Towards Unsupervised Learning for Handwriting Recognition

Michał Kozielski, Malte Nuhn, Patrick Doetsch, Hermann Ney
Human Language Technology and Pattern Recognition Group
Computer Science Department
RWTH Aachen University, D-52056 Aachen, Germany
{*kozielski,nuhn,doetsch,ney*}*@cs.rwth-aachen.de*

***Abstract*—We present a method for training an off-line handwriting recognition system in an unsupervised manner. For an isolated word recognition task, we are able to bootstrap the system without any annotated data. We then retrain the system using the best hypothesis from a previous recognition pass in an iterative fashion. Our approach relies only on a prior language model and does not depend on an explicit segmentation of words into characters. The resulting system shows a promising performance on a standard dataset in comparison to a system trained in a supervised fashion for the same amount of training data.**

I. Introduction

The training process of a state-of-the-art handwriting recognition system requires a considerate amount of annotated images of text sentences. The annotation process is expensive and further may prove difficult if the images are degraded or the structure of text is not clear. Additionally the high variety of writing styles increases the effort needed to collect a representative dataset.

The problem of lack of a sufficient amount of training data has been already addressed with approaches based on the semi-supervised learning. The underlying idea of those approaches is to transcribe the unannotated data with a recognition system, and then use part of the data for retraining. Dreuw [1] selects the additional training data using state confidences. Frinken [2] proposes another approach that is based on keyword spotting. In another publication Frinken [3] utilizes the concept of co-training, where two systems try to improve each other. Still all those methods rely on some small amount of annotated data [4] or a bootstrap system [5].

The early work on completely unsupervised training has been done in the field of the machine-printed text recognition (OCR) and utilized cipher breaking algorithms [6]. It is clear that assuming perfect segmentation and clustering of characters the problem transforms to breaking a simple substitution cypher [7]. Some methods have been also developed that do not rely on an explicit segmentation [8][9].

The decipherment problem has been studied extensively in a number of symbol substitution problems like 1:1 substitution ciphers [10][11], homophonic ciphers [12][13], and machine translation [14][15]. In case of 1:1 substitution ciphers, each symbol of the cipher text is substituted by a unique substitute. The general idea is to choose the model parameters — i.e. the substitution $\phi : V_f \rightarrow V_e$ of ciphertext tokens into plaintext characters — in such a way, that the resulting decipherment yields the highest possible language model probability:

$$\tilde{\phi} = \arg\max_{\phi} \{p(\phi(f_1)\phi(f_2)\dots\phi(f_N)\}$$

Although this case seems very simple, it has been shown that finding the optimal ϕ is NP hard [16]. In contrast to this simple problem, decipherment for handwriting recognition is even more complicated: Instead of discrete symbols f_1^N, we have continuous valued observations x_1^T. Furthermore we are facing the alignment problem of assigning the input observations x_1^T to the output words w_1^N. Instead of learning the substitution ϕ, we want to learn the parameters ϑ of the visual (acoustic) model $p_\vartheta(x_1^T|w_1^N)$.

In this paper we demonstrate a method that applies the idea of decipherment directly to the recognition of isolated handwritten words. We propose a method to iteratively uncover the transcriptions of the images and improve the parameters of a system. Our method neither makes use of any amount of annotated data nor of a bootstrap system. The segmentation is implicitly discovered during the training process. We obtain promising results even with a large vocabulary and in the presence of a high out-of-vocabulary (OOV) rate. The language model was trained on a text that comes from a different source than the hidden transcriptions of the images.

Our approach fits well into the standard Hidden Markov Model (HMM) framework based on the sliding window [17][18][19]. The same concept can be extended to problems such as speech recognition or sign language recognition.

II. Proposed method

We train the recognition system in an iterative fashion. The transcriptions of the images and the parameters of the system are updated in every iteration. Our method (depicted in Figure 1) consists of training and recognition passes on one dataset. We use the best hypothesis from a recognition pass as the transcriptions for the training pass in the next iteration. Section II-C gives an insight into this procedure. One iteration of the training procedure of the unsupervised

system is equivalent to the whole training procedure of the supervised system, with the exception that the transcriptions are predicted by the system itself and are not given. The system trained in the last iteration is the final system. Special considerations (described Section II-B) have to be made during the initialization procedure, because we neither have initial transcriptions nor initial models. Our system is based on a standard HMM recognizer with features extracted directly from preprocessed images using the sliding window approach. The structure of the system is described in Section II-A.

