Handwritten text recognition: from isolated text lines to whole documents

Denis Coquenet, Clément Chatelain, Thierry Paquet

Handwritten text recognition: from isolated text lines to whole documents

D. Coquenet[1,2,3] C. Chatelain[1,4] T. Paquet[1,2]

[1] LITIS Laboratory - EA 4108, France
[2] Rouen University, France
[3] Normandie University, France
[4] INSA of Rouen, France

{denis.coquenet, clement.chatelain, thierry.paquet}@litislab.eu

Abstract

The handwriting recognition task is largely dominated by deep neural networks. However, it remains challenging for these advanced computer vision systems. Recently, the models have become more sophisticated, moving from line-level recognition to paragraph-level and even page-level recognition. In this paper, we will study those advances and the constraints that come with them, mainly focusing on two models we proposed: the Simple Predict & Align Network and the Vertical Attention Network. Both handle paragraph images, and we outperformed the state of the art on three datasets: RIMES, IAM and READ 2016.

Keywords

Handwritten text recognition, Seq2seq model, FCN, Attention

1 Introduction

The offline Handwritten Text Recognition (HTR) task consists in extracting the sequence of characters corresponding to the text present in an image. It is generally carried out in two steps: first, a segmentation model produces some bounding boxes in order to extract the different text regions (line or word) of a document ; then, a HTR model is applied on those text regions. The predictions are then concatenated following a predefined reading order based on the text region positions in the original image. These two-step models come with several drawbacks. As a matter of fact, training these models requires segmentation annotations at line level which are very costly to produce by hand. In addition, errors in the segmentation step induce additional recognition errors. Moreover, the fixed reading order can lead to errors with complex layouts such as document with two columns of text.
In order to get rid of these drawbacks, we recently proposed two end-to-end models to tackle the task of HTR at paragraph level, each with their pros and cons:

- the Simple Predict & Align Network (SPAN) [10] is a Fully Convolutional Network (FCN) predicting characters and interlines labels at once. This non-recurrent model implies an alignment between the prediction and the ground truth transcription at paragraph level, without needing line breaks in the annotation.
- the Vertical Attention Network (VAN) [9] follows a sequence-to-sequence architecture. It includes an attention module that enables to focus on a specific line of text inside a paragraph image. This way, it recurrently predicts the different text line transcriptions of the image, with the help of line break annotations.

The aim of this paper is to compare raw architecture performances, so we do not focus on additional module such as external language models for example. We do not enter into details for each module of the networks, the goal is to keep a high-level vision.
The paper is organized as follows. In Section 2, we provide an overview of models performing HTR on isolated text line images. Section 3 is dedicated to HTR models on single-column text documents. We focus on the SPAN and the VAN, comparing them with the state of the art. Finally, we present the emerging works, including ours, to handle whole documents with a complex layout and highlight the constraints related to this new step in Section 4. We draw conclusion in Section 5.

2 HTR at line level

HTR on line images are mainly performed after a prior segmentation stage, in a two-step process [5, 6, 18]. State-of-the-art results for HTR models at line level are reached by deep neural networks. A large variety of models has been proposed the last years but they mainly follow the same model: an encoder is used to extract features from the input image. It can be followed by an optional module to internally model the language; and finally, a prediction module outputs the sequence of characters. Many kinds of architecture have been studied: Multi-Dimensional Long Short Term Memory (MD-LSTM) [20], Convolutional Neural Network (CNN) [8], Fully Convolutional Network (FCN) [24, 7] or even combination of CNN and LSTM [21].

Traditionally, the vertical axis is collapsed before the prediction in order to reduce to a one-dimensional alignment problem between the prediction and the ground truth. As a matter of fact, the prediction sequence length is of variable size, depending on the input image width (it can also be fixed with reshaping as preprocessing); and the ground truth is also of variable size depending on the number of characters in the image. To solve this problem, the Connectionist Temporal Classification (CTC) [11] is generally used.

Another way to get around the problem is to use a sequence-to-sequence architecture. Indeed, instead of predicting one character or the CTC blank label for each frame of the one-dimensional sequence of features as for standard models, the characters are predicted step by step, recurrently focusing on part of the features through attention weights [1]. The process then stops when an end-of-prediction token is predicted [14].

