The A2iA French handwriting recognition system at the Rimes-ICDAR2011 competition

Farès Menasri, Jérôme Louradour, Anne-Laure Bianne-Bernard and Christopher Kermorvant

A2iA SA, Paris, France

ABSTRACT

This paper describes the system for the recognition of French handwriting submitted by A2iA to the competition organized at ICDAR2011 using the Rimes database. This system is composed of several recognizers based on three different recognition technologies, combined using a novel combination method. A framework multi-word recognition based on weighted finite state transducers is presented, using an explicit word segmentation, a combination of isolated word recognizers and a language model. The system was tested both for isolated word recognition and for multi-word line recognition and submitted to the RIMES-ICDAR2011 competition. This system outperformed all previously proposed systems on these tasks.

Keywords: Isolated French word recognition, the Rimes database, HMM, recurrent neural networks, system combination

1. INTRODUCTION

Even if automatic handwriting recognition is an old challenge, the research in this field seems to have switched gears since the introduction of international competitions. Twenty years after the first evaluations in automatic speech recognition organized by the NIST,[1] international handwriting recognition competitions have been organized in 2005 for Arabic and 2007 for French. Since then, the number of competitions organized annually during the major conferences of the domain has increased, maybe to reach a too large number relatively to the size of the community. However, the competitions are a good way to stimulate the development of complete systems for a target application, by proposing tasks of increasing complexity and by tracking the progress of the different research teams thanks to controlled evaluation on unseen data.

The Rimes Database[2] was designed to provide data for a variety a recognition problems related to automatic processing of handwritten documents: document layout analysis, isolated character, isolated word or paragraph recognition. If the first competitions were dedicated to isolated word recognition, the last competition organized at the conference ICDAR in 2011 contained a task on paragraph recognition, which is closer to the final application.

We have submitted a system to the Rimes-ICDAR2011 competition[3] for the two proposed tasks: isolated word recognition and multi-word recognition. This system is a combination of several recognizers based on three different technologies: grapheme-based hybrid Hidden Markov Models (HMM), Gaussian mixture HMM and recurrent neural networks. These three technologies encompass the majority of the systems proposed in the previous competitions. By combining them, we hope to take advantage of the strengths of each technologies.

In this paper, we first present the three technologies used for the isolated word recognizers. Then, we presents our method for optimizing the combination of the isolated word recognizers, based on a weighted sum-rule in which the mixing coefficients are trained. We sketch the principles of our framework for multi-word recognition, based on weighted finite state transducers. In this framework, thanks to the use of an explicit word segmentation, we can base our multi-word recognizer on a combination of isolated word recognizers and we introduce the use of language models. Finally, we report our results on the two different tasks of the competition, isolated and multi-word recognition. For both tasks, our system achieve the best performances ever reported on this database.

2. ISOLATED WORD RECOGNITION SYSTEMS

We present in this section the three different technologies on which our combined system was based: grapheme-based MLP-HMM, Gaussian mixture HMM and recurrent neural networks.

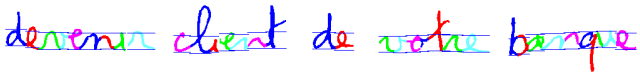

Figure 1. Decomposition of the French line "devenir client de votre banque" into graphemes.

2.1 Grapheme-based MLP-HMM

This recognizer was the first recognizer developed at A2iA. After a binarization of the input image and baseline extraction, the word to be recognized is decomposed into parts of letters called graphemes as shown on Fig. 1. This segmentation is an over-segmentation which means that a grapheme is either a character or a subpart of a character. On each grapheme, 74 geometric features are extracted as described in detail by Bianne et al.[4] These features are given as input to a multi-layer perceptron (MLP), with one hidden layer (400 units), as many output neurons as grapheme classes (172) and a softmax transfer function which is trained with a supervised stochastic back-propagation algorithm.

2.2 sliding window Gaussian mixture HMM

This recognizer is based on a modeling of each letter with a hidden Markov model (HMM) using Gaussian distribution mixture for the observation probabilities. Each character HMM is composed of 10 emitting states with a Bakis left-to-right topology allowing self-loops. 33 features are extracted with sliding windows with 8 pixels width and 4 pixels shift. Features are a combination of 25 geometric and statistical features and 8 directional features based on histogram of gradients and adapted to our extraction windows.[4] We also append delta coefficients, which are computed for each feature using a regression on the 4 neighbouring windows.

