Importance of recurrent layers for unconstrained handwriting recognition

Denis Coquenet, Yann Soullard, Clément Chatelain, Thierry Paquet

LITIS Laboratory - EA 4108 Normandie University - University of Rouen, France

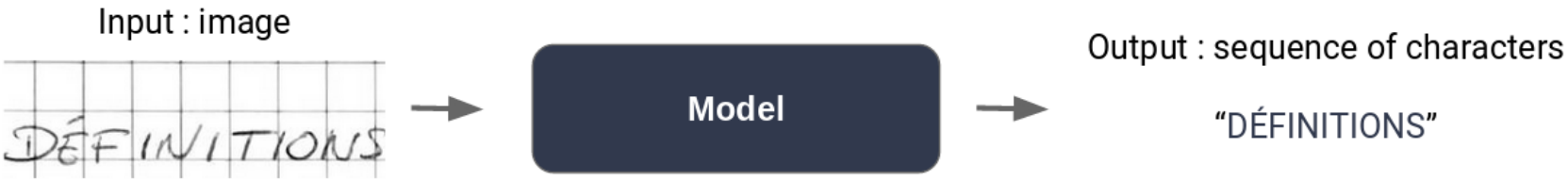
SIFED, 6th June 2019

Table of Contents

Handwriting recognition system

Constraints

- Images (input) of variable size
- Sequence of characters (output) of variable length

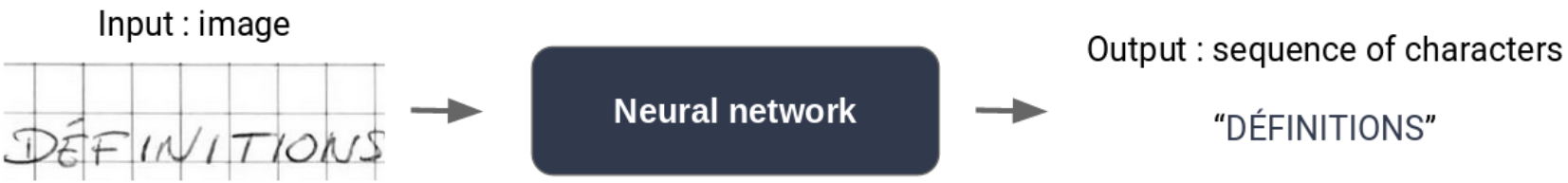


- Towards a heavy use of neural networks

Deep learning handwriting recognition system

Architecture

- Recurrent layers (LSTM) and/or non-recurrent ones (CNN)
- Language model inclusion