In [9], we propose an FCN model for HTR at line level. It reaches competitive results on three datasets while providing many advantages: the module requires few parameters (1.7M), few GPU memory and it can handle input of variable sizes. This model is illustrated in Figure 1. The FCN encoder corresponds to a stack of convolutional blocks. Some strides are used in order to alleviate the memory consumption, reducing the shape to $\frac{H}{32} \times \frac{W}{8}$. Full description of the encoder can be found in [9]. As it can be noticed, an AdaptiveMaxPooling layer is used to collapse the vertical axis since the input images are not constrained in height nor width. Finally, a last convolutional layer predicts character and CTC blank label probabilities for each of the $\frac{W}{8}$ frames (N being the size of the character set).

Results are presented in Table 2, compared to state-of-the-art models in the same conditions *i. e.* without any external language model nor lexicon constraints. We used three datasets of reference for comparison purposes: RIMES [12], IAM [13] and READ 2016 [16]. They all correspond to handwriting digitized documents at a resolution of 300 dpi in French, English and Early Modern German respectively. They provide several levels of segmentation annotations. Table 1 describes the different splits used and the number of characters for each dataset.

Table 1: Datasets split in training, validation and test sets and associated number of characters in their alphabet

Dataset	Level	Training	Validation	Test	Charset size
RIMES	Line	10,532	801	778	100
	Paragraph	1,400	100	100	
IAM	Line	6,482	976	2,915	79
	Paragraph	747	116	336	
READ 2016	Line	8,349	1,040	1,138	89
	Paragraph	1,584	179	197	

3 HTR for single-column text document

Two-step processes have several drawbacks, mainly cumulative errors and they are annotation-consuming. Only few works have been proposed to tackle these problems, proposing end-to-end models to perform HTR at paragraph level, or more globally for documents with a single column of text. [2] proposed a model with an attention mechanism based on MD-LSTM layers. The model recurrently generates attention weights for each features row, implicitly segmenting the image into lines. Text lines are predicted through a recurrent process for a fixed number of iterations (big enough to handle the largest paragraph). The model first predicts the lines in a human-logical order (from top to bottom here) and then focuses on interlines to predict blank labels only. Line features are then concatenated to get a single one-dimensional sequence. [23] proposed a recurrence-free FCN model. An encoder generates the two-dimensional features. Then, some interpolation layers and convolutional layers are used to flatten the features. The aim is to unfold the image *i. e.* to concatenate all the lines to obtain a single large line. [2, 23] use the CTC loss to train their models. Since the alignment is carried out on the whole text, they do not need line breaks in the transcription.
We now present the models we proposed: the SPAN and the VAN. The first one is an FCN model free from attention and recurrence, while the second follows the seq2seq idea using an original vertical hybrid attention mechanism.

3.1 SPAN

The SPAN is an FCN model. It is designed to perform HTR on documents with a single column of texts while remaining as simple as HTR applied to text line images. By simple, we mean that the whole model is trained with the standard CTC, without recurrence. The idea is to use the CTC blank label to predict interlines, in addition to its usual use. An overview of the SPAN is depicted in Figure 2. As one can see, the SPAN is similar to the HTR model for text line images (Figure 1). Indeed, it uses the same encoder architecture (but twice wider); however, it differs in two major points:

- We do not collapse the vertical axis through AdaptiveMaxPooling, this would lead to preserve only one character per column of features *i. e.* only keep one text line prediction.
- Instead, we reshape the two-dimensional features to a one-dimensional sequence through row concatenation, preserving the whole paragraph prediction.

The reshaping operation is detailed in Figure 3: rows of features are concatenated from top to bottom to obtain a single large sequence of prediction representing the whole paragraph. This enables to get back to a one-dimensional alignment problem that is handled with the standard CTC.

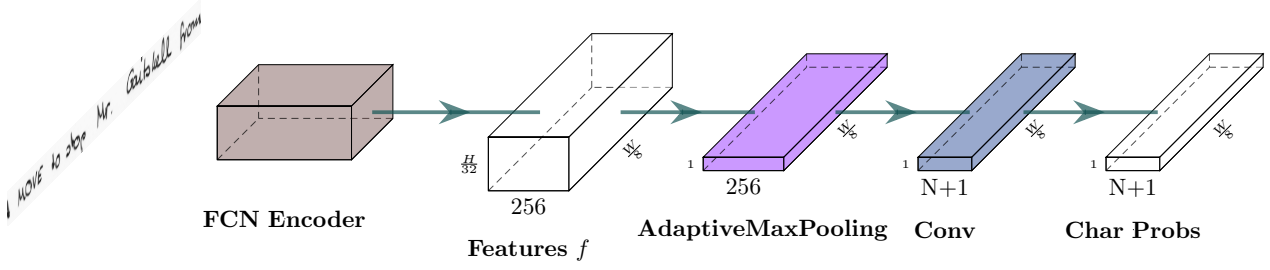


Figure 1: Overview of an example of HTR model at line level (from [9])

Table 2: Comparison with the state of the art on the test sets of IAM, RIMES and READ 2016.