A. System overview

One character is represented by a left-to-right HMM with loop transitions. Every HMM consist of six states. The transitions between states (treated as penalties in negative log scale) are constant and fixed across all models. The loop penalty is equal to 3, and the forward transition is not penalized. There is an additional penalty for exiting the HMM model, which controls the number of insertions during recognition. The HMM representing whitespace is special in the sense that it has only one state and no loop penalty. As emission distributions we use Gaussian mixtures with 128 densities per state and a globally-pooled diagonal covariance matrix. We train the system with the Viterbi algorithm using maximum likelihood (ML) as training criterion.

For preprocessing, we correct the slant of gray-scale images with a median of angle values estimated by three different deslanting algorithms [19]. Then we segment the images with a sliding window of a constant shift and width, and a height equal to the size of the original image. A horizontal cosine window is applied to each frame to smooth the image on its borders. Each frame is then normalized to a size of 8×32 pixels using 1st- and 2nd-order moments [20]. We then take gray-scale values of all pixels and reduce their number to 20 components using PCA transformation. The final feature vector of size 24 is augmented by original moment values. The details of the preprocessing scheme are described in [19].

B. Initialization

The initialization procedure is a critical part of the system because there is always a risk of running into a poor local optimum right from the beginning. The major problem we face here is that the images contain an unknown amount of whitespace that is not represented in our language model and thus impossible to predict a priori. Therefore in the first place we use a heuristic [21] to reliably distinguish the whitespace from the text.

The algorithm assumes that the sequence of frames consists of three fragments: whitespace, text, whitespace. It then tries to learn the parameters of two Gaussian densities representing whitespace and text by minimizing the log likelihood criterion and implicitly finding the boundaries between those three fragments. We then use those two Gaussian densities as emission distributions of our two initial HMMs. The first one represents whitespace and the second one represents an arbitrary character. We call the second HMM a gap model later on. All six states of this model share the same emission distribution resulting from the linear segmentation procedure. Throughout the training process we use the gap model to represent any character that is not explicitly modeled with a separate HMM. In the first iteration every character from the character inventory is represented with the gap model.

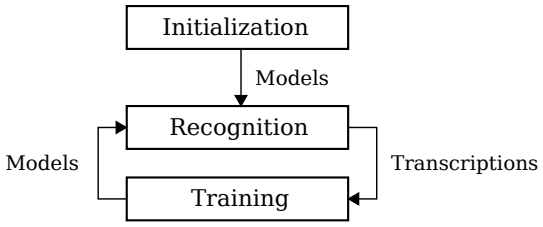

Figure 1: Illustration of the unsupervised training procedure.

C. Iterative training

In the first recognition pass different words of the same length are undistinguishable for the decoder because they are just sequences of gap HMMs. It means that the difference in probabilities between those words comes only from the language model. Consequently in the first iteration we are able to hypothesize only a very limited set of words - those with the highest prior probability among the words of the same length. However we expect that those words are also the most frequently occurring in the images.

In the second iteration we use the best hypothesis from the previous iteration as transcriptions of the training set to retrain the models from scratch (including the whitespace model). The odds are high that some of the less frequent characters do not appear in this transcription and therefore we are not be able to train HMM models for them. To solve this problem we apply the same trick that we applied in the initialization procedure, namely we represent every unseen character in the lexicon with the gap model.

In the following iterations the system should be gaining more confidence about the most frequent characters and thus be able to guess the less frequent characters, because we enforce the structure of a word using the fixed vocabulary. Thus we will be training precise models for an increasing number of characters and therefore be predicting a larger number of words from the vocabulary with a higher confidence.

D. Convergence

We measure the Levenshtein alignment between the uncovered transcriptions in two succeeding training iterations to detect convergence. If the dissimilarity falls below a certain threshold we stop. It could be tempting to use

Table I: Comparison of perplexities on character- and word-level on the IAM validation set. The language was created either using the reduced or the complete vocabulary.

Vocabulary	train		dev	
	char	word	char	word
Reduced (10k)	3.57	808	3.49	673
Complete (44k)	3.71	1332	3.66	1190

Table II: Comparison between error rates in the supervised and unsupervised scenario on the IAM dataset. The language model was created using the complete vocabulary.

	train [%]		dev [%]	
	CER	WER	CER	WER
Supervised	7.5	14.9	9.7	20.5
Unsupervised	15.8	30.7	13.7	28.6

the error rate on a validation set instead, but this is not possible in a fully unsupervised scenario because we have no annotations. The choice of the correct stopping criterion and avoiding overfiting is however not critical, because by definition we are trying to fit the training set as good as possible. We discuss the convergence in more details in Section III-C.