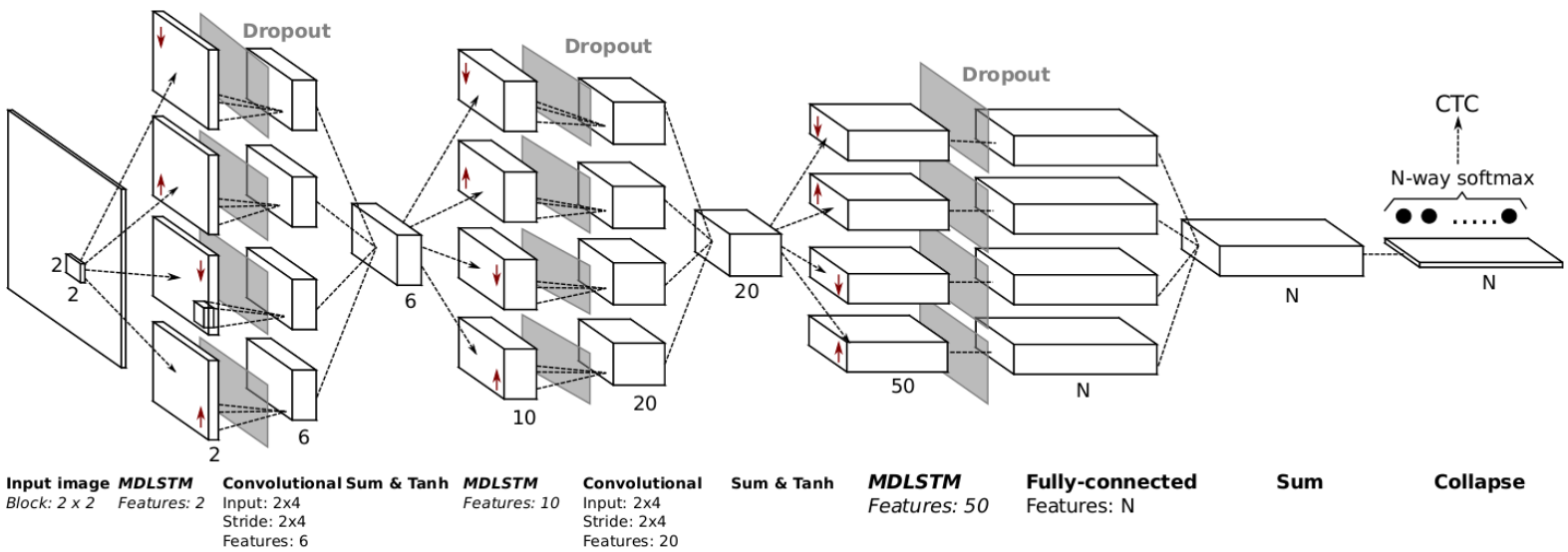
Sequence alignment

Connectionist Temporal Classification (CTC)

- Focus on optical model only without language model nor lexicon constraints

State of the art

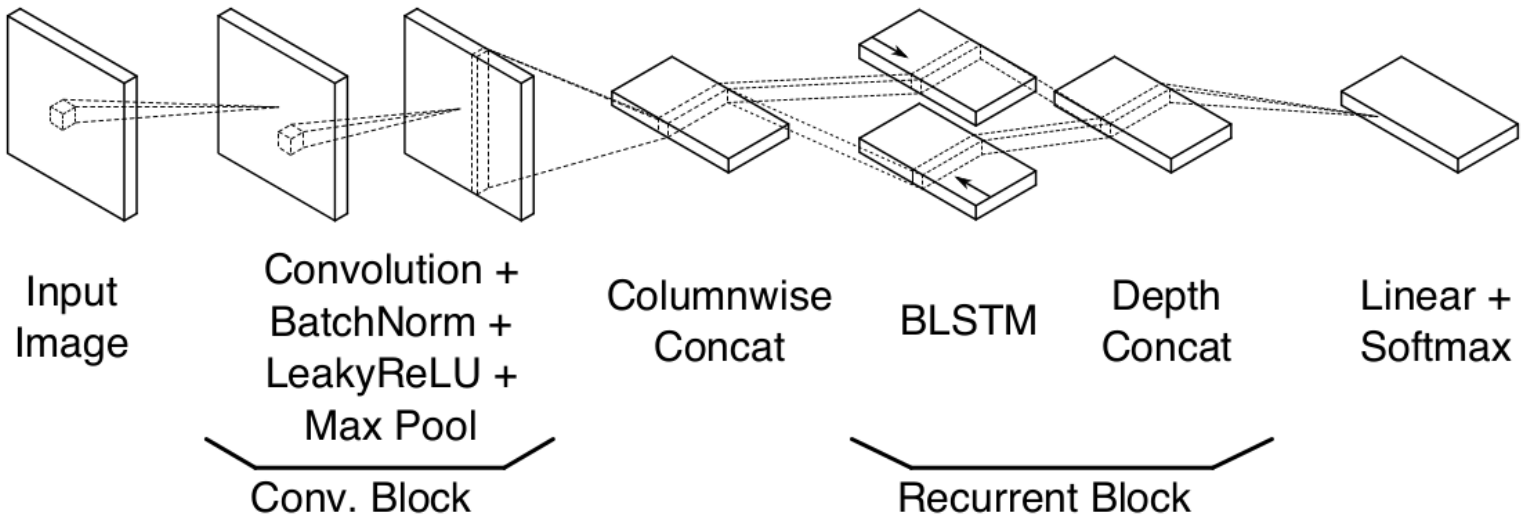
Recurrent models - Multi-Dimensional Long-Short Term Memory (MDLSTM) [Pham2014].