In order to model the variations of a letter depending on its neighborhood, the character HMMs are calculated context-dependently. The new models are called trigraphs. A trigraph centered on a given character models this character given a preceding and following context. Since the use of context-dependent models increase dramatically the number of parameters, a tying of states is done. The tying is based on decision trees, and state clusters are computed for each state position of each central character. Thanks to the tree-based state tying, we defined a total of 2472 state clusters and used them to model 1664 trigraphs. More details can be found in.[4]

2.3 Recurrent neural networks

The third type of system is a Multi-Dimensional Long-Short Term Memory (MDLSTM)[5] recurrent neural network*. Unlike the two previous systems, this recognizer takes the image of a word as input (the raw values of pixels) and does not rely on a handcrafted feature extraction. The system trains its own embedded feature extraction given the data.

The number of outputs of the recurrent neural network is equal to the number of symbols in the alphabet (80 : upper-case and lower-case letters, plus digits and a few punctuations symbols), plus one output for the white space character. The training is made at word level without requiring character-level pre-segmented data, thanks to the use of Connectionist Temporal Classification (CTC).[6]

When the neural networks are trained with stochastic gradient descent, different random initializations yield different systems but with similar performances. Combining those instances of neural networks is a simple yet very powerful way to improve the overall recognition rate (see section 5). As illustrated in,[7] this is probably due to the fact that a combination of systems computes a more precise overall estimation of the decision boundaries than an individual expert.

3. SYSTEM COMBINATION METHODS

We have compared in a previous work[8] different kinds of combination strategies for isolated word recognition and we have shown that improved results can be achieved with a simple sum-rule that affects equal weights for all the recognizers. In this work, starting from the weighted sum-rule combination method, we show how to optimize the weights to take into account the effectiveness of each recognizer, as well as their complementarity.

*We used the RNNLib (http://sourceforge.net/projects/rnnl/) to train the models and to decode.

Suppose we have N recognizers to combine. Provided that each recognizer $i = 1 \dots N$ returns confidence scores along with several hypotheses for the word to predict, the weighted sum-rule with weights $\omega_i \geq 0$ consists in choosing the word $\hat{k}$ which maximizes a linear combination of the scores:

$$\hat{k} = \arg\max_k \sum_{i=1}^{N} \omega_i \times s_i(k) \tag{1}$$

where $s_i(k)$ is the confidence score of recognizer i for word k. A major advantage over simple sum-rule ($\omega_i = 1$, $i = 1 \dots N$) is that it saves from doing any assumption about the range of the confidence scores. The goal is to optimize the N weights with a discriminative criterion and find efficiently the best way to combine recognition scores. The optimal setting depends on the relative recognition accuracies and on how complementary the recognition scores are. This information can be supplied by recognition results on some data that has not been used to train the recognizers (otherwise, the distribution of scores would be biased).

Note that equation (1) can be generalized using any strictly increasing function σ

$$\hat{k} = \arg\max_k \sigma\big(\textstyle\sum_i \omega_i s_i(k)\big) \tag{2}$$

Here we propose to perform logistic regression using $\sigma(x) = 1/(1+\exp(-x-b))$, where b is a bias parameter. Each word is associated to a combined confidence score $\sigma\big(\sum_i \omega_i s_i(k)\big)$ bounded in [0,1]. We can associate each combined score with a 0/1 target and consider the Kullback-Leibler (KL) divergence to optimize the weights ω_i along with the bias b. If k^* denotes the true word to be recognized, this loss can be expressed as:

$$\mathcal{L}_{\mathrm{KL}}\Big(\big\{s_i(k)\big\}_{i,k}, k^*\Big) = -\log\Big(1 - \sigma\big(\textstyle\sum_i \omega_i s_i(k^*)\big)\Big) - \displaystyle\sum_{k \neq k^*} \log\Big(\sigma\big(\textstyle\sum_i \omega_i s_i(k)\big)\Big) \tag{3}$$

Another possible loss function is the mean squared error (MSE) with the same 0/1 targets. Both losses are popular, and preliminary experiments showed that KL generally performs at least as well as MSE. It only requires one precaution: values of σ must be constrained to lie in a range $[\epsilon, 1-\epsilon]$ $(\epsilon > 0)^\dagger$ in order to avoid numerical problems.

