The A2iA Multi-lingual Text Recognition System at the second Maurdor Evaluation

Bastien Moysset*, Théodore Bluche*†, Maxime Knibbe*, Mohamed Faouzi Benzeghiba*,
Ronaldo Messina*, Jérôme Louradour* and Christopher Kermorvant*
*A2iA, 39 rue de la Bienfaisance, 75008 - Paris - France
†LIMSI CNRS, Spoken Language Processing Group, Orsay - France

***Abstract*—This paper describes the system submitted by A2iA to the second Maurdor evaluation for multi-lingual text recognition. A system based on recurrent neural networks and weighted finite state transducers was used both for printed and handwritten recognition, in French, English and Arabic. To cope with the difficulty of the documents, multiple text line segmentations were considered. An automatic procedure was used to prepare annotated text lines needed for the training of the neural network. Language models were used to decode sequences of characters or words for French and English and also sequences of part-of-arabic words (PAWs) in case of Arabic. This system scored first at the second Maurdor evaluation for both printed and handwritten text recognition in French, English and Arabic.**

I. Introduction

Following the trend existing in other research communities, the handwriting recognition community has started to organize international evaluations of the technology ten years ago, and the number of evaluations keeps increasing. In 2005, the first evaluations concerned the recognition of isolated words [1] and, in 2011, the systems have reached a plateau around 5% to 7% error rate [2]. The most recent evaluations are now oriented toward large vocabulary text line recognition [3][4], in which the best systems are between 10% and 20% error rate. A step further has been taken with the Maurdor evaluation campaign [5], where the complete process of document analysis and recognition is evaluated on difficult and realistic documents.

In this paper, we describe the system submitted by A2iA to the Maurdor 2013 evaluation campaign for handwritten and printed text recognition.

II. The Maurdor challenge

The goal of the Maurdor evaluation campaign [6] was to evaluate the performance of automatic document processing systems on a large variety of complex multi-lingual documents, as show on Figure 1. The complete document processing chain was decomposed into autonomous modules: document layout analysis, write type identification, language identification, text recognition, logical organization and information extraction. Each module was evaluated in isolation with the ground-truth value of the data from the previous module in the sequence. The text recognition modules were evaluated using as input the co-ordinates of the text zone, the write type of the text and the language. We describe in this paper our system for this task and the result of the evaluation. The Maurdor database was provided to train and evaluate the systems. Official splits of the database with the number of text zones are presented on Table I.

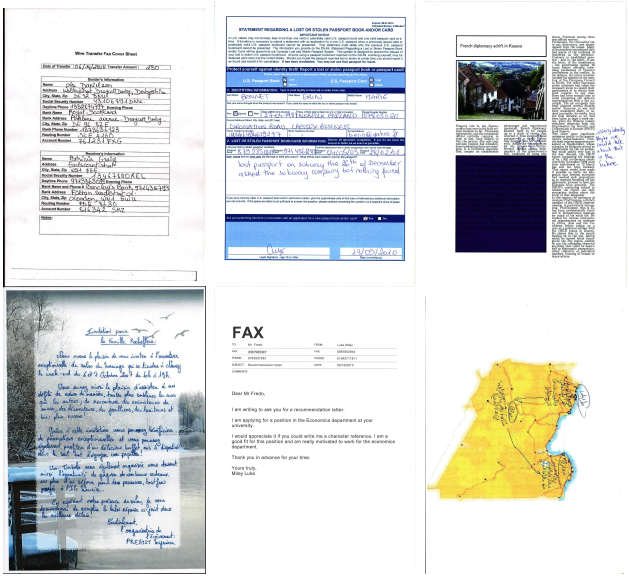


Fig. 1: Samples of documents from the Maurdor database.

TABLE I: The Maurdor database : official splits in Train, Dev and Test sets with the number of text zones for each writing types and languages

Set	Pages	Zones					
		Printed zones			Handwritten zones		
		French	English	Arabic	French	English	Arabic
Train2	6 592	141 683					
			105 002			36 681	
		57 821	25 773	21 263	18 417	8 530	9 729
Dev2/Test1	1 110	25 663					
			19 205			6 458	
		9 908	5 124	4 122	2 857	1 765	1 835
Test2	1 072	25 180					
			18 907			6 273	
		11 519	4 131	3 210	3 241	1 450	1 582
Total	8 774	192 526					
			143 114			49 412	
		79 248	35 028	28 595	24 515	11 745	13 146

III. Text line detection

The input of the system was the image of the complete document with the coordinates of the text zone, its writing type and its language. Since the zones given in annotation are at paragraph level, a line segmentation algorithm was required. To apply the line segmentation algorithm, color and binary images were first converted to grayscale and rescaled to a resolution

of 300 dots per inch (dpi).