Architecture	Attention	IAM		RIMES		READ 2016		# Param.
		CER (%)	WER (%)	CER (%)	WER (%)	CER (%)	WER (%)	
HTR applied on line images								
[15] CNN+BLSTM	✗	5.8	18.4	2.3	9.6			9.6 M
[24] FCN	✗	4.9						> 10 M
[14] Seq2seq (CNN+BLSTM)	character	4.87				4.66		
[16] BYU - CNN+RNN	✗					5.1	21.1	
[9] Ours - FCN	✗	4.95	16.24	3.19	10.25	4.28	19.71	1.7 M
Two-step approaches								
[5] RPN+CNN+BLSTM	✗	15.6						
[6] RPN+CNN+BLSTM	✗	8.5						
[22] RPN+CNN+BLSTM	✗	6.4	23.2	2.1	9.3			
End-to-end approaches for one-column text documents								
[2] CNN+MDLSTM	line	7.9	24.6	2.9	12.6			
[23] FCN	✗	4.7						16.4 M
[10] Ours (SPAN) - FCN	✗	5.45	19.83	4.17	15.61	6.20	25.69	19.2 M
[9] Ours (VAN) - FCN+LSTM	line	**4.32**	**16.24**	**1.90**	**8.83**	**3.63**	**16.75**	2.7 M
End-to-end approaches for documents with complex layout								
[3] CNN+MDLSTM	character	16.2						
[17] CNN+Transformer	character	6.7						27 M
Ours (CAN) - FCN+Transformer	character	6.01	20.98	5.13	15.75	7.31	24.14	3.4 M

The illustrated order enables to handle languages such as french or english, but it could be reversed to carry out HTR with arabic text for example.

An example of prediction is shown in Figure 4. Since the SPAN is recurrence-free, all the characters are predicted at once. One can note that, due to the reshaping operation, the character predictions of text lines that stretch over multiple feature rows are aligned vertically. For a same text line, the predictions can only goes from top to bottom, the opposite would lead to mixtures of sub-strings. However, this constraint has the advantage of enabling the model to handle slightly downward inclined lines.

3.2 VAN

The VAN follows a seq2seq architecture. An overview of the model is depicted in Figure 5. It is made up of the same encoder as depicted in Figure 1, an original vertical hybrid attention module and a decoder. The process flow is as follows:

- The encoder aims at extracting features from the input image: it generates a two-dimensional representation of the characters, preserving the location information
- The hybrid attention module has two functions: to generate the current line representation and to detect the end of the transcription. On one hand, it recurrently computes a weighted mask over the vertical axis *i. e.* a probability distribution with one weight per feature row. Those weights determine how much the feature row should be taken into account to predict the current text line. Thus, some line features are generated as the weighted sum among the feature rows, leading to a one-dimensional sequence of features representing the current text line. On the other