As the number of candidate words is very high, the right-hand sum of loss (3) involves a lot of terms which encourage the combined confidence σ to take small values (target 0), to the detriment of the left-hand term (target 1). In order to have a disciminant criterion that is better "balanced", some variants can be used as proposed by Tulyakov and Govindaraju[9] :

- *Standard (top-10 list for all recognizers):*
  We take into account all the words that appear in the top-10 list for at least one recognizer (other words are treated as if $s_i(k) = 0$). In our experiments, the recognition scores often decrease quickly so this approach is very close to optimizing the "full" loss (3).

- *Best Impostor:*
  Instead of the right-hand sum of (3), we only take the word with the highest combined confidence score:

$$\mathcal{L}_{\mathrm{KL}}^{\mathrm{BestImp}}\Big(\big\{s_i(k)\big\}, k^*\Big) = -\log\Big(1 - \sigma\big(\textstyle\sum_i \omega_i s_i(k^*)\big)\Big) - \log\Big(\displaystyle\max_{k \neq k^*} \sigma\big(\textstyle\sum_i \omega_i s_i(k)\big)\Big) \tag{4}$$

- *1-Best (or "Mixed"[9]):*
  The same as *Best Impostor* but we only take into account the "best impostor" when the combined system is wrong:

$$\mathcal{L}_{\mathrm{KL}}^{\mathrm{1Best}}\Big(\big\{s_i(k)\big\}, k^*\Big) = -\log\Big(1 - \sigma\big(\textstyle\sum_i \omega_i s_i(k^*)\big)\Big) \left[-\log\Big(\displaystyle\max_{k \neq k^*} \sigma\big(\textstyle\sum_i \omega_i s_i(k)\big)\Big) \quad \text{if} \quad \sigma\big(\textstyle\sum_i \omega_i s_i(k^*)\big) < \displaystyle\max_{k \neq k^*} \sigma\big(\textstyle\sum_i \omega_i s_i(k)\big) \right] \tag{5}$$

[†] we used $\epsilon = 1.e^{-10}$: $\sigma_{safe} = \max(\epsilon, \min(1-\epsilon, \sigma))$

Note that these two last variants involve non-continuous loss functions (4) and (5), the notion of "best impostor" being dependent of the weights parameters ω_i. However, we optimize these losses using stochastic gradient descent[10]; In practice, this method converges for a large range of learning rates, and the choice of the learning rate is not critical[‡]. We initialize the gradient descent with equal positive weights $\omega_i = 5/N$ and bias $b = -2.5$ because the logistic sigmoid $x \mapsto 1/(1 + \exp(-5 \times x + 2.5))$ is close to identity (and because all the recognition scores are calibrated in $[0, 1]$).

4. MULTI-WORD RECOGNITION SYSTEM

We have developed a framework for multi-word recognition based on weighted finite state transducers (WFST), as it has been proposed for speech recognition.[11] The three steps of our multi-word recognition system - word segmentation, word recognition and language models - are defined using WFST. The steps are chained using composition operations between the different WFST and the final recognition results is obtained with a best-path algorithm.

4.1 Weighted finite state transducers

Recently, weighted finite state transducers have seen a wide adoption for designing automatic speech recognition systems. WFST are an extension of the well known finite state automata, in which each transition is associated to an input symbol, an output symbol and a weight. The weight are the elements of a semi-ring which means that two internal operations, denoted $\bigoplus$ and $\bigotimes$, are defined together with their associated identity elements $\bar{0}$ and $\bar{1}$. For handwriting recognition, following what is done for speech recognition, we chose to use the tropical semi-ring $(\mathbb{R} \cup \infty, \min, +, \infty, 0)$, in which the weights are negative log probabilities and the log addition $\bigoplus_l$ is approximated with a Viterbi algorithm (the log scale is used for numerical stability).

We have based our recognition framework on the OpenFst library[§] which provides all the algorithms needed to create and manipulate WFST. We have developed all the extensions specific to handwriting recognition (or written text recognition in general): attach an image to a WFST transition, unicode representation of recognized characters and words, writing type (printed, hand-printed, cursive).

Our multi-word recognition process can be decomposed into 3 different WFST :

S : the word segmentation WFST. The edges of this transducer have a word image hypothesis as input and output, and a negative log probability of the word segmentation as weight.

R : the word recognition WFST. The edges of this transducer have a word image hypothesis as input, a word recognition hypothesis as output and the negative log probability (or likelihood) of the recognition as weight.

G : the language model WFST. The edges of this transducer have a word recognition hypothesis as input and output, and the negative log probability of the n-gram sequence as weight.

