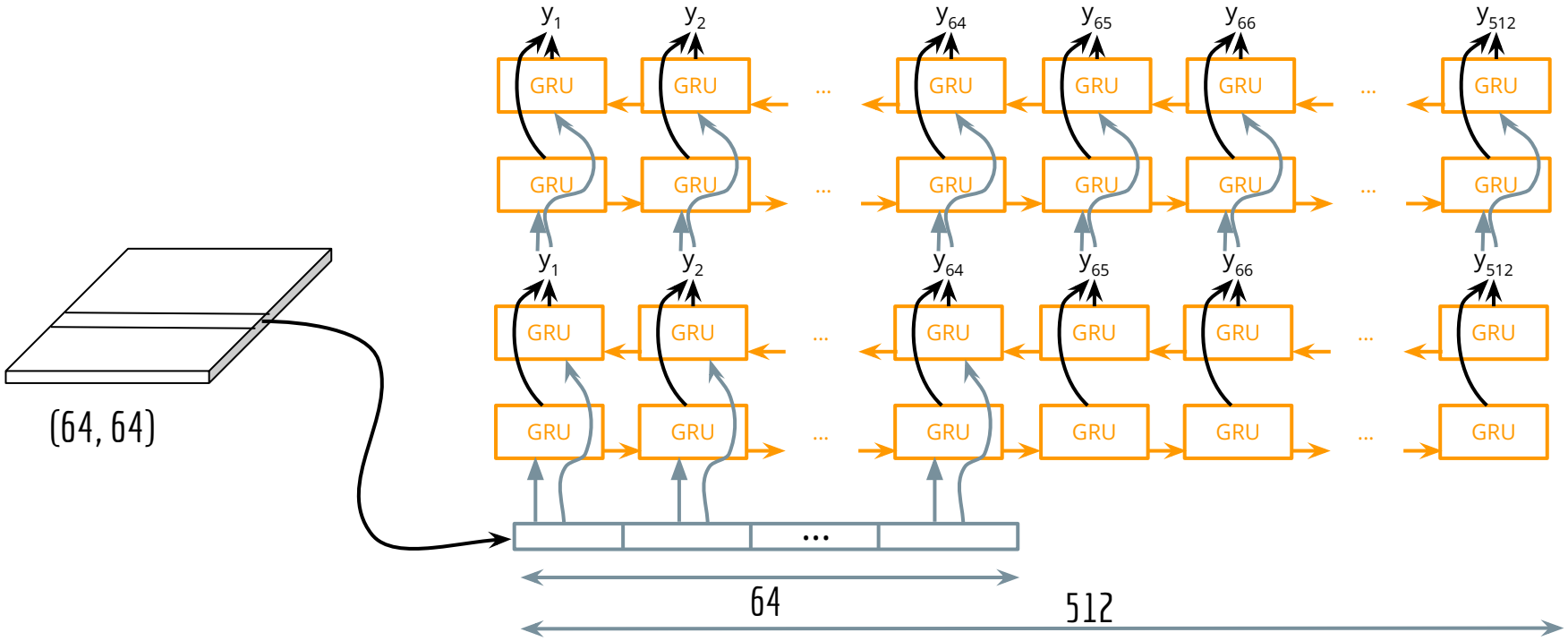
Models



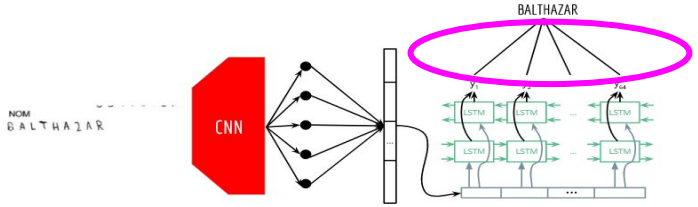


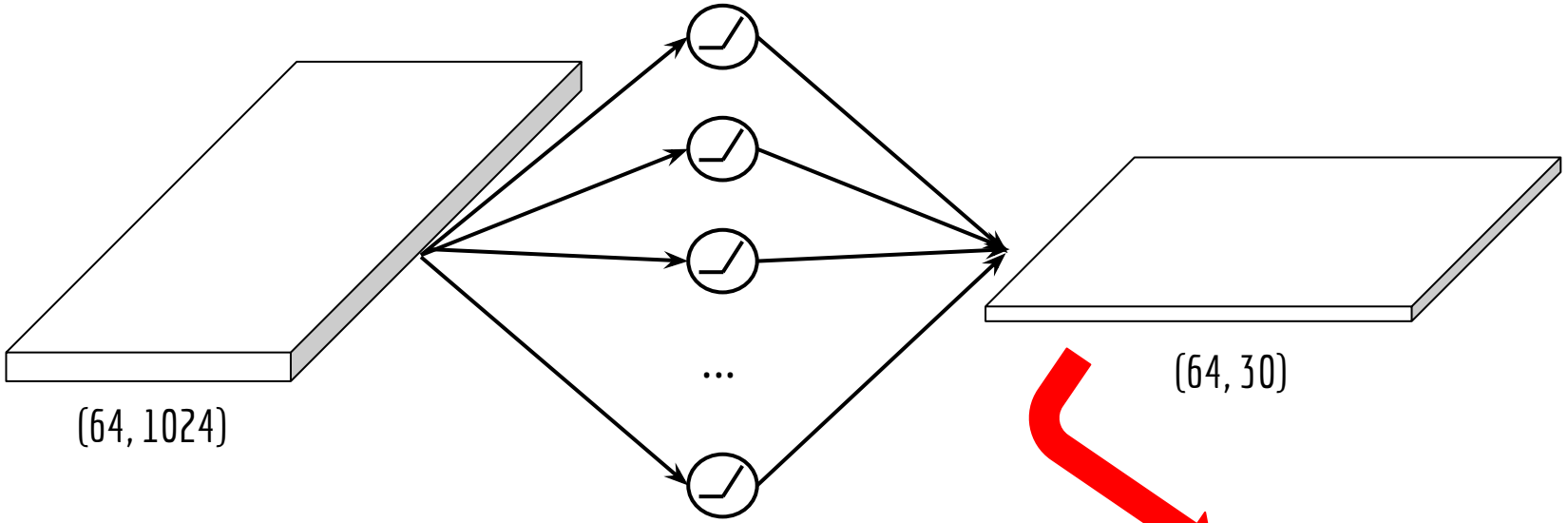
Models



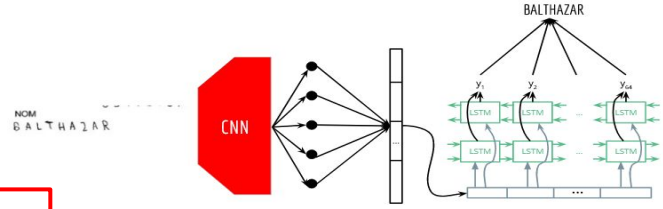


Models

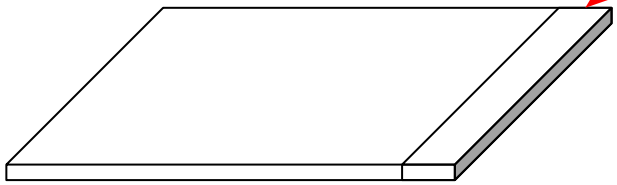

BALTHAZAR
NOM
BALTHAZAR
CNN
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM


(64, 1024)
...
(64, 30)
30 neurons - ReLU
He normal initialisation
Log-softmax

Models



Why 64x30 ?



Probability distribution

alphabet = " ABCDEFGHIJKLMNOPQRSTUVWXYZ-'"

$$Pr(\pi|\mathbf{x}) = \prod_{t}^{T} y_{\pi_t}^{t}$$

Joint probability of a path

- 'WORD'
- → 'WWWOORRRDDD'
- → 'WOOOOORRDDD'
- → ...