Then, two line detection algorithms were used to get the boxes corresponding to text lines. To improve the efficiency of these algorithms, pre-processing of the paragraph images were performed. However, the recognizer used the images obtained from the unprocessed 300 dpi grayscale images with the detected text line boxes, without any denoising.

A. Algorithms

Two line detection algorithms were used in this system. The first algorithm was based on grouping connected components. Connected components were extracted from the binarized image after denoising, deskewing and deslanting. Based on their skeleton and statistical heuristics, the connected components were grouped into words and text lines.

The second algorithm was based on projection profile. In this algorithm, the pre-processing step included binarization, deskewing, removal of background lines, denoising with a Gaussian filtering, morphological closure (to fill small holes between components) and background inversion if needed. A post processing step was also performed to merge lines lying at the same level.

B. Line segmentation hypotheses

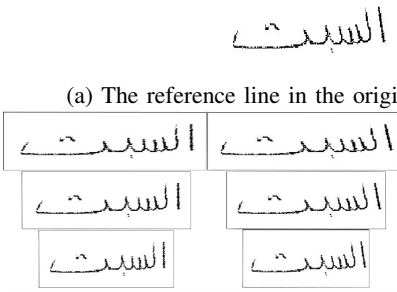
The first and the second line detection algorithms were used to process handwritten and printed paragraphs, respectively. To improve the text line segmentation, line segmentation hypotheses were introduced. Pre-processing was performed to create several different images of the same paragraph. For the handwritten paragraphs, the deskewed image from the paragraph was added. Horizontally stretched and shrinked images were also added as well as the small unsegmented paragraphs in case there was just one line in the paragraph. For the printed paragraphs, the segmentation from both line detection algorithms were considered. Images with normalized connected components mean height were also added. This technique resulted in 4 to 7 line segmentation hypotheses per paragraph. A recognition step was performed on all these line hypotheses and the best segmentation alternative was chosen based on the recognition score and some heuristics. These heuristics were introduced to encourage the system to keep a high number of lines and to choose the lines as wide as possible.

IV. Optical model

The optical model was in charge of computing sequences of character posterior probabilities from variable-sized images. Each vector contained character posterior probabilities. The length of the sequences depended on both the width of the image and the width of the sliding window of the optical model.

A. Training data preparation

The annotation of the training data was given at paragraph level. But the training of the neural network required text line images with their corresponding transcriptions. We developed an automatic system [7] to align the line images with the annotation.



(a) The reference line in the original training set

(b) Nine versions of the reference line with different transformations: slanting, shrinking and expansion.

Fig. 2: Increasing the training set with image transformations.

a) Text line image annotation: The alignment process was performed on grayscale images normalized to 300 dpi. The presence of line breaks in the annotation helped the process. First, a text line detection was performed on the images of paragraphs. On each text line, a constrained recognition was performed with a system trained on text zones containing only one line of text. This recognition was constrained so that the system could either recognize one of the line present in the ground-truth text, or part of a line (which could for example correspond to a line split in several parts by the line segmenter), or nothing (unmatched line). The constraints were encoded using finite state transducers as explained in [7].

Line images in which nothing or just a part of a text line was recognized were considered as unreliable and discarded. If several lines share the same line text, the one with the highest recognition score was kept, the other were discarded. The remaining lines were used to train the recognition system which in turn was used to perform a new constrained alignment. Since this system was trained on more data, the alignments were better and more annotated text line images were produced. This alignment cycle was performed twice and the improvement on the recognition system are described in the Results section.

b) Noising of training images: For handwriting recognition, some transformations are applied on the images in the training data. The goal of this technique is to introduce some variability in the training data to enhance the generalization capabilities of the neural network [8]. As illustrated in Figure 2, each image in the training data was first slanted in both directions, resulting in 3 different images including the original one. Each of these 3 images was then shrinked and expanded in the horizontal direction, resulting in a total of 9 images.