E. Search space

We use the word conditioned tree search strategy in our decoder [22]. For the unsupervised learning method described in this paper it is critical to choose a task that is small enough so that we can explore the whole search space without pruning at the state level. The state-level pruning is applied long before we consider the language model score at word end hypotheses, and because our initial acoustic models are weak, the best hypotheses are likely to be pruned before the word end is eventually reached.

Beside that, there are also other means to reduce the size of the search space. The search space is exponential in the order of the language model [22] which is why we rely on unigrams only. Another way is to use a smaller vocabulary at the cost of increasing the OOV rate.

III. Experiments

We evaluate our approach by comparing a system trained in an unsupervised fashion to the supervised-trained system. Error rates and statistics are calculated using the Levenshtein alignment between reference and hypothesis. As evaluation metrics for our experiments we use the word error rate (WER) and character error rate (CER). Both systems have been trained on the training dataset, with the exception that in the unsupervised scenario we forget about the transcriptions of the images. The validation set was unseen to both systems.

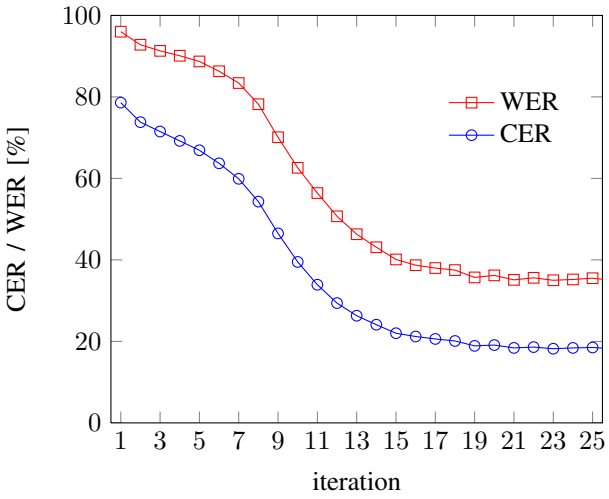

Figure 2: Character and word error rates after every iteration on the IAM training set. The language model was created using the complete vocabulary.

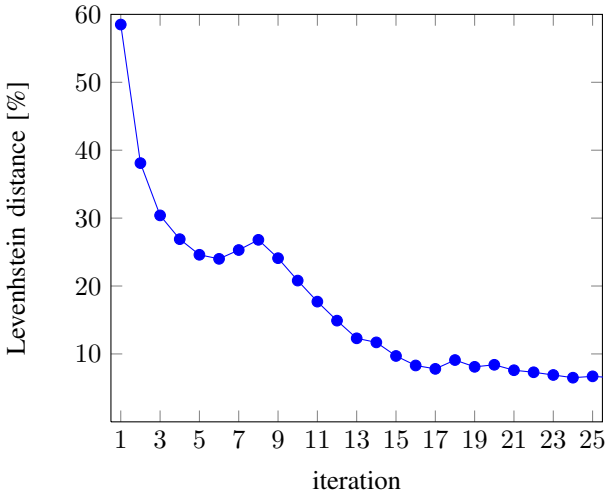

Figure 3: Levenshtein distance as a percentage of the total number of words between transcriptions in every iteration on the IAM training set.

The system structure and preprocessing scheme was the same for both training strategies: supervised and unsupervised one. System parameters such as: language model scale, number of HMM states, transition probabilities; were optimized during the supervised training. Those parameters were not optimized during the unsupervised training, so that the effect of unsupervised training can be isolated and measured.

A. Datasets

The IAM dataset [23] consist of images of isolated handwritten English words. From this dataset we have selected only regular words (we discarded numbers and punctuation

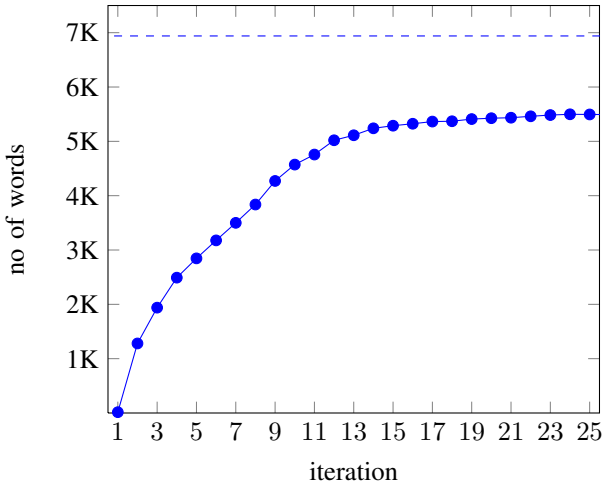

Figure 4: Number of unique words recognized by the system in every iteration on the IAM training set.