The complete recognition model corresponds to $S \circ R \circ G$, where $\circ$ is the transducer composition operation. In the next section, we describe the three transducers in more detail.

4.2 Multi-word recognition using WFST

Word segmentation: Our approach to multi-word recognition is based on an explicit word segmentation. This approach assumes that the words in a line can be segmented based on the spaces between them. This condition is usually met for clean documents written in European languages but not in Arabic for example. The word segmentation module takes an image of a line as input, and outputs a weighted segmentation graph which contains the different hypothesis of the segmentation of the line into words. The algorithm for generating the word segmentation hypothesis is the following:

[‡]the results are the same, as long as the learning rate is small enough

[§]http://www.openfst.org/

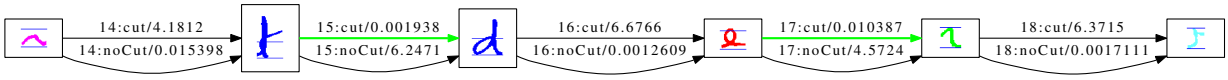

Figure 2. Graph of segmentation cuts on line "devenir client de votre banque" (the weights on edges are negative log probabilities). Only middle part of the line, centered on word "de", is displayed. True cut edges are displayed in bold green.

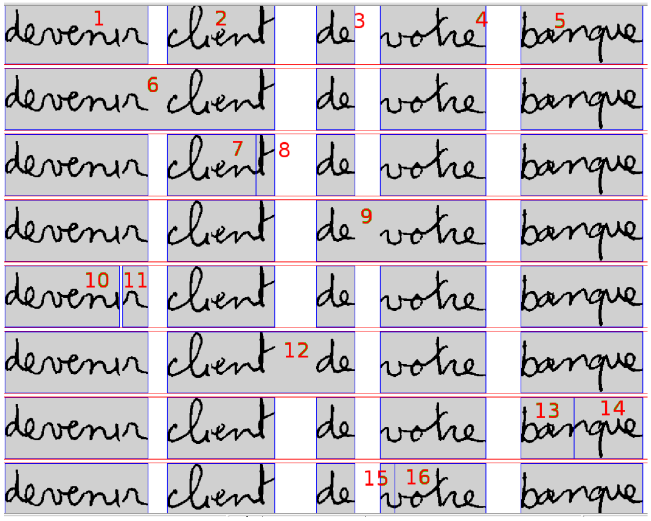

Figure 3. List of alternatives ways of segmenting a line into words.

1. the grapheme segmentation is computed on the whole line (see Fig. 1).
2. a neural network trained to compute the posterior probability of segmenting into words between each pair of two consecutive graphemes is used to compute the graph of segmentation cuts (see Fig. 2).
3. word segmentation hypothesis are built by computing N-Best paths in the graph of segmentation cuts.
4. a segmentation WFST is constructed with the word segmentation hypothesis

The result on a handwritten line example is given on Fig. 3.

Recognition lattices at line level Given a segmentation graph, each recognizer outputs a recognition lattice at line level. The recognition lattice has as many states as the segmentation graph, and each edge in the segmentation graph is replaced by the N-Best word candidates provided by the recognizer. The fact that the different word recognizers rely on the same segmentation graph implies that their recognition lattices will share the same topology, therefore making their combination straightforward using combination methods described in section 3.

Language models Smoothed language models can be modeled with WFST as shown in 12. For the composition of the recognition transducer and the language model transducer, a scaling factor is used to compensate for the difference in scale between the recognition log probabilities and the language model log probabilities. The value of the language scaling factor must be optimized on a validation set. We have trained the language models using the SRILM toolkit[¶] and we made the conversion from the arpa format to the openfst format.

[¶]http://www-speech.sri.com/projects/srilm/

Table 1. Isolated word recognition : word error rates of the individual systems on ICDAR2009 and ICDAR2011 test set.

Id	System type	System details	ICDAR2009 test set	ICDAR2011 test set
1	Grapheme MLP-HMM	binary images	24.9	25.0
2	Sliding GMM-HMM	binary images, context-independent	28.6	
3		binary images, context-dependent	21.5	22.0
4		grey-level images, context-dependent	22.5	23.1
5	Recurrent NN	grey level images, init 1	10.5	8.9
6		grey level images, init 2	10.0	9.2
7		binary images, init 1	10.2	9.6
8		binary images, init 2	9.9	9.5

5. EXPERIMENTAL RESULTS

In this section, we report the recognition rates of our system on the two tasks of the Rimes-ICDAR2011 competition.