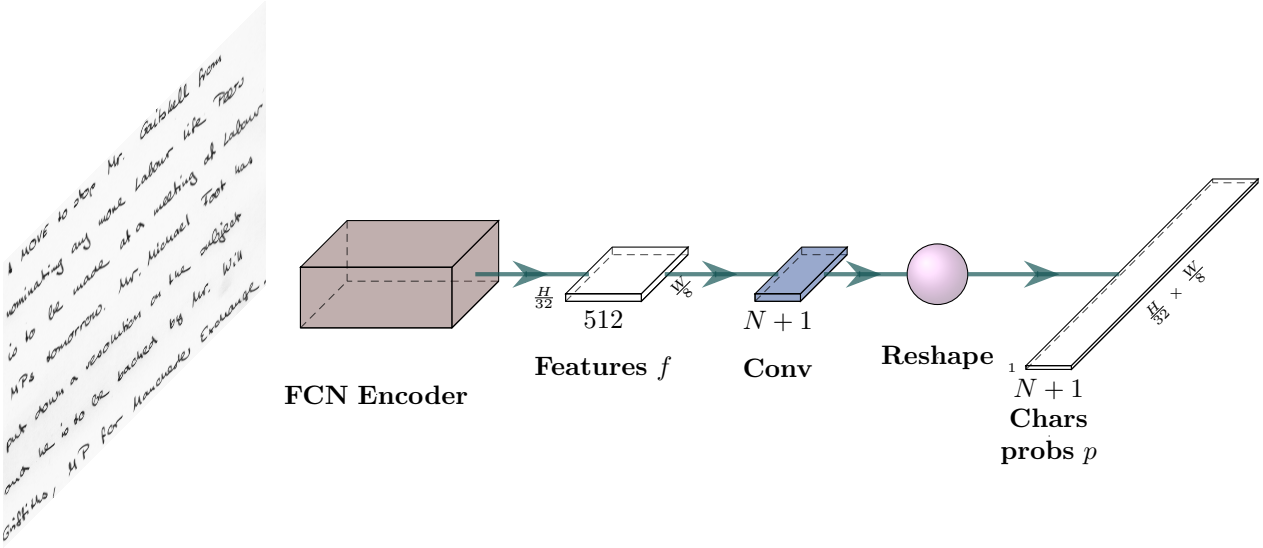


Figure 2: SPAN overview

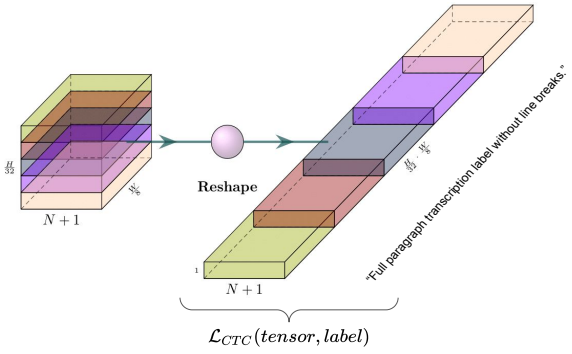


Figure 3: Focus on the SPAN reshape operation

hand, it also generates some probabilities to stop or to continue the process, determining whether or not the end of the transcription has been reached.

- The decoder objective is to predict the characters and CTC blank label probabilities from the line features, as in standard HTR at line level, after the vertical collapse.

- A *whitespace* token is added between the line transcriptions to get the final document transcription.

The VAN is trained using a composite loss made up of the CTC loss for the line-level alignment between each line predicted transcription and ground truth, and the cross-entropy loss for the end-of-transcription prediction. The VAN uses the line breaks in the transcription to perform this line-level alignment and to generate ground truth of the second loss: the number of line breaks indicates the number of lines in the document.

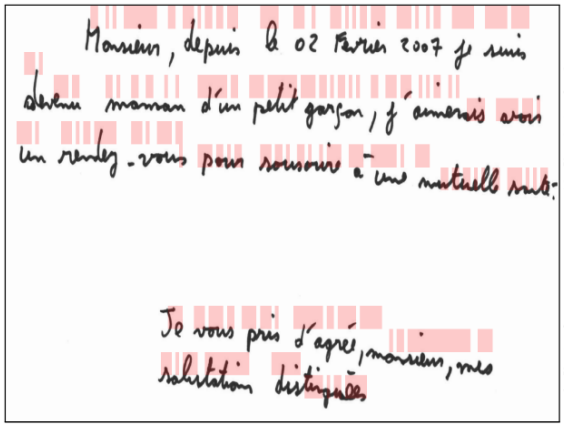
To better understand the working process, two iterations of the prediction of a paragraph from the RIMES validation set is depicted in Figure 4. Attention weights for each iteration are represented by the red color intensity and projected to the input image, explaining the width of the red areas. One can clearly see that only one weight is computed per feature row, that is why the red color intensity is the same along the horizontal axis. The second iteration is a good example of the power of this model since its flexibility in the weights computation enables it to spread the attention across multiple feature rows to handle inclined lines, regardless of their direction of inclination.

3.3 Discussion

Results for the SPAN and the VAN are presented in Table 2. It has to be noted that, contrary to [9, 2], the SPAN encoder has to generate characters representation, and to align them vertically as well. This additional task could explain the difference in performance, notably compared to [9] which contains the same encoder. [9, 10, 2] use pretraining on line-level images to reduce the convergence time and to improve the performances. In [23], the interpolation height and width are chosen specifically to each dataset, which could explain that this model converges quickly without the need for any pretraining.

The VAN reaches state-of-the-art results on the three datasets, even compared with HTR models applied on line images (ground truth line segmentation).

As mentioned previously, these models are limited to single-column text documents, limiting their usage. They can also be used in a two-step process with a prior paragraph segmentation step. The aim is now to go a step further *i. e.* to handle documents with a complex layout.