B. Multi-Directional LSTM Recurrent neural networks

In our text recognition system, the optical model was a Recurrent Neural Network (RNN) working directly on the pixel values. The two-dimensional recurrence was applied on neighboring pixels using Long-Short Term Memory (LSTM) cells [9], which were carefully designed to model both local and scattered dependencies within the input images. Besides, 4 LSTM layers were applied in parallel, one for each possible scanning direction, and their outputs were combined.

We trained a specific RNN for each one of the six tasks (each one of the three languages, typed or handwritten), using

TABLE II: Number of hidden (and output) units per layer, used in the Recurrent Neural Networks. The last line indicates the total number of free parameters to be optimized.

	Handwritten			Typed		
	English	French	Arabic	English	French	Arabic
Layers:						
(1) LSTM	4	4	2	4	4	4
(2) Convolution	12	12	6	12	12	12
(3) LSTM	20	20	10	20	20	24
(4) Convolution	32	32	20	32	32	60
(5) LSTM	100	100	200	100	100	150
(6) Linear	92	110	164	108	136	179
# free parameters	547 148	554 366	1 827 908	553 564	564 792	1 268 515

Connectionnist Temporal Classification [10], since an explicit character segmentation of the data was not available. The entire neural network architecture was similar to the one initially proposed by [11] and gave state-of-the-art performances for Latin and Arabic text recognition [12], [13]. More details about how to train and decode with Multi-Directional LSTM Recurrent Neural Networks can be found in these previous papers [11], [13]. Two main modifications to the published architectures were made. First we adapted the sub-sampling filter sizes to fit input images at 300 dpi: the input image was first divided into blocks of size 2×2 (tiling), and the filter sizes of the two sub-sampling layers that came respectively after the two first LSTM layers were 2×4 (convolution without overlapping). Thus the RNN predicted posterior probabilities on non-overlapping window with a 8 pixels width ($2 \times 2 \times 2 = 8$). Second, we tuned the hidden layer sizes (number of intermediate activations) to optimize the performance on the validation dataset. The number of hidden units for each layer, depending on the language to recognize, is given in Table II.

An important improvement to the training procedure was also achieved by using dropout [14], a powerful regularization technique that consists in randomly sparsifying the intermediate activations. The details on how to apply dropout on RNNs are given in [15].

Besides, we followed the principle of "Curriculum Learning" [16] to optimize the RNN by stochastic gradient descent. In fact, previous works showed that training a neural network first on "simple" labeled examples before switching to the full dataset of interest (which includes noisy and difficult examples) can lead not only to faster convergence, but also to better generalization performance [17]. For this reason, and because the Maurdor's data were especially difficult , we did not run gradient descent directly on randomly initialized RNN models, but on models that were already (pre)trained on some public datasets that are "clean" to a certain extent (constant background color, unique digitization process): The Rimes dataset [18] for all the Latin scripts (French and English, typed and handwritten), the OpenHaRT 2013 dataset [4] for handwritten Arabic script, the APTI dataset [19] for typed Arabic script.

V. Language models

To create language models, the training data was normalized and tokenized. For each language and each writing type, we first gathered all training data published for the Maurdor evaluation and decided on a character set to be used for recognition. Characters that had a small number of occurrences were not modeled; they were either replaced by an equivalent symbol (e.g. for different round bullet marks we just kept one of the symbols), or removed from the data (e.g. a telephone symbol, copyright mark). A particular characteristic of the Arabic language is that the same character (or letter) may have different presentation (written) forms depending on its position in the word (i.e. isolated, initial, middle and final forms). We modeled this presentation form for the Arabic systems. The conversion of a word to its presentation forms was performed with the open-source *fribidi* [20] algorithm.

The lines that contained characters not retained for modeling were ignored and did not contribute to the language model. Arabic (respectively Latin) characters that were present in the Latin (respectively Arabic) data were simply ignored. Some ligatures (such as "ff", "fl", "œ") are replaced with the individual characters to simplify the modeling procedure.

After clean-up and normalization of the characters, we tokenized the words using some of the rules of the evaluation tool and other rules specific to our systems. Space characters in the annotations were replaced with an arbitrary symbol, present in the optical model to signify the inter-word space. This allowed us to split digit strings into their constituent digits, reducing the size of the vocabulary of the language model (LM) and also simplifying the recognition of codes, dates and numbers. As expected, the bigram counts showed that the digit "1" was the most frequent at the beginning of a string (34%) followed by "2" (23%), much similar to what Benford's law predicts, but the data is not large enough to closely follow that distribution. Inter-word space and punctuation symbols (comma, stop, dash, quotes, , etc.) were all treated as regular "words" in the LM.