Connectionist Temporal Classification Loss

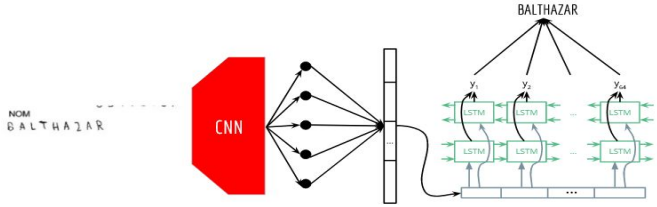
$$Pr(\mathbf{l}|\mathbf{x}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{l})} Pr(\pi|\mathbf{x})$$

Sum over all possible paths

(CTC Loss)

$$\mathcal{L}_{CTC} = -\sum_{(\mathbf{x},\mathbf{l}) \in Z} \ln Pr(\mathbf{l}|\mathbf{x})$$

Models

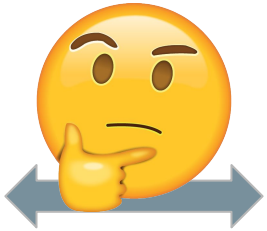


How to obtain the real label ?

By taking each max prob, we obtain : [23, 23, 16, 16, 16, 19, 19, 19, 19, 5, 5, -1, -1, -1, ..., -1]

64 characters

[23, 23, -, 16, 16, 16, -, 19, 19, 19, 19, -, 5, 5, -, -1, -1, ..., -1]



[23, 16, 19, 5, -1, -1, -1, ..., -1]