Monsieur, depuis la 02 Ferries 2007 je suis

de
enu maman d'un petit garçon, j'aimer
ais avoir
un rendez-vous
s pour sousoire à une m
ntuelle sont

Je vous pris d'agrée, m
ansieurs, mes
salutation dis
tinguées

Monsieur, depuis l**a** 02 Fer**r**ie**s** 2007 je suis de**v**enu maman d'un petit garçon, j'aimerais avoir un rendez-vous**s** pour sous**o**ire à une m**n**tuelle s**o**nt***é***. Je vous pris d'agrée, m**a**nsieurs, mes salutation**s** distinguées

Figure 4: SPAN predictions visualization for a RIMES test example. Left: predictions are reshaped to their original 2D shape and projected on the input image. Transparency indicates a blank label prediction and character predictions are shown in red. Right: row by row text prediction. Bottom: full text prediction where missing letters are shown in italic and errors are shown in bold.

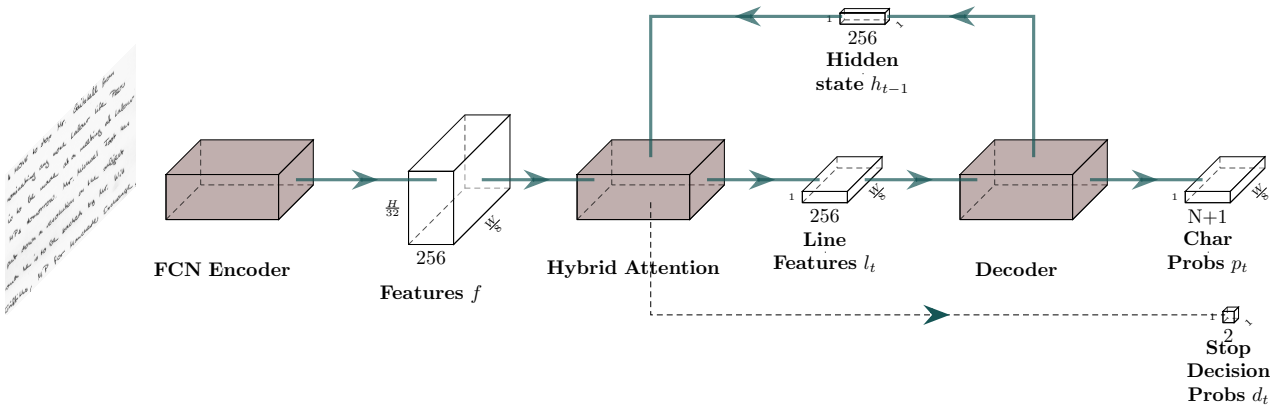
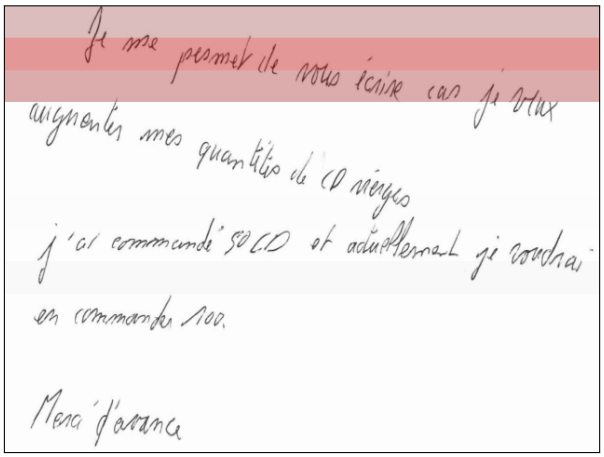


Figure 5: VAN overview

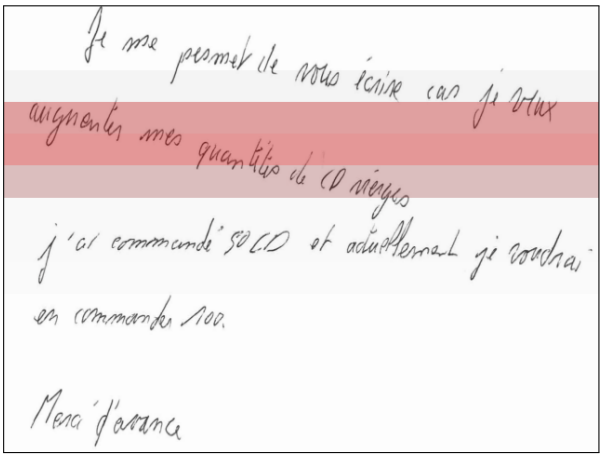
4 Towards HTR for documents with complex layouts