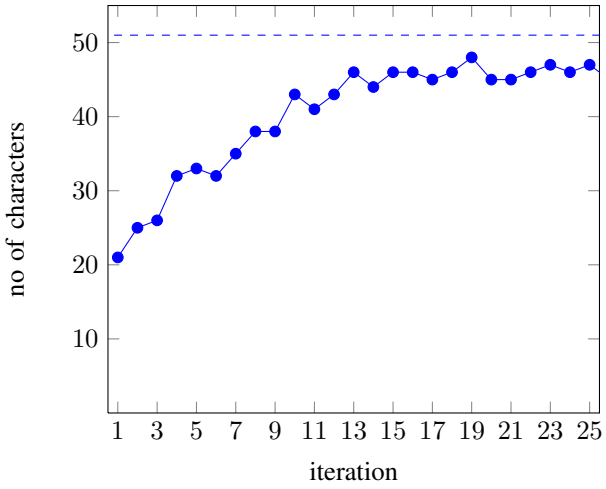

Figure 5: Number of characters (upper and lower case) models learned by the system in every iteration on the IAM training set.

marks), which results in 46k running words for training and 7k running words for validation. We trained the language model on a corpus of English text [24] containing 1 million running words. The transcriptions of the IAM dataset are not included in this corpus. The size of the vocabulary is 44k words and the associated OOV rate on the IAM training set is 5%.

The RIMES dataset [25] consist of images of isolated handwritten French words. There are 52k running words for training and 7k running words for validation. We trained the language on the annotations of the training set, which results in a zero OOV rate.

We perceive that the IAM dataset is a harder task – because of the larger vocabulary and a high OOV rate – and use it for the purpose of evaluating our method. The RIMES dataset was used only for a comparison with the state-of-the-art, because up to our knowledge no results were published in the literature on the IAM word dataset so far.

B. Language model

We use an unigram language model with Good-Turing discounting. For the unsupervised training procedure we have reduced the vocabulary to 10k most frequent words to reduce the search overhead. The associated increase of the OOV rate to 12% on the IAM training set should not in fact pose a problem. Although we cannot discover the accurate identity of an OOV word, we can still get most of the characters right by finding a similar word in the vocabulary.

The computation of final results on the training and validation datasets is done using the largest possible vocabulary (44k). Table I shows the perplexities computed on the training and validation set using both language models. The computation of the character-level perplexities includes the word boundary after every word (even the last one) as a special symbol. Please note that the word-level perplexities can be always recomputed to give the character-level perplexities and vice-versa. The perplexities computed with the reduced vocabulary are lower because the computation does not include OOV words.

C. Experimental results

Table II shows the results on the training and the validation set. The performance of the unsupervised-trained system in comparison to the supervised-trained one is promising. It must be noted that the unsupervised training procedure cannot outperform the supervised one for the same amount of training data, because the presence of annotations is always an unbeatable advantage. In fact we did not try to beat the supervised system but wanted to demonstrate that it is possible to train a good system in an unsupervised manner. An interesting result is that the difference in error rates is much bigger on the training set than on the validation. This is because the supervised-trained system has the tendency to overfit the training data. The system trained using our method has the capability to generalize well.

Figure 2 shows the character error rate (CER) and word error rate (WER) on the training set in every iteration. In the first iteration the models are very weak, thus the improvement is not so significant. As the system improves, it can make stronger predictions, but still has a lot of space for improvement - hence the learning pace is very fast. In the end the curve converges, where the system has no capability to learn anymore. We have been however unable to reach a strict optimum because the error rate was fluctuating, even after 35 iterations. This fluctuation was however marginal (below 1% WER) and did not affect the performance of the final system. The results in Figure 2 are worse than those

presented in Table II because the recognition here includes a considerable OOV rate caused by the reduced vocabulary.