- Recurrence over horizontal and vertical axis (in both directions) : 4 LSTM/layer

State of the art

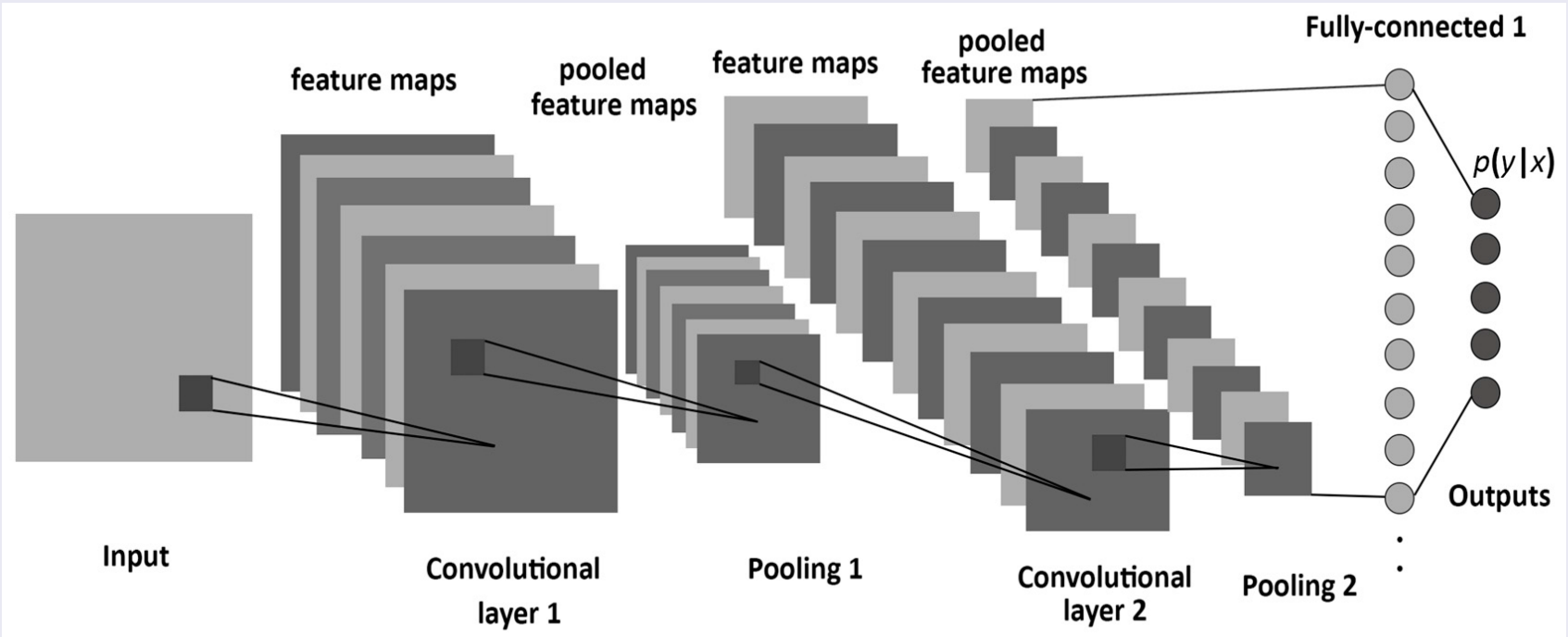
Recurrent models - Convolutional Neural Network + Bidirectional Long-Short Term Memory (CNN+BLSTM) [Puigcerver2017].



- Recurrence over horizontal axis only (in both directions) : 2 LSTM/layer

State of the art

Non-recurrent models - Convolutionnal Neural Network (CNN)



State of the art

CNN for handwriting recognition

- Fully Convolutional Networks (FCN) + CTC [Ptucha2018]
  - Standard CNN without dense layer
- FCN with gating mechanism + CTC [Yousef2018]
  - Gates (tanh, sigmoid)
  - Residual connections
  - Depthwise Separable Convolutions
  - High normalization (batch & layer)
- FCN with gating mechanism + CTC[Ingle2019]
  - Gates (ReLU, sigmoid)
  - Shared weight layers