Two-step approaches could handle documents with complex layouts but they should be homogeneous within the same dataset. Indeed, given that the reading order is not as simple as from top to bottom and from left to right, handcrafted rules should be defined to simulate a human-like reading order. But this is not maintainable for heterogeneous and varied layouts. This way, the reading order should be learned. Moreover, we cannot make the assumption that one row of features corresponds to a single line of text anymore. To tackle these new problems, [3, 17] proposed end-to-end models that output the document transcription character by character. While the model from [3] includes a hybrid attention mechanism based on MD-LSTM layers to recurrently focus on the different characters, the model proposed in [17] uses a transformer architecture [19], only using the already predicted tokens to compute its attention weights. The transformer architecture enables the model to be trained without recurrence using teacher forcing, decreasing the computation needs at training time.

Our current work is inspired by this transformer architecture; the model is called CAN for Character Attention Network and is presented in Figure 7. As one can see, a FCN encoder generates features which are then enhanced with a two-dimensional positional embedding. This Encoder is different from the ones used in the previously described models. It notably includes more instance normalization for numerical stability. The transformer decoder is a stack of multi-head attention layers (as defined in [19] in which the queries are the previously predicted token embedding, enhanced with a one-dimensional position embedding and



Je me permet de vous écrire ca**s** je v**M**ux



augmente**s** mes quantités de CD vierges

Figure 6: VAN attention weights visualization on a sample of the RIMES validation set. First and second iterations are represented respectively on the left and on the right. Transcription predictions are given for each line and errors are shown in bold.

the keys and the values are the features. Through those multi-head attention layers, the transformer decoder generates a weighted mask over the features, whose weighted sum represents the current character. Character and end-of-transcription token probabilities are computed from this representation.

Preliminary results are shown in Table 2. As one can see, there is still room for improvement to bridge the gap with the state-of-the-art model. As a matter of fact, working with an attention mechanism at character level increases by far the number of iterations compared to line-level attention. Concerning the datasets used in this work, the maximum number of lines per paragraph is 26 and the maximum number of characters is 1,195. This leads to bigger training and prediction times. This also prevents from back propagating the whole process at once when using hybrid attention. Moreover, the input images are bigger and thus the number of potential locations to focus on is increased; the number of repeated subsequences also grows, bringing confusion to the model. Another point, specific to the transformer architecture, is the growing number of computations through the iterations, related to the use of the already predicted tokens. To alleviate this point, we only use the last 50 predicted token as in [17]; that seems sufficient to model the language.

The prediction process of the CAN is depicted in Figure 8. As one can note, this process is rather heavy since we must iterate for each character. As for the VAN prediction visualization, the attention weights for each iteration are represented by the red intensity and projected to the input image. One can note that the model has learned the human reading order, placing its attention on the top left character, avoiding the first indent. It then moves from character to character.

4.1 Discussion

In this paper, results are given for paragraph-level images on three datasets of reference for comparison purposes. One can note the lack of realistic datasets for HTR on documents with a complex layout. To our knowledge, only the MAURDOR dataset [4] meets the requirements. Moreover, while processing complex documents, one face the problem of the reading order. In the case of maps, schema or any other complex structure for example, there is no single humanly-logical reading order: there is a multitude of possible valid reading orders. This raises the question of the training process, and more specifically of the loss and metrics that are currently used. They should be more flexible to take into account this issue.

Another important point is the actual use of segmentation labels. Indeed, these models do not strictly require line-level segmentation labels to be trained and to carry out prediction. But in fact, all the proposed models ([3, 17], including ours, require synthetic data. Those new samples are generated using the line-level segmentation annotation, combining them randomly. Without them, the models do not converge. This problem could be alleviate with the recent works on contrastive self-supervised learning whose aim is to pretrain the model without annotated data.

5 Conclusion

In this paper, we have highlighted the recent advances made for the task of Handwritten Text Recognition. We have focused on two models we proposed, the SPAN and the VAN which outperforms the state-of-the-art on three datasets: RIMES, IAM and READ 2016. We also present our current work in progress to handle whole documents with complex layout and we brought to light the constraints and limitations we are currently facing.