We treated capitalized variants of the "same" word as different entities, so there were different n-grams in the LM. This was mainly because some words were quite frequent at the beginning of a sentence where they appear capitalized. It could be interesting to assess the effect on performance if the words were treated as a single entity, and different parallel paths in the grammar accounted for the different capitalizations.

The Arabic data was further processed to decompose rare words into their Part-of-Arabic-Words (PAW). The frequent words (i.e. words that have appeared more than once) were kept as they are without PAW decomposition. However, rare words (i.e. words that have appeared only once) were decomposed into their PAWs. These decomposed words were then replaced -in the training data- by their PAWs. Unlike word separation (i.e. use of an arbitrary symbol as an inter-word space), a standard space was used as an inter-PAWs space. This made the concatenation of PAWs into word much easier during the recognition phase. It was done by simply removing the standard space. The resulting vocabulary was an hybrid vocabulary that contained both regular words and PAWs. The effectiveness of this decomposition procedure was demonstrated during the development of the Arabic systems, and confirmed by post-evaluation experiments.

Table III reports the amount of data from handwritten and printed sources for the three languages and the two writing types. The values between parentheses indicates the counters

TABLE III: Statistics on the textual data available for training, for the two writing types and the three languages.

Language		Handwritten	Typed
English	#Words	98520 (960467)	857873 (956418)
	#Vocabulary	7564 (28805)	26757 (30059)
	#Hapax	4508 (13144)	12085 (14132)
	#Chars	210180 (2506643)	2296309 (2506535)
	$\|charset\|$	91	107
French	#Words	291069 (1739926)	1441428 (1732628)
	#Vocabulary	17109 (43520)	36867 (44746)
	#Hapax	9944 (21621)	17713 (22359)
	#Chars	698599 (4356676)	3658529 (4357436)
	$\|charset\|$	109	135
Arabic	#Words	94528 (458952)	365444 (459986)
	#Vocabulary	9579 (21594)	18392 (21621)
	#Hapax	4478 (6149)	5682 (6164)
	#Chars	352406 (1711200)	1360821 (1713256)
	$\|charset\|$	163	178

after combining the two sources (i.e. handwritten and printed data sets) under a given character-set. In all cases the inter-word space was counted as a "word" and figures are after splitting the digit strings.

It is worth reminding here that for a given language and for each writing type, the characters/forms set was selected using the writing type specific dataset only. However, the language model was generated from both handwritten and printed data sets. So, in Table III, the statistics between parentheses have to be considered.

Table III presents a rate of hapaxes (words occurring only once in the data) of about 50%, which is quite normal for a reduced database. Arabic had larger character (forms) set than French and English due to the use of presentation (written) forms. French had a larger character-set than English due to accented characters. Printed models had extra characters not present (or not modeled) in the handwritten part of the corpus.

Trigram LMs were generated for each language and writing type using Witten-Bell smoothing [21]. For Latin (French and English) languages and printed type systems, we investigated the use a hybrid word/character model that can recognize out-of-vocabulary (OOV) words. This model was more efficient when a word-level LM demonstrated a low word error rate (WER). However, experimental results showed a small degradation in the performance.

To give an idea of the complexity of the test set (i.e. *Test2*), we re-estimated word language models with the tokenized data and without inter-word space symbol and did the same for the test corpus to compute the perplexity and the hit-ratio of the n-grams (the number of times the n-gram is present in the LM). We preferred to use these LMs than those used in the submitted system, because the perplexity value would not make sense in the case of modeling with the inter-word spaces. Table IV reports the perplexity, the out-of-vocabulary words percentage and the hit-ratio for each system.

The ratio of OOV words was quite low (to some extent due to splitting the digit strings) and the language models also presented relatively low values for the perplexity. It can be expected that most of the difficulty in recognizing the test data comes from the variability in the images.

TABLE IV: Perplexity (PPL), out-of-vocabulary rate (%OOV) and n-gram hit-ratio estimated on the test data (*Test2*).

Language	Type	PPL	%OOV	%Hit-Ratio		
				3-gram	2-gram	1-gram
English	PRN	111	3.9	37.1	41.8	21.1
	HWR	66	4.5	49.7	35.1	15.2
French	PRN	48	3.6	54.7	31.9	13.4
	HWR	73	3.9	48.6	36.6	14.7
Arabic	PRN	146	11.2	31.8	33.9	34.3
	HWR	134	8.4	29.5	44.9	25.6

VI. Decoding

The decoding was performed using the Kaldi toolkit [22]. The decoder searched a graph based on Weighted Finite-State Transducers (WFST), composed of the main system components.