Figure 3 shows the difference between the uncovered transcriptions of the training set in succeeding iterations measured as the Levenshtein distance to the total number of words. There are two interesting extrema on this plot: in the first iterations when the system has just started learning and the predictions are inaccurate; and between the 7th and 9th iteration which corresponds to the rapid drop of the error rates. On this figure it is even more visible that we are unable to arrive at a strict optimum, as there is always some difference between succeeding transcriptions, even after 25 iterations. We used the value of 6% as a stopping criterion.

The system was able to learn 80% of the vocabulary of the training set. Figure 4 shows the number of unique words recognized in every iteration. The pace at which the system is learning new words corresponds roughly to the pace of error rate improvement. Figure 5 shows that the system was also able to learn models for almost all characters. The fluctuation in this figure is an effect of a varying transcriptions and some of the less frequent characters appearing and disappearing.

D. Discussion

Successfully predicting the length of the word is a crucial aspect of training a system in an unsupervised manner. This is a clear advantage of HMMs which have the property that the length of one character (in number of frames) is enforced in a probabilistic sense by the length of the HMM model (in number of states). Predicting a priori a most probable word of a correct length can be seen as a best joint a priori prediction of all separate characters. Successively it means that on each character position we will predict the most frequent character. During the estimation of system parameters the means will tend towards correct characters. That is why the system is able to converge into the right direction.

Another reason behind the use of HMMs is their relative speed. In our experiments we needed at least 20 iterations to obtain reasonable results, which means that this training procedure takes 20 times longer than the standard procedure of supervised training. Such a long execution time may prove prohibitive with other methods.

E. Comparison with the state-of-the-art

Table III shows the comparison of the results on the RIMES dataset. We achieve a word error rate of 10.9% in the supervised scenario and 16.9% in the unsupervised scenario, which shows that our method is a viable alternative for training a recognition system. However it is unfair to compare those two training methods directly as the presence of annotations is always an unbeatable advantage and the merits of using unsupervised learning methods go beyond obtaining good error rates. Nevertheless up to our knowledge no papers tackling the problem of training a handwriting recognition system in an unsupervised manner have been published so far in the literature.

Table III: Comparison with results reported by other groups on the RIMES validation set.

Systems	WER [%]
Supervised	10.9
Unsupervised	16.9
A2iA [18]	4.8
TUM [26]	9.0
UPV [26]	16.8
ParisTech [26]	23.7
IRISA [26]	25.3
SIEMENS [26]	26.8

Our system is a pure HMM system, yet it has been shown that an HMM system can be improved by using neural networks [19] or system combination [18]. The systems proposed by A2iA [18], UPV and ParisTech [26] were combinations of different classifiers including recurrent neural networks and HMMs. TUM [26] used a system based solely on a multi-dimensional neural network. The systems from IRISA and SIEMENS [26] were based on a single HMM with the sliding window approach.

IV. Conclusions

We have presented a proof of concept of how to train a handwriting recognition system in an unsupervised manner using only a prior language model and no annotations of the images. Surprisingly we can train the system successfully even by using a large vocabulary and in a presence of a high OOV rate. We achieve a character error rate of 13.7% on the dataset of handwritten English words which is 4% absolute worse than the baseline trained in a supervised fashion for the same amount of training data.

The use of the unsupervised learning method described in this paper is not limited to the situation where there are no annotations of the training data. It can be also used in a semi-supervised scenario when the bootstrap system is weak or unreliable. Moreover this technique allows us to use datasets orders of magnitude larger than the ones we have been using so far, which for various reasons are impossible to annotate.

Acknowledgments. This work was partially supported by a Google Research Award and by the Quaero Programme, funded by OSEO, French State agency for innovation.

REFERENCES

[1] P. Dreuw, G. Heigold, and H. Ney, "Confidence-based discriminative training for model adaptation in offline arabic handwriting recognition," in *International Conference on Document Analysis and Recognition (ICDAR)*, Barcelona, Spain, Jul. 2009, pp. 596–600.

[2] V. Frinken, M. Baumgartner, A. Fischer, and H. Bunke, "Semi-supervised learning for cursive handwriting recognition using keyword spotting," in *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Sep. 2012.

[3] V. Frinken, A. Fischer, H. Bunke, and A. Fornes, "Co-training for handwritten word recognition," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2011, pp. 314–318.

[4] C. Gollan and M. Bacchiani, "Confidence scores for acoustic model adaptation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, NV, USA, Apr. 2008, pp. 4289–4292.