W	W	-	O	O	O	-	R	R	R	R	-	D	D
W	-	O	-	R	-	D							
W	O	R	D										

Training

Training setup

First training

Size of the training sample : **64 000 images**

Size of the validation sample : **6 400 images**

Batch size : **128**

Adam optimizer

Learning rate of **0.001**

10 epochs

Full training

Size of the training sample : **300 800 images**

Size of the validation sample : **30 080 images**

Batch size : **128**

Adam optimizer

Learning rate of **0.001**

10 epochs

Tests

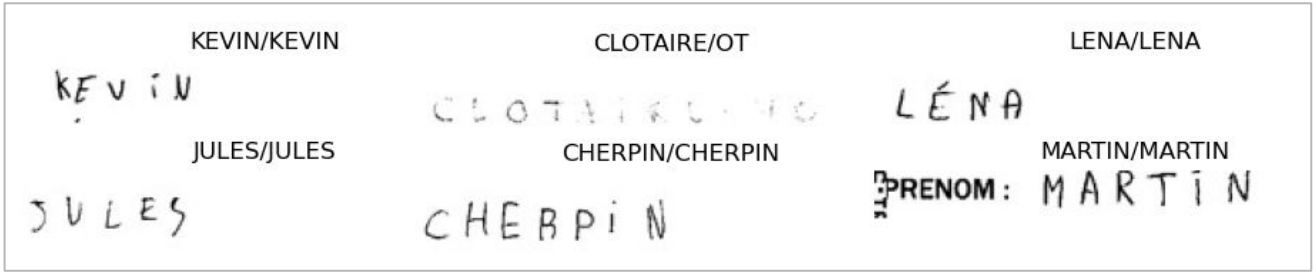
Tests - first training

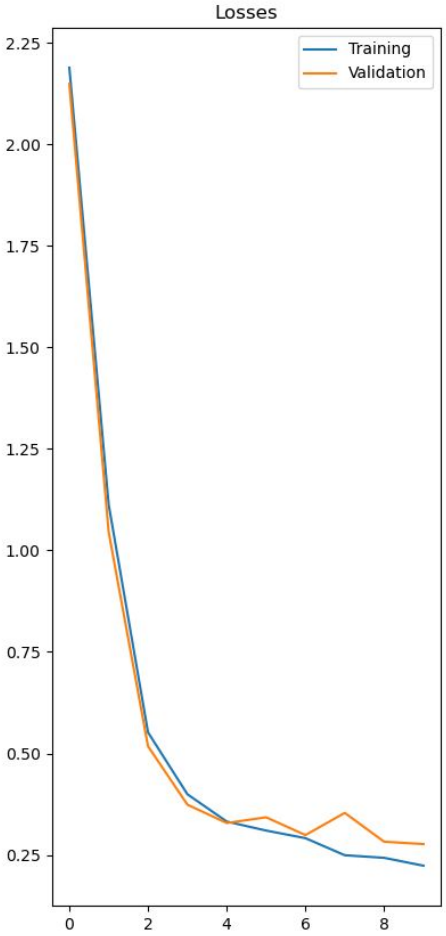
GRU model :

- Best word accuracy : **74.4%**
- Best letter accuracy : **89.9%**

6 689 822 parameters

No overfitting





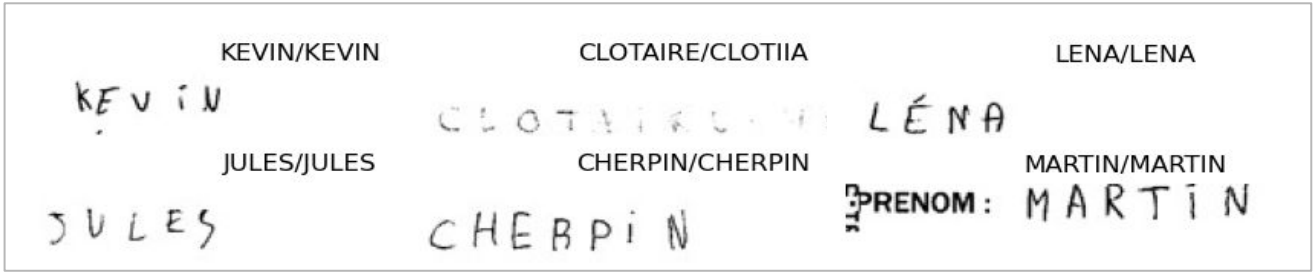
Tests - first training

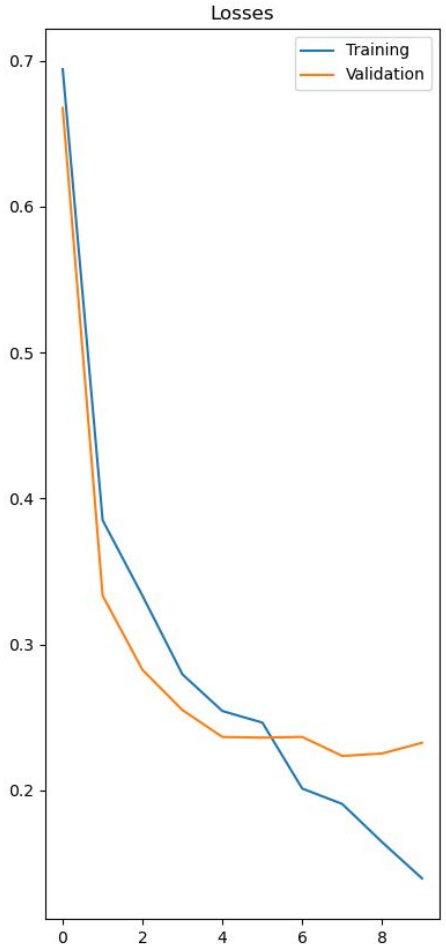
LSTM model :

- Best word accuracy : **78.1%**
- Best letter accuracy : **91.3%**

8 856 606 parameters

An **overfitting** begins (more parameters)