The RNN produced a sequence of characters/forms predictions, that could be constrained with a lexicon and a language model to recognize the most likely sequences of valid words. System components could be represented as WFSTs, and then composed to create the decoding graph, as explained in [23].

We adopted an hybrid Hidden Markov Model (HMM) - RNN approach. Each RNN output class (character, or form, plus white-space and blank) was represented by a one-state HMM, with a self-loop and an outgoing transitions. HMM state emission likelihoods were estimated by dividing RNN class (i.e HMM state/character) posteriors $p(s|x)$ (where s is the state/character and x is the observation) by the class priors $p(s)^{\kappa}$ scaled by a tunable factor κ. Class priors were estimated using the training dataset. The HMMs were transformed into a WFST H.

The lexicon FST L transformed a sequence of characters and blank symbols into words. We took into account the blank symbol (the "no-character" prediction) in the structure of the WFST. In the decomposition of the word, we allowed to read an optional blank symbol between two characters. However, when a character was doubled in a word, the blank transition became mandatory.

The language model was created with the SRILM toolkit [24] and transformed into an WFST (G) with Kaldi.

Once built and composed, the final FST HLG was the decoding graph, taking as inputs the character - plus blank - predictions provided by the optical model and outputting the recognized sequence of words.

VII. Results

The official results for the three top systems (A2iA, RWTH and LITIS) are shown in Figure 3. RWTH (Aachen, Germany) submitted a system based on a tandem RNN/HMM [25], and did not submit a system for the recognition of printed text. The system of the LITIS (Rouen, France) was also based on HMMs and feature extraction [26]. Results reported in Figure 3 confirm the position of recurrent neural networks as the current state-of-the-art for text recognition, with a significant gap with respect to the pure HMM based approach.

To illustrate the improvement of our system with more training data and with the work presented in this paper,

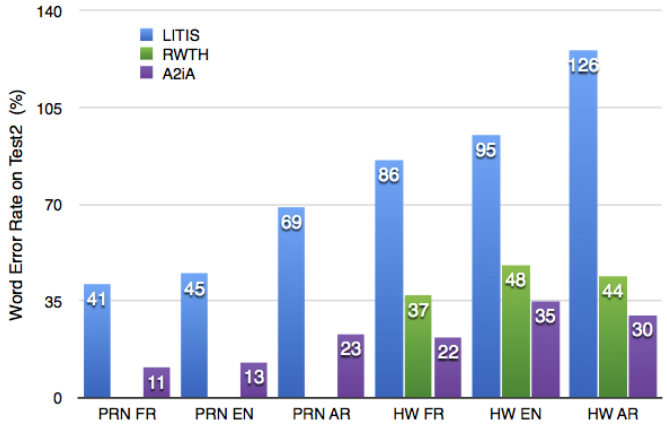


Fig. 3: Official results of second the Maurdor evaluation : word error rate of the top three systems (A2iA, RWTH and LITIS), on the second test set (*Test2*), for printed (PRN) and handwritten (HW) recognition in French (FR), English (EN) and Arabic (AR).

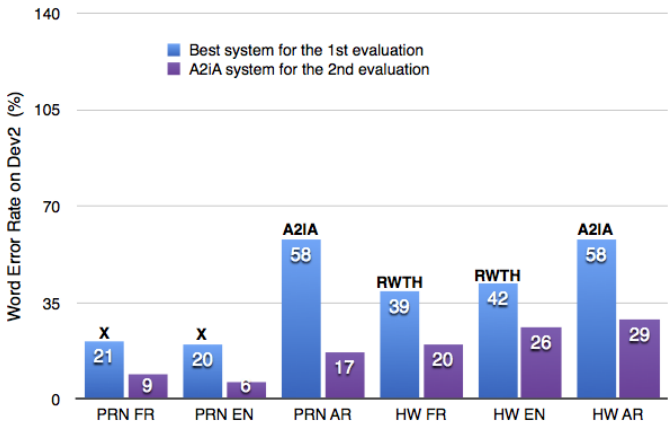


Fig. 4: Comparison of the word error rate of the best systems of the first evaluation (RWTH, A2iA and Anonymous) and the A2iA system of the second evaluation on the test set of the first evaluation.