5.1 Database and task description

Two tasks were proposed at the ICDAR2011 competition : isolated word recognition and multi-word recognition. The experiments were conducted on the different subsets of the Rimes database.[2]

For the isolated word recognition task, the different sets are composed of images containing a single handwritten word. The ICDAR2011 train set (51,739 images) was used to train the isolated word recognizers. The ICDAR2011 validation set (7,464 images) was used for the training of the combination coefficients. Note that this set is equal to the ICDAR2009 test set, allowing direct comparison with the results of the previous competition. The ICDAR2011 test set (7,776 images) was used only once for the final evaluation of the combined recognizer. For this task, the vocabulary contained 5,744 different words, was closed (no out-of-vocabulary word) and given to the participants.

For the line level recognition, the ICDAR2011 train set (1500 paragraph images) was only used to select the vocabulary and to train the language models (first 1300 images of the train set), and to optimize the language scaling factor (last 200 images of the train set). The test set was composed of 100 paragraph images and the test set vocabulary was not given. The line level recognizers were not trained on the line level train set but only on the word level train set.

In both cases, the evaluation was done after normalizing the recognized text and the ground truth value to lower case, which means that the evaluation was case insensitive ("*Ose*" and "*ose*" were considered equal). On the other hand, the accents were considered for the evaluation ("*ose*" and "*osé*" were considered different). For the line level recognition task, the evaluation was performed with the Sclite scoring software[‖].

5.2 Isolated word recognition task

5.2.1 Individual system evaluation

The recognition results of each recognizer individually are shown on Tab. 1. The best results are obtained by the recognizer based on recurrent neural networks, with an error rate around 10%, whereas the recognizers based on HMM (hybrid MLP-HMM and GMM-HMM) yield an error rate at least twice higher. As shown previously[4] using context-dependent GMM-HMM improves over context-independent GMM-HMM (25% of error rate reduction).

[‖] http://www.itl.nist.gov/iad/mig/tools/

Table 2. Results of several strategies to combine recognizers on ICDAR 2009 and 2011 test sets. The best performance is indicated in a gray cell, and results in bold are the ones with no significant difference with this best reference (based on a 95% two-sided paired Student's t-test). The evaluation is done using case-insensitive words with accent.

Evaluation type	**WER (%) (our systems)**				**WER (%) (other systems)**		
	Sum Rule	Logistic Regression			Jouve	IRISA	TUM
		Standard	*Best Impostor*	*1-Best*			
ICDAR2009 Test set	5.33	**4.93**	5.05	**4.82**	-	25.31[13]	8.98[13]
ICDAR2011 Test set	5.16	**4.86**	5.00	**4.75**	12.53[14]	21.41[14]	-

Table 3. multi-word recognition : word error rates of the combined systems on the ICDAR2011 test set.

Test set	**LM perplexity**	**OOV rate**	**Error rate**
Validation set (our system)	185	4.99%	11.8%
ICDAR2011 test set (our system)	196	4.74%	15.2%
Telecom ParisTech[14]	-	-	31.2%

5.2.2 Combined system evaluation

For the combined system, we consider the following recognizers (with their identifier as presented in Table 1) : Grapheme based MLP-HMM (1), context-dependent sliding window GMM-HMM on binary (3) and grey level (4) images, four variants of recurrent neural networks (5, 6, 7, 8).

We present in Table 2 the evaluation of the different methods to combine recognizers described in Section 3. The system combining the recognizers with a equal weight sum rule achieve an error rate of just above 5%, which corresponds to more than 50% of error rate reduction compared to the best individual recognizer. Moreover, a significant improvement can be achieved by learning the weights of the combination, instead of arbitrarily fixing all weights to 1, leading to an error rate of the combined system below 5%. Besides, our experiments corroborate the results obtained by Tulyakov and Govindaraju[9] : The *1-Best* training strategy (5) outperforms others, even though the improvement over standard optimization may be not significant.

5.3 Multi-word recognition task

For the multi-word recognition task, we could not use the best combination of recognizer presented in the previous section because we had only binary images at the output of the line segmentation algorithm. We used a combination of isolated word recognizer trained on binary images only : 3 recurrent neural networks with different random initializations and the grapheme-based MLP-HMM.