Context

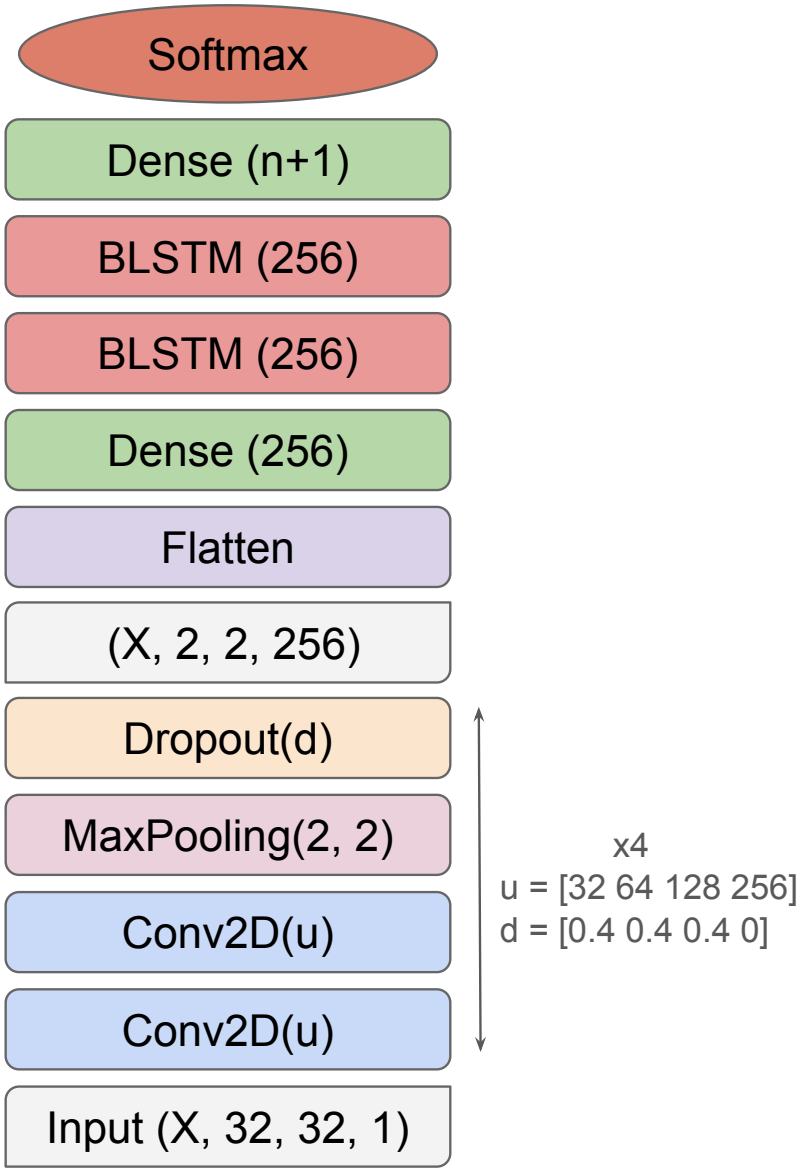
Objective

Design a convolutional network competitive with recurrent ones to reduce the training time : G-CNN

Main questions

- Are recurrent layers really necessary for handwriting recognition ?
- Are CNN really lighter than recurrent model in terms of parameters ?
- Is it possible to easily obtain competitive results without recurrence ?