TABLE V: Evolution of RNN performance after each loop of automatic data annotation on the handwritten English subset.

RNN training set	**# of training lines**	**word error rate**
Single lines	7310	54.7%
First step of automatic location	10570	43.8%
Second step of automatic location	10925	35.2%
Total number of lines (without location)	11608	-

Figure 4 brings the results of the best systems in the first and in the second evaluation on the first test set (*Dev2*). On average, the error rate was divided by a factor two between the two evaluations.

This significant reduction of the error rate is partly due to the increase in the number and the quality of line snippets used for training the optical model, as described in Section IV-A. Extracting line segmentation alternatives (c.f. Section III-B) also improved the performance of the system. Language models in general were improved by the augmentation in the amount of data available to train them and also by careful cleaning up and tokenization of the data. For Arabic systems, the use of hybrid language models (c.f. Section VII-B) improved the results, in particular for printed system. The reduction in WER is more important for printed data, as the quality of the RNN predictions profited the most from the data preparation. Handwriting remains more challenging as the variability is much higher (but we expect the performance to improve with more training data). Overall, the error rates are a bit lower than in the final campaign; we observed a difference in the distribution of the test data sets, where the first one was more similar to the training set, which could explain that difference.

The following sections describe the contribution of the different methods we use in our system.

TABLE VI: WER of Arabic printed and handwritten systems using word and hybrid (word+PAW) LMs.

System	*Dev2[%]*		*Test2[%]*	
	word LM	hybrid LM	word LM	hybrid LM
Printed	19.1	17.3	26.4	22.6
Handwritten	31.0	29.2	30.5	29.5

A. *Impact of the training data preparation?*

Table V shows the importance of the training data preparation explained in Section IV-A for training the English handwriting recognizer. Two cycles of constrained text line alignment were performed. The results show an increase of the number of training lines from 7310 to 10925. Moreover, the alignement on multi-line paragraphs helped the system to better recognize large paragraphs. The second loop did not increase much the number of lines but the quality of the alignments was better. In this case, training data preparation helped to lower the word error rate from 54.7% to 35.2%.

B. *Language models (LM) with part of Arabic word (PAW)*

To evaluate the contribution of PAW, experiments with systems that differ only by the type of LM were conducted. Two types of LMs were compared, the word LM and the hybrid LM generated using the hybrid vocabulary (word+PAW) as explained in Section V. Table VI reports the results for both printed and handwritten Arabic systems, on both *Dev2* and *Test2* datasets. LMs for systems evaluated on the *Dev2* dataset are generated using *Train2* dataset only.

Results show that systems using hybrid LMs consistently outperformed those using word LMs, in particular for printed text.

C. *Impact of the text line detection alternatives*

We assessed the impact of line segmentation alternatives, described in Section III-B. The results are shown in Table VII. We observe that while the substitution and insertion rates show just a little increase between the system with alternatives and

TABLE VII: Improvement due to text line segmentation alternatives.

Type	Language	Del.	Ins.	Sub.	WER.
		Without alternatives			
Hand	French	5.9%	3.0%	16.7%	25.6%
	English	13.6%	5.5%	23.6%	42.7%
	Arabic	7.5%	4.8%	20.9%	33.3%
Printed	French	12.6%	1.7%	5.2%	19.5%
	English	21.6%	1.9%	4.0%	27.5%
	Arabic	7.0%	1.5%	13.7%	22.2%
		With Alternatives			
Hand	French	4.1%	2.9%	15.2%	22.2%
	English	8.3%	5.8%	21.1%	35.2%
	Arabic	6.6%	3.5%	19.8%	29.8%
Printed	French	5.4%	1.1%	4.8%	11.3%
	English	5.8%	1.8%	5.2%	12.8%
	Arabic	6.2%	2.6%	14.0%	22.8%

the system without alternatives, the deletion rate is multiplied by 2 or 3 when there is no alternatives. This can be explained by a poor line segmentation on some paragraphs. If two lines were merged or if a line was not detected, deletion inevitably occured. Giving several line segmentation hypotheses to the system helped to alleviate this problem.

VIII. Conclusion and future work

In this paper, we described the multi-lingual text recognition system developed by A2iA during the Maurdor evaluation campaigns. Based on recurrent neural networks and weighed finite-state transducers, this system was successfully applied to both printed and handwritten text recognition in French, English and Arabic. Thanks to thorough training data preparation, multiple line segmentation hypotheses and hybrid character/word (and PAW for Arabic) language models, the error rate was divided by a factor two on average between the first and the second evaluation.