The vocabulary was collected on the human transcription corresponding to the first 1300 images of the official train set and the language model was trained on the same set. Since the number of words for training the language model was very limited (71,580 occurrences of 6039 different words), we only used bigrams. The language model scaling factor was optimized on a validation set composed of the last 200 images of the official train set, not used for training the language model.

Table 3 shows the recognition results on the validation and test sets. The word error rate on the evaluation set (15.2%) is higher than expected, as measured on the validation set (11.8%). This can be due to a higher perplexity, 196 on the test set versus 185 on the evaluation set. The out-of-vocabulary rate is stable just below 5%, which explains 30% of the error rate on the evaluation set.

6. CONCLUSION AND FUTURE WORK

We have presented a French handwriting recognition system based on the combination of three different technologies. For multi-word recognition, this system can integrate the isolated word recognizer combination thank to the use of an explicit word segmentation and include language models in a framework based on weighted finite state transducers. As shown in tables 2 and 3, this system achieved the best recognition rate reported so far on this database for both isolated word recognition and multi-word recognition tasks.

However, the explicit word segmentation cannot be used for handwriting in which the word separation is ambiguous, for example for cramped handwriting, or for Arabic whose words contain spaces. In future work,

we plan to develop multi-word a recognizer without explicit word segmentation, for example by extending our sliding window recognizer from a single word to several words. Other combination methods for line level recognition will then become necessary, using Word Transition Networks (WTNs) like for example in [15] , or other ROVER-based algorithms.[16]

Acknowledgments

This work was partly achieved as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

REFERENCES

[1] National Institute of Standards and Technology, "The History of Automatic Speech Recognition Evaluations at NIST."

[2] Augustin, E., Brodin, J.-m., Carré, M., Geoffrois, E., Grosicki, E., and Prêteux, F., "RIMES evaluation campaign for handwritten mail processing," in [*Proc. of the Workshop on Frontiers in Handwriting Recognition*], (1) (2006).

[3] Grosicki, E. and El-Abed, H., "ICDAR 2011: French handwriting recognition competition," in [*Proc. of the Int. Conf. on Document Analysis and Recognition*], (2011).

[4] Bianne, A.-L., Menasri, F., Al-Hajj, R., Mokbel, C., Kermorvant, C., and Likforman-Sulem, L., "Dynamic and Contextual Information in HMM modeling for Handwriting Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence* (2011).

[5] Graves, A. and Schmidhuber, J., "Offline handwriting recognition with multidimensional recurrent neural networks," in [*Advances in Neural Information Processing Systems*], 545–552 (2009).

[6] Graves, A., Fernández, S., and Gomez, F., "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in [*Proc. of the International Conference on Machine Learning*], 369–376 (2006).

[7] Polikar, R., "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine* **6**(3), 21–45 (2006).

[8] Kermorvant, C. and Louradour, J., "Handwritten mail classification experiments with the Rimes database," in [*Proc. of the Int. Conf. on Frontiers in Handwriting Recognition*], (2010).

[9] Tulyakov, S. and Govindaraju, V., "Neural Network Optimization for Combinations in Identification Systems," in [*Proc. of the 8th International Workshop on Multiple Classifier Systems*], (2009).

[10] Bottou, L., "Stochastic learning," in [*Advanced Lectures on Machine Learning*], Bousquet, O. and von Luxburg, U., eds., *Lecture Notes in Artificial Intelligence, LNAI 3176*, 146–168, Springer Verlag, Berlin (2004).

[11] Mohri, M., "Finite-state transducers in language and speech processing," *Computational Linguistics* **23**, 269–311 (June 1997).

[12] Allauzen, C., Mohri, M., and Roark, B., "Generalized algorithms for constructing statistical language models," in [*Proc. of Annual Meeting on Association for Computational Linguistics*], 40–47 (2003).

[13] Grosicki, E. and Abed, H. E., "Icdar 2009 handwriting recognition competition," in [*ICDAR*], 1398–1402, IEEE Computer Society (2009).

[14] Grosicki, E. and El Abed, H., "Icdar 2011: French handwriting recognition competition," 1459–1463 (2011).

[15] Bertolami, R. and Bunke, H., "Hidden markov model-based ensemble methods for offline handwritten text line recognition," *Pattern Recogn.* **41**, 3452–3460 (November 2008).

[16] Fiscus, J., "A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER)," *Workshop on Automatic Speech Recognition and Understanding Proceedings* (February), 347–354 (1997).