Our baseline model - CNN+BLSTM

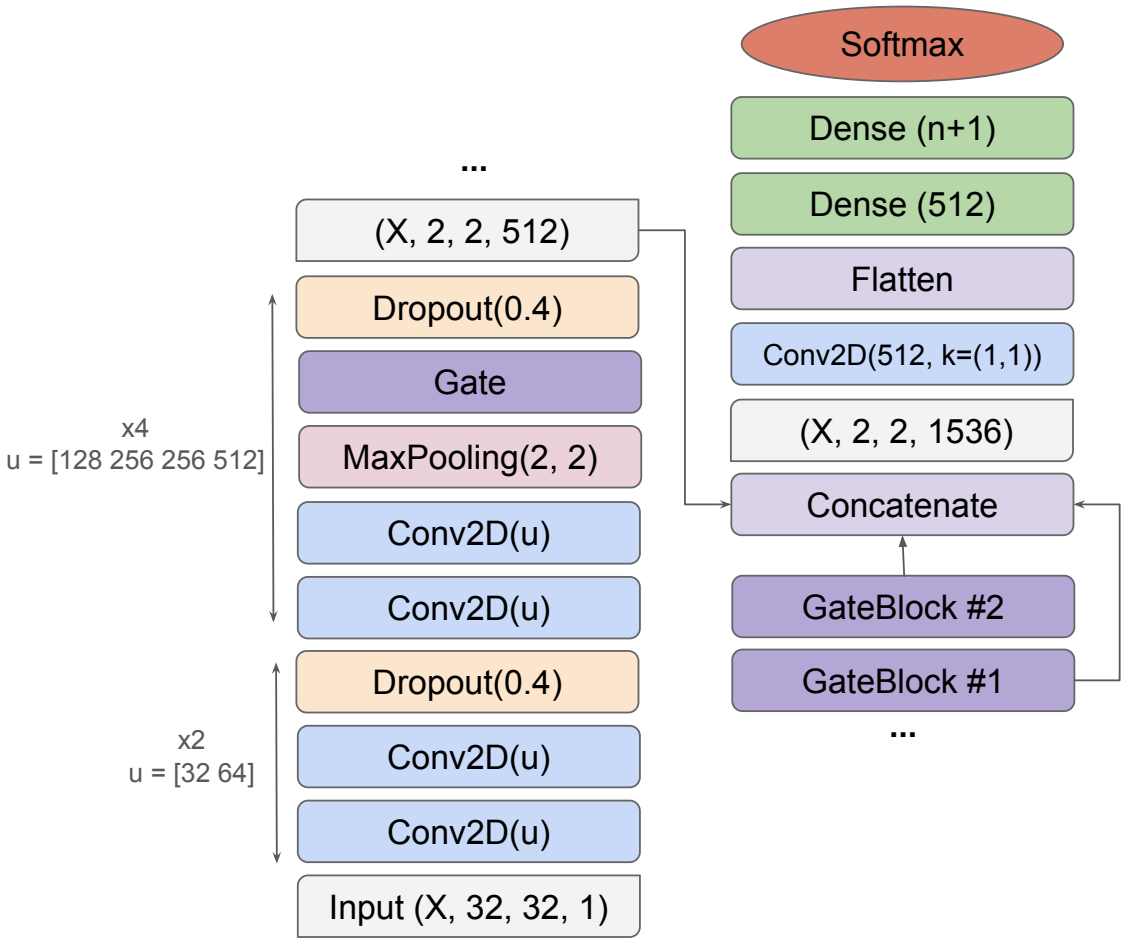


Features

- From [Soullard2019] (state-of-the-art results)
- Recurrent model
- 8 convolutions
- 2.5 million of parameters