The main challenge on the documents from the Maurdor database is now to develop a successful complete text recognition system which interconnect the Document Layout Analysis module and the text recognition module.

Acknowledgment

This work was partially funded by the French Defense Agency (DGA) through the Maurdor research contract with Airbus Defense and Space (Cassidian) and supported by the French Grand Emprunt-Investissements d'Avenir program through the PACTE project.

References

[1] V. Märgner, M. Pechwitz, and H. El Abed, "Arabic Handwriting Recognition Competition," in *International Conference on Document Analysis and Recognition*, 2005.

[2] V. Märgner and H. El Abed, "ICDAR 2011 – Arabic Handwriting Recognition Competition," in *International Conference on Document Analysis and Recognition*, 2011.

[3] E. Grosicki and H. El-Abed, "ICDAR 2011: French handwriting recognition competition," in *International Conference on Document Analysis and Recognition*, 2011.

[4] A. Tong, M. Przybocki, V. Märgner, and H. E. Abed, "NIST 2013 open handwriting recognition and translation (openhart'13) evaluation," in *International Workshop on Document Analysis Systems*, 2014.

[5] S. Brunessaux, P. Giroux, B. Grilheres, M. Manta, M. Bodin, K. Choukri, O. Galibert, and J. Kahn, "The maurdor project - improving automatic processing of digital documents," in *International Workshop on Document Analysis Systems*, 2014.

[6] I. Oparin, J. Kahn, and O. Galibert, "First Maurdor 2013 Evaluation Campaign in Scanned Document Image Processing," in *International Conference on Acoustics, Speech, and Signal Processing*, 2014.

[7] T. Bluche, B. Moysset, and C. Kermorvant, "Automatic Line Segmentation and Ground-Truth Alignment of Handwritten Documents," in *International Conference on Frontiers of Handwriting Recognition*, 2014.

[8] P. Simard, D. Steinkraus, and J. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *International Conference on Document Analysis and Recognition*, 2003.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine Learning*, 2006.

[11] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Conference on Neural Information Processing Systems*, 2008.

[12] F. Menasi, J. Louradour, A.-L. Bianne-Bernard, and C. Kermorvant, "The A2iA French handwriting recognition system at the Rimes-ICDAR2011 competition," in *Document Recognition and Retrieval Conference*, 2012.

[13] T. Bluche, J. Louradour, M. Knibbe, B. Moysset, F. Benzeghiba, and C. Kermorvant, "The A2iA Arabic Handwritten Text Recognition System at the OpenHaRT2013 Evaluation," in *International Workshop on Document Analysis Systems*, 2014.

[14] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012.

[15] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," in *International Conference on Frontiers of Handwriting Recognition*, 2014.

[16] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *International Conference on Machine Learning*, 2009.

[17] J. Louradour and C. Kermorvant, "Curriculum learning for handwritten text line recognition," in *International Workshop on Document Analysis Systems*, 2014.

[18] E. Grosicki and H. ElAbed, "ICDAR 2009 handwriting recognition competition," in *International Conference on Document Analysis and Recognition*, 2009.

[19] F. Slimane, R. Ingold, S. Kanoun, A. M. Alimi, and J. Hennebert, "A new arabic printed text image database and evaluation protocols," in *International Conference on Document Analysis and Recognition*, 2009, pp. 946–950.

[20] "GNU FriBidi." [Online]. Available: http://fribidi.org

[21] I. Witten and T. Bell, " The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression," *IEEE Transactions on Information Theory*, vol. 37, no. 4, 1991.

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *Workshop on Automatic Speech Recognition and Understanding*, 2011.

[23] M. Mohri, "Finite-State Transducers in Language and Speech Processing," *Computational Linguistics*, vol. 23, pp. 269–311, 1997.

[24] A. Stolcke, "SRILM – An Extensible Language Modeling Toolkit," in *International Conference on Spoken Language Processing*, 2002.

[25] M. Kozielski, P. Doetsch, M. Hamdani, and H. Ney, "Multilingual off-line handwriting recognition in real-world images," in *International Workshop on Document Analysis Systems*, 2014.

[26] K. Ait-Mohand, T. Paquet, and N. Ragot, "Combining structure and parameter adaptation of HMMs for printed text recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.