[5] F. Wessel and H. Ney, "Unsupervised training for broadcast news speech recognition," in *IEEE Automatic Speech Recognition and Understanding Workshop*, Trento, Italy, Dec. 2001.

[6] G. Huang, E. Learned-Miller, and A. McCallum, "Cryptogram decoding for optical character recognition," University of Massachusetts, Amherst, MA 01003, Tech. Rep., 2006.

[7] S. Peleg and A. Rosenfeld, "Breaking substitution ciphers using a relaxation algorithm," *Commun. ACM*, vol. 22, no. 11, pp. 598–605, Nov. 1979.

[8] A. Kae and E. Learned-Miller, "Learning on the fly: Font-free approaches to difficult ocr problems," in *International Conference on Document Analysis and Recognition (ICDAR)*, 2009, pp. 571–575.

[9] T. Berg-Kirkpatrick, G. Durret, and D. Klein, "Unsupervised transcription of historical documents," in *Annual Meeting of the Assoc. for Computational Linguistics*, Sofia, Bulgaria, Aug. 2013.

[10] K. Knight, A. Nair, N. Rathod, and K. Yamada, "Unsupervised analysis for decipherment problems," in *Proceedings of the COLING/ACL on Main conference poster sessions*, 2006, pp. 499–506.

[11] S. Ravi and K. Knight, "Attacking decipherment problems optimally with low-order n-gram models," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 812–819.

[12] ——, "Bayesian inference for zodiac and other homophonic ciphers," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, 2011, pp. 239–247.

[13] M. Nuhn, J. Schamper, and H. Ney, "Beam search for solving substitution ciphers," in *Annual Meeting of the Assoc. for Computational Linguistics*, Sofia, Bulgaria, Aug. 2013, pp. 1569–1576.

[14] S. Ravi and K. Knight, "Deciphering foreign language," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 12–21.

[15] M. Nuhn, A. Mauser, and H. Ney, "Deciphering foreign language by combining language models and context vectors," in *Annual Meeting of the Assoc. for Computational Linguistics*, Jeju, Republic of Korea, Jul. 2012, pp. 156–164.

[16] M. Nuhn and H. Ney, "Decipherment complexity in 1:1 substitution ciphers," in *Annual Meeting of the Assoc. for Computational Linguistics*, Sofia, Bulgaria, Aug. 2013, pp. 615–621.

[17] H. Bunke, S. Bengio, and A. Vinciarelli, "Offline recognition of unconstrained handwritten texts using hmms and statistical language models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 6, pp. 709–720, 2004.

[18] A.-L. Bianne-Bernard, F. Menasri, R.-H. Mohamad, C. Mokbel, C. Kermorvant, and L. Likforman-Sulem, "Dynamic and contextual information in HMM modeling for handwritten word recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 10, pp. 2066–2080, 2011.

[19] M. Kozielski, P. Doetsch, and H. Ney, "Improvements in RWTH's system for off-line handwriting recognition," in *International Conference on Document Analysis and Recognition (ICDAR)*, Washington, DC, USA, Aug. 2013.

[20] M. Kozielski, J. Forster, and H. Ney, "Moment-based image normalization for handwritten text recognition," in *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Bari, Italy, Sep. 2012, pp. 256–261.

[21] N. S. J.S. Bridle, "A method for segmenting acoustic patterns, with applications to automatic speech recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, May 1977, pp. 656–659.

[22] D. Rybach, H. Ney, and R. Schlüter, "Lexical prefix tree and WFST: A comparison of two dynamic search concepts for LVCSR," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1295 –1307, Jun. 2013.

[23] U.-V. Marti and H. Bunke, "The IAM-database: an English sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, pp. 39–46, Nov. 2002.

[24] W. Francis and H. Kucera, "Brown corpus manual, manual of information to accompany a standard corpus of present-day edited American English," Tech. Rep., 1979.

[25] E. Augustin, M. Carré, G. E., J. M. Brodin, E. Geoffrois, and F. Preteux, "Rimes evaluation campaign for handwritten mail processing," in *Proceedings of the Workshop on Frontiers in Handwriting Recognition (IWFHR)*, 2006, pp. 231–235.

[26] E. Grosicki and H. El Abed, "ICDAR 2009 handwriting recognition competition," in *International Conference on Document Analysis and Recognition (ICDAR)*, Jul. 2009, pp. 1398–1402.