n : number of characters in the alphabet

Our G-CNN - overview

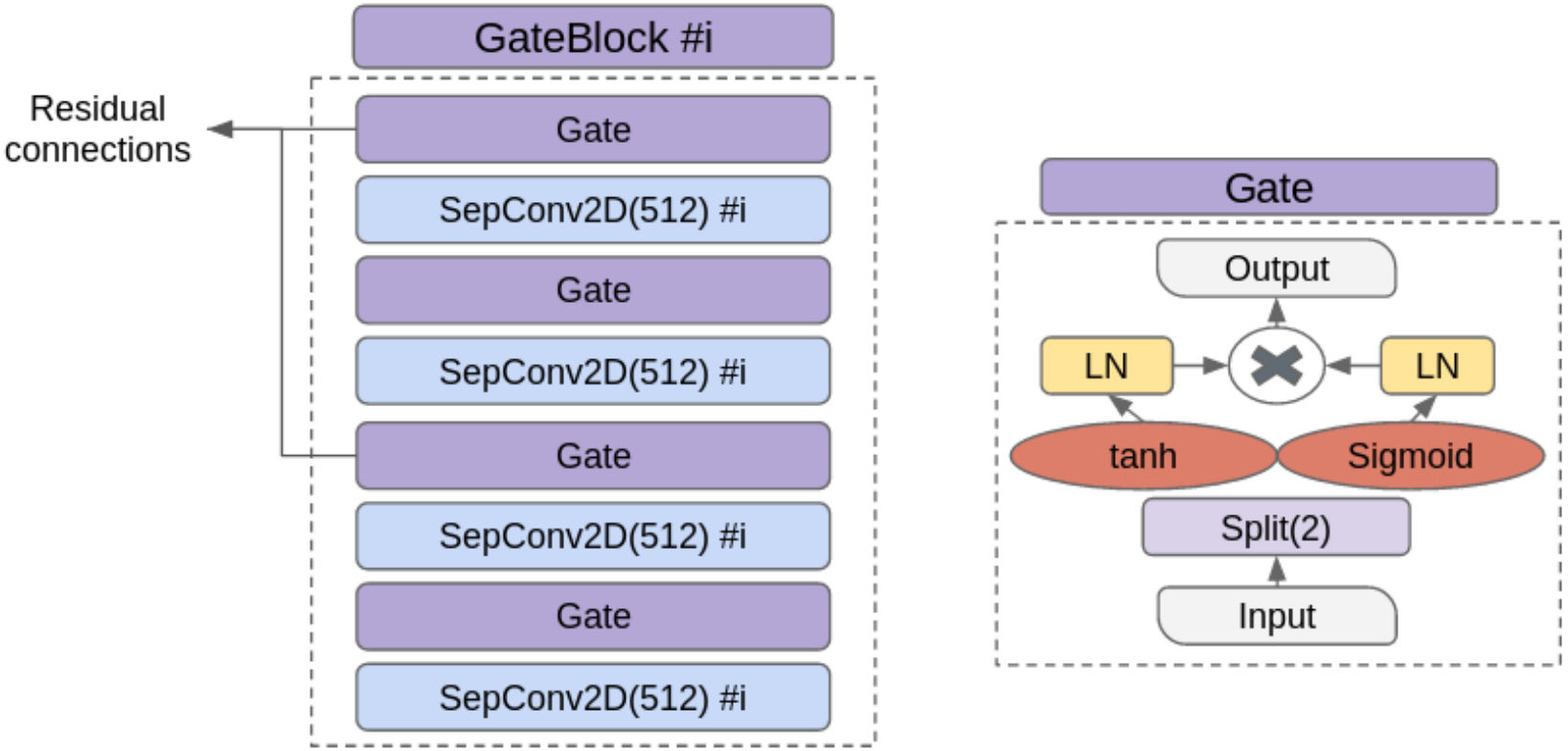


Features

- Based on baseline model
- Non-recurrent model
- 21 convolutions
- 6.9 million of parameters
- Shared weight layers
- Depthwise Separable Convolutions
- Residual connections

Our G-CNN - gates

Gating mechanism



Training details

Hyperparameters

- Sliding window : 32x32 px
- Loss : CTC
- Optimizer : Adam
- Initial learning rate : 10^{-4}
- Momentum : 0.9

Model evaluation

Criteria

- Character Error Rate (CER)
- Number of parameters
- Training time

Raw model comparison

- We focus only on the network performance alone
- No language model
- No lexicon constraints

Dataset

RIMES (lines)

RIMES dataset

Dataset characteristics

- +1,300 writers
- French writings
- 12,723 pages segmented into lines

RIMES dataset split

Training	Validation	Test	Alphabet
9,947	1,333	778	100

Example

First experiment : Raw comparison

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
CNN+BLSTM	**6.98**	**6.88**	1d22h59	4.1
CNN+Dense only	17.73	19.03	1h10	1.5
G-CNN	9.92	10.03	10h00	6.9

- BLSTM layers responsible for a large amount of parameters (2.6 M)

First experiment : Raw comparison

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
CNN+BLSTM	**6.98**	**6.88**	1d22h59	4.1
CNN+Dense only	17.73	19.03	1h10	1.5
G-CNN	9.92	10.03	10h00	6.9

- BLSTM layers responsible for a large amount of parameters (2.6 M)
- BLSTM layers increase performance dramatically (-12.15% in test)

First experiment : Raw comparison

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
CNN+BLSTM	**6.98**	**6.88**	1d22h59	4.1
CNN+Dense only	17.73	19.03	1h10	1.5
G-CNN	9.92	10.03	10h00	6.9