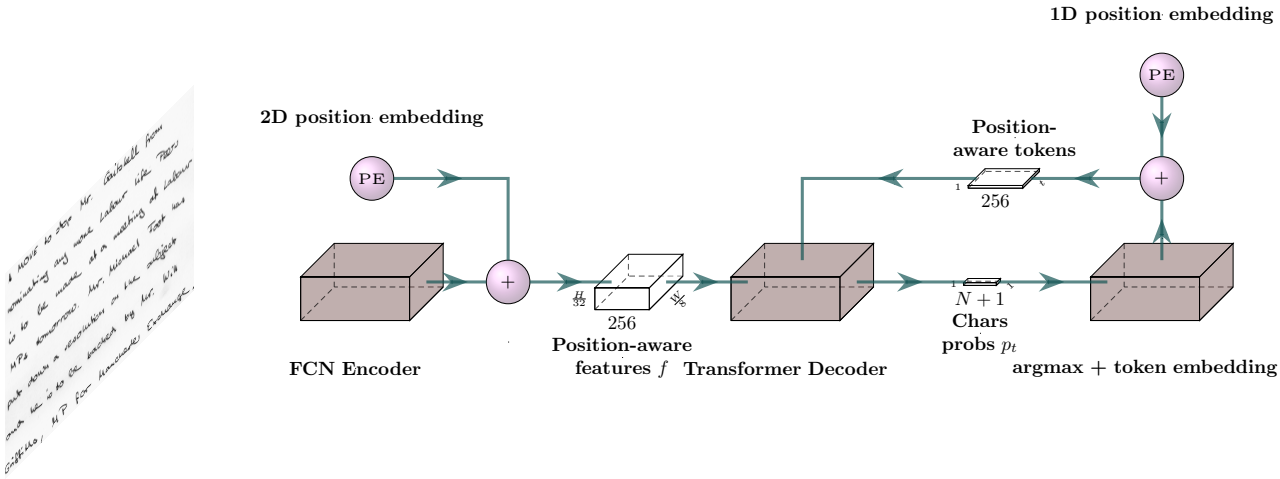
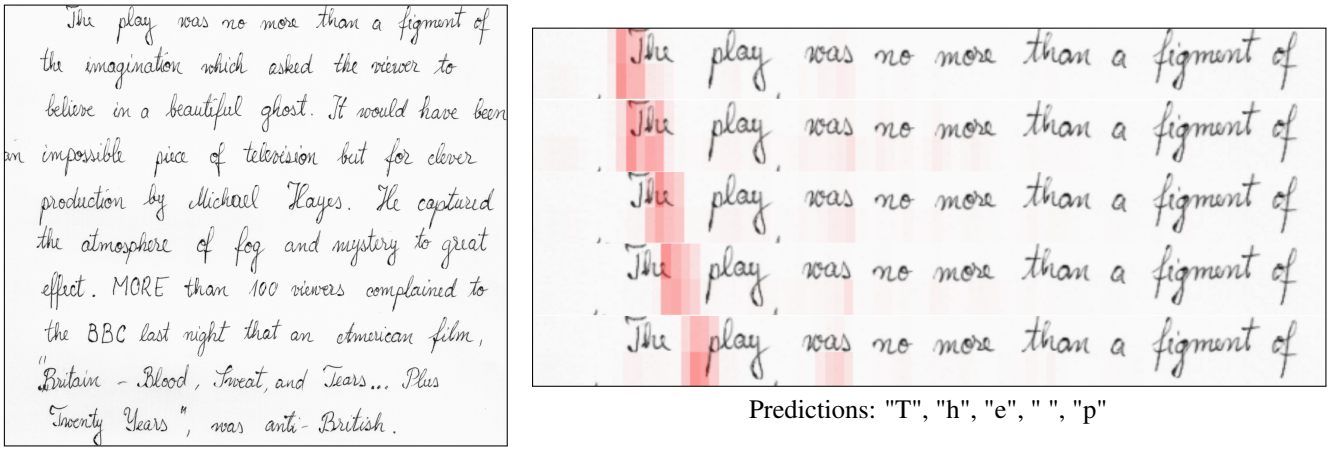


Figure 7: CAN overview



Predictions: "T", "h", "e", " ", "p"

Figure 8: CAN attention weights visualization on a sample of the IAM validation set. Left: original input image. Right: zoom on the first line for the first five iterations.

Acknowledgments

The present work was performed using computing resources of CRIANN (Normandy, France) and HPC resources from GENCI-IDRIS (Grant 2020-AD011012155). This work was financially supported by the French Defense Innovation Agency and by the Normandy region.