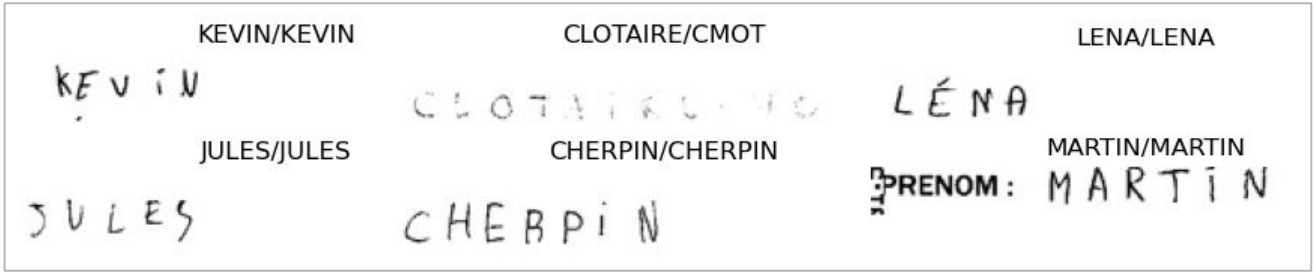
Tests - full training

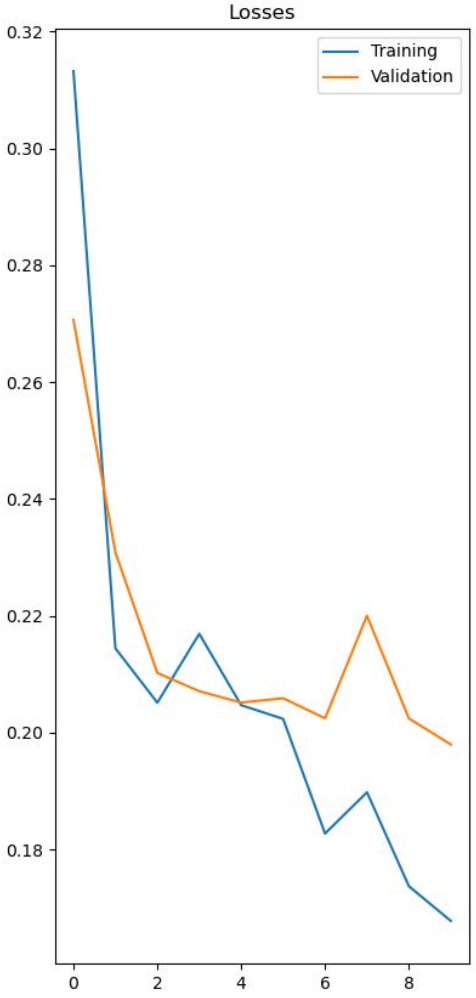
GRU model :

- Best word accuracy : **81 %** (+6.6%)
- Best letter accuracy : **92%** (+2.1%)