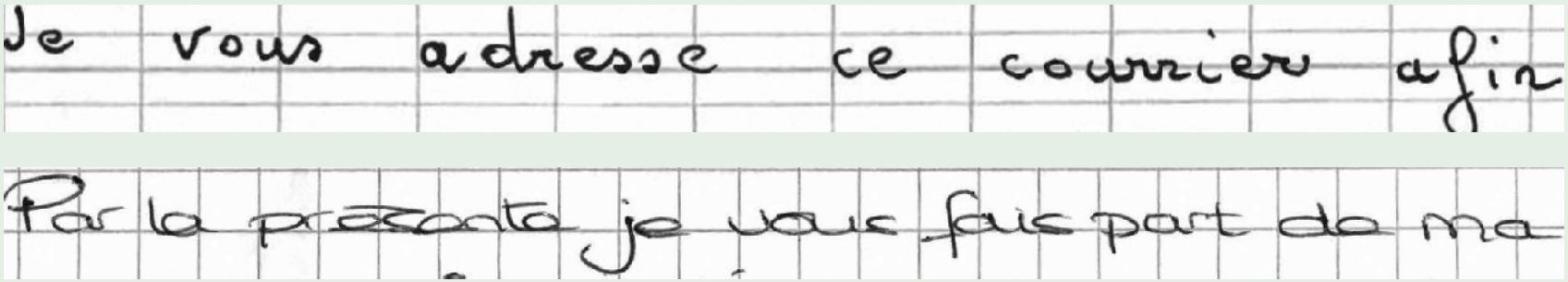
- BLSTM layers responsible for a large amount of parameters (2.6 M)
- BLSTM layers increase performance dramatically (-12.15% in test)
- G-CNN : more parameters but training time shorter (parallel computing)

Second experiment - Robustness against complexified data

Modified version of RIMES dataset

Lined paper background addition

Examples

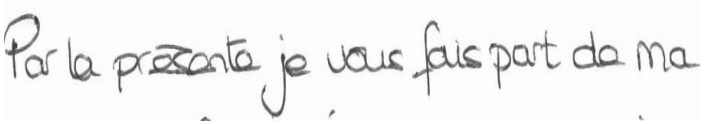
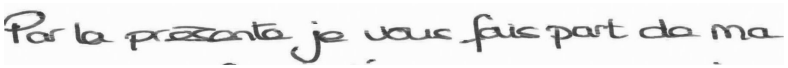


Second experiment - Robustness against complexified data

Architecture	Background	CER(%) validation	CER (%) test	Training time
CNN+BLSTM	Without	6.98	6.88	1d22h59
	With	8.81	9.27	1d1h29
G-CNN	Without	9.92	10.03	10h00
	With	11.70	12.55	8h27

- Similar behavior - CER increased by 2.39% for the CNN+BLSTM and 2.52% for the G-CNN (in test)

Third experiment - Impact of data augmentation

	RIMES
1. Raw	
2. Contrast	
3. Sign flipping	
4. Long scaling	
5. Short scaling	
6. Width dilation	
7. Height dilation	

Third experiment - Impact of data augmentation

Architecture	Data augmentation	CER(%) - validation	CER (%) - test
CNN+BLSTM	Without	6.98	6.88
	With	6.59	5.94
G-CNN	Without	9.92	10.03
	With	8.93	8.73

- CER decreased by 1.30% for the G-CNN and 0.94% for the CNN+BLSTM
- Assumption : G-CNN needs more examples whereas CNN+BLSTM compensates with its use of context

Ablation study - Part 1

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
G-CNN	9.92	10.03	10h00	6.9
(1) Only standard convolutions	**10.02**	**9.97**	**6h41**	**9.0**
(2) Max pooling from the very beginning	13.31	13.35	2h57	6.9
(3) No shared weight layers	9.78	9.85	8h54	7.7

- (1) Depthwise Separable Convolutions enables saving 2.1 M of parameters preserving the performance (+0.06%)

Ablation study - Part 1

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
G-CNN	9.92	10.03	10h00	6.9
(1) Only standard convolutions	10.02	9.97	6h41	9.0
(2) Max pooling from the very beginning	**13.31**	**13.35**	**2h57**	**6.9**
(3) No shared weight layers	9.78	9.85	8h54	7.7

- (1) Depthwise Separable Convolutions enables saving 2.1 M of parameters preserving the performance (+0.06%)
- (2) Delaying the use of max pooling increases the performance (by 3.32%)

Ablation study - Part 1

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
G-CNN	9.92	10.03	10h00	6.9
(1) Only standard convolutions	10.02	9.97	6h41	9.0
(2) Max pooling from the very beginning	13.31	13.35	2h57	6.9
(3) No shared weight layers	**9.78**	**9.85**	**8h54**	**7.7**