References

[1] Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR (2015)

[2] Bluche, T.: Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. In: Lee, D.D., Sugiyama, M., von Luxburg, U., Guyon, I., Garnett, R. (eds.) Annual Conference on Neural Information Processing Systems. pp. 838–846 (2016)

[3] Bluche, T., Louradour, J., Messina, R.O.: Scan, attend and read: End-to-end handwritten paragraph recognition with MDLSTM attention. In: International Conference on Document Analysis and Recognition. pp. 1050–1055 (2017)

[4] Brunessaux, S., Giroux, P., Grilheres, B., Manta, M., Bodin, M., Choukri, K., Galibert, O., Kahn, J.: The maurdor project: Improving automatic processing of digital documents. pp. 349–354 (04 2014)

[5] Carbonell, M., Mas, J., Villegas, M., Fornés, A., Lladós, J.: End-to-end handwritten text detection and transcription in full pages. In: International Work-

shop on Machine Learning, WML@ICDAR. pp. 29–34 (2019)

[6] Chung, J., Delteil, T.: A computationally efficient pipeline approach to full page offline handwritten text recognition. In: Workshop on Machine Learning, WML@ICDAR. pp. 35–40 (2019)

[7] Coquenet, D., Chatelain, C., Paquet, T.: Recurrence-free unconstrained handwritten text recognition using gated fully convolutional network. In: 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 19–24 (2020)

[8] Coquenet, D., Soullard, Y., Chatelain, C., Paquet, T.: Have convolutions already made recurrence obsolete for unconstrained handwritten text recognition ? In: Machine Learning workshop@ICDAR. pp. 65–70 (2019)

[9] Coquenet, D., Chatelain, C., Paquet, T.: End-to-end handwritten paragraph text recognition using a vertical attention network. CoRR **abs/2012.03868** (2020)

[10] Coquenet, D., Chatelain, C., Paquet, T.: SPAN: a simple predict & align network for handwritten paragraph recognition. CoRR **abs/2102.08742** (2021)

[11] Graves, A., Fernández, S., Gomez, F.J., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: International Conference on Machine Learning(ICML). vol. 148, pp. 369–376 (2006)

[12] Grosicki, E., Abed, H.E.: ICDAR 2011 - french handwriting recognition competition. In: International Conference on Document Analysis and Recognition,. pp. 1459–1463 (2011)

[13] Marti, U., Bunke, H.: The iam-database: an english sentence database for offline handwriting recognition. Int. J. Document Anal. Recognit. **5**(1), 39–46 (2002)

[14] Michael, J., Labahn, R., Grüning, T., Zöllner, J.: Evaluating sequence-to-sequence models for handwritten text recognition. In: International Conference on Document Analysis and Recognition. pp. 1286–1293 (2019)

[15] Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition? In: International Conference on Document Analysis and Recognition. pp. 67–72 (2017)

[16] Sánchez, J., Romero, V., Toselli, A.H., Vidal, E.: ICFHR2016 competition on handwritten text recognition on the READ dataset. In: International Conference on Frontiers in Handwriting Recognition. pp. 630–635 (2016)

[17] Singh, S.S., Karayev, S.: Full page handwriting recognition via image to sequence extraction. CoRR **abs/2103.06450** (2021)

[18] Tensmeyer, C., Wigington, C.: Training full-page handwritten text recognition models without annotated line breaks. In: International Conference on Document Analysis and Recognition. pp. 1–8 (2019)

[19] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Annual Conference on Neural Information Processing Systems. pp. 5998–6008 (2017)

[20] Voigtlaender, P., Doetsch, P., Ney, H.: Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In: International Conference on Frontiers in Handwriting Recognition, ICFHR. pp. 228–233 (2016)

[21] Wigington, C., Stewart, S., Davis, B.L., Barrett, B., Price, B.L., Cohen, S.: Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network. In: International Conference on Document Analysis and Recognition, ICDAR. pp. 639–645 (2017)

[22] Wigington, C., Tensmeyer, C., Davis, B.L., Barrett, W.A., Price, B.L., Cohen, S.: Start, follow, read: End-to-end full-page handwriting recognition. In: European Conference on Computer Vision. Lecture Notes in Computer Science, vol. 11210, pp. 372–388 (2018)

[23] Yousef, M., Bishop, T.E.: Origaminet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. In: Conference on Computer Vision and Pattern Recognition. pp. 14698–14707 (2020)

[24] Yousef, M., Hussain, K.F., Mohammed, U.S.: Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. Pattern Recognit. **108**, 107482 (2020)