6 689 822 parameters

An **overfitting** begins to occur





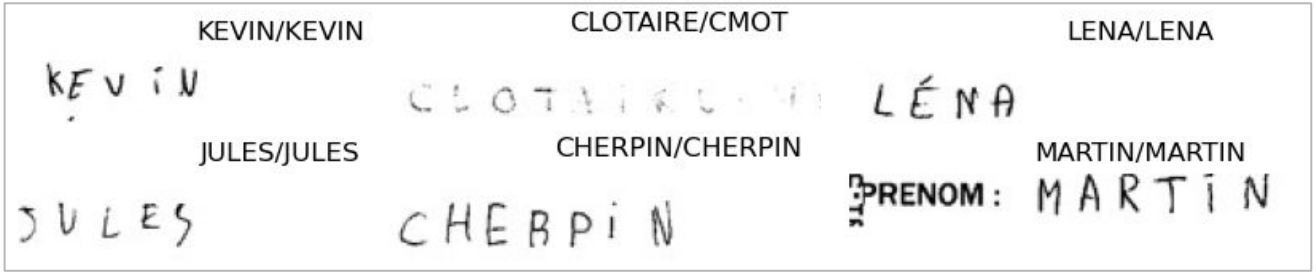
Tests - full training

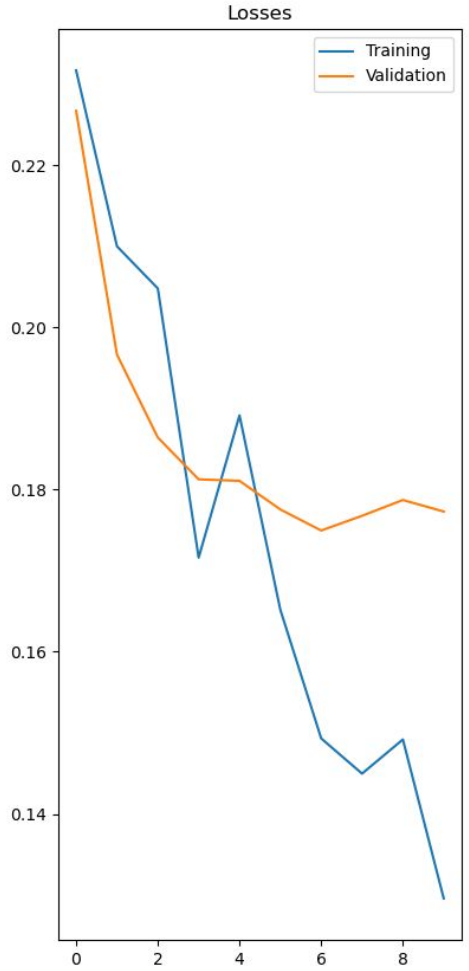
LSTM model :

- Best word accuracy : **82.8%** (+4.7%)
- Best letter accuracy : **92.5%** (+1.5%)

8 856 606 parameters

Clear **overfitting**





Tests - Our best model

Top **mispredicted** letters :

'N' predicted as 'M'

'H' predicted as 'M'

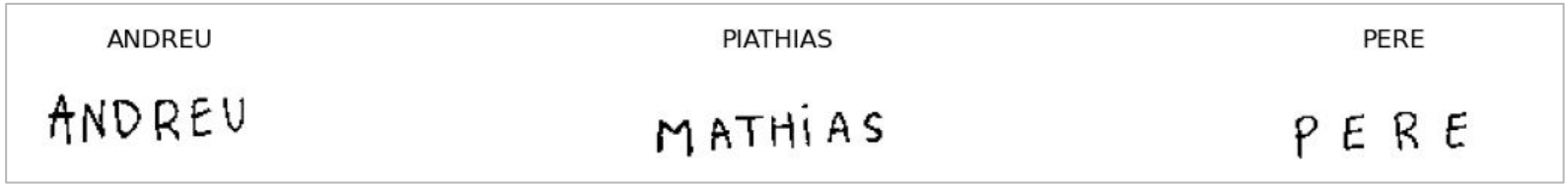
'O' predicted as 'U'

→ Similar letter shapes : the model has learnt something

On **mispredicted** words :

→ on average, **64.5%** of the letters are well predicted

Test on homemade images :



Tests - Comparison of the models

	GRU (first)	LSTM (first)	GRU (full)	LSTM (full)
Word accuracy	74.4	78.1	81	82.8
Letter accuracy	89.9	91.3	92	92.5
Time of training	6'37"	7'21"	31'12"	33'31"
Overfitting ?	x	+	+	++
Number of parameters	6 689 822	8 856 606	6 689 822	8 856 606

Tests - Our best model

Kaggle Baseline Model

*Correct characters predicted : **82.16%***
*Correct words predicted: **69.10%***

Problems

Double letters problem	→	*Changing the decoding function.*
Poor performance for personal images	→	*Preprocess the images properly.*
Nan values for weights	→	*Changing the dimension for log_softmax(x,dim=2)*

Conclusion

Thanks for listening