- (1) Depthwise Separable Convolutions enables saving 2.1 M of parameters preserving the performance (+0.06%)
- (2) Delaying the use of max pooling increases the performance (by 3.32%)
- (3) Shared weight layers enable saving 0.8 M parameters, with a similar CER (+0.18% only)

Ablation study - Part 2

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
G-CNN	9.92	10.03	10h00	6.9
(4) Doubled convolutions in GateBlocks	**9.96**	**10.15**	**4h10**	**7.4**
(5) Removal of the 2 GateBlocks	10.09	10.33	6h37	6.1

- (4) Increasing the number of convolutions between gates is not necessary (+0.12%)

Ablation study - Part 2

Architecture	CER(%) validation	CER (%) test	Training time	Parameters (M)
G-CNN	9.92	10.03	10h00	6.9
(4) Doubled convolutions in GateBlocks	9.96	10.15	4h10	7.4
(5) Removal of the 2 GateBlocks	**10.09**	**10.33**	**6h37**	**6.1**

- (4) Increasing the number of convolutions between gates is not necessary (+0.12%)
- (5) The majority of the work is done before the GateBlocks (+0.3%)

Conclusion - recurrent models

Structure

- Convolutional part (CNN): feature extraction
- Recurrent part (LSTM): sequence modeling

Advantages

- Performance
- Simple architectures

Drawbacks

Recurrent models have long training times:

- LSTM implies a large amount of parameters
- Recurrence implies sequential computations

Conclusion - G-CNN models

Structure

- Feature extraction similar to CNN+BLSTM
- Gating mechanism to filter information

Advantages

Convolution = parallelizable operation & few parameters

- Reduced training time
- Deeper networks, bigger receptive fields

Drawbacks

- Number of hyperparameters, hard tuning
- Complex architecture
- Performance hardly competitive

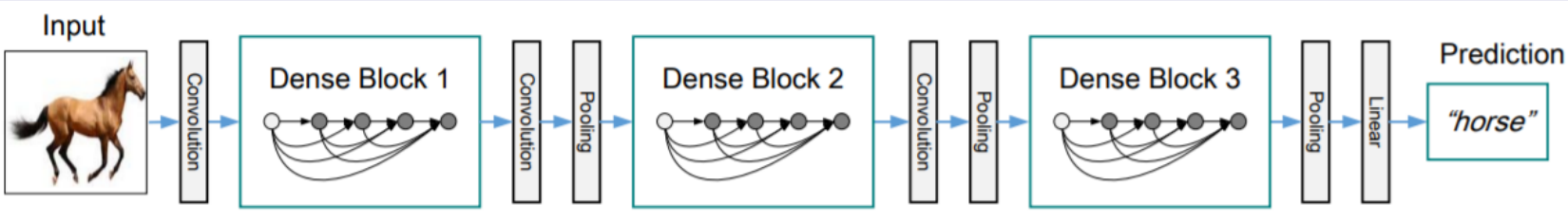
Future works

Toward an even lighter network

Give up densely connected layers to build a Fully Convolutionnal Network

Exploring other alternatives

- Attention models [Chowdhury2018]
- Dense net [Huang2016]



References

[Pham2014] V. Pham et al. "Dropout Improves Recurrent Neural Networks for Handwriting Recognition". In: *ICFHR* (2014).

[Huang2016] Huang et al. *Densely Connected Convolutional Networks*. 2016.

[Puigcerver2017] J. Puigcerver. "Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?". In: *ICDAR*. 2017, pp. 67–72.

[Chowdhury2018] C. Arindam et al. *An Efficient End-to-End Neural Model for Handwritten Text Recognition*. 2018.

[Yousef2018] M. Yousef et al. *Accurate, Data-Efficient, Unconstrained Text Recognition with Convolutional Neural Networks*. 2018.

[Ptucha2018] Felipe Petroski Such et al. "Intelligent Character Recognition using Fully Convolutional Neural Networks". In: *Pattern Recognition* 88 (Dec. 2018).

[Soullard2019] Y. Soullard et al. *CTCModel: a Keras Model for Connectionist Temporal Classification*. 2019.

[Ingle2019] R. Ingle et al. *A Scalable Handwritten Text Recognition System*. 2019.