Improvements in Hidden Markov Model Based Arabic OCR

Rohit Prasad, Shirin Saleem, Matin Kamali, Ralf Meermeier, Prem Natarajan
BBN Technologies, Cambridge, MA 02138, USA
{rprasad, ssaleem, mkamali, rmeermeier, pnataraj}@bbn.com

Abstract

This paper describes recent advances in hidden Markov model (HMM) based OCR for machine-printed Arabic documents. A combination of script-independent and script-specific techniques are applied to glyph models and language models (LM). Script-independent techniques we applied are higher order n-gram LMs for N-best rescoring and discriminative estimation of glyph HMMs. Arabic specific techniques include the use of context-dependent HMMs for glyph modeling and Parts-of-Arabic-Words in language modeling. We present experimental results that demonstrate a 40% relative reduction in word error rate over the baseline configuration on a corpus of machine-printed Arabic documents.

1. Introduction

Most optical character recognition (OCR) systems are designed for a particular script or language. In [1] we introduced a novel approach to OCR that is script-independent and can be used for the vast majority of languages. The core feature extraction, training, and recognition components remain the same for all languages; only the data-specific components, such as the dictionary and the language model, depend on the specific language. In [2], we demonstrated the efficacy of the script-independent OCR approach by applying it to recognition of printed text in multiple languages. Recently, we demonstrated the efficacy of the same approach for multilingual offline handwriting recognition [3].

The basic modeling paradigm we employ is that of hidden Markov models (HMM) [4]. We decompose the OCR problem into a combination of two 1-D pattern recognition tasks [2]. The goal of the first task, called line finding, is to locate the individual lines of text on a page. The goal of the second task, also called the recognition task, is to extract a set of features for the line and then use these features and the glyph HMMs to generate a sequence of characters or words for the line. A word or character language model (LM) is used to constrain the search.

An advantage of the HMM based approach is that a line of text is not required to be pre-segmented into characters before recognition. The segmentation of the line into character boundaries is a byproduct of the recognition process. For connected scripts, such as Arabic and cursive handwritten text, character segmentation is non-trivial and often in-accurate in the presence of noise artifacts. The *segmentation free* nature of HMM based OCR has resulted in widespread adoption of this approach [1]-[3],[5].

In this paper we focus on improving glyph models and LMs for Arabic OCR using a combination of script-independent and Arabic-specific techniques. For glyph modeling, we explore two techniques. The first technique is the use of context-dependent HMMs for modeling shape variations in Arabic characters based on their position in the word and the adjacent characters. The second technique is the discriminative estimation of the HMM parameters. For language modeling, we explore the use of higher-order n-grams for rescoring N-best lists. In addition, we experiment with Parts-of-Arabic-Words (PAWs) as lexical units for modeling wider context than a character LM.

2. BBN Byblos OCR System

The BBN Byblos OCR system [2] can be subdivided into two functional components: training and recognition. Both components share a common pre-processing and feature extraction stage. The pre-processing starts off by first deskewing the scanned image and then locating the positions of the text lines on the deskewed image. Next, the feature extraction program computes a feature vector, which is a function of the horizontal position within the line. For feature extraction, each line of text is horizontally segmented into a sequence of thin, overlapping frames. Then, we compute a script-independent feature vector that is a numerical representation of the frame. The script-independent features include: energy, the percentile of intensities, angle, and correlation features.

The character or glyph models comprise of multi-state, left-to-right HMMs. The character HMMs are trained on transcribed text lines in the Maximum Likelihood (ML) framework using EM algorithm.

The recognition engine performs a two-pass search using both the glyph model and the LM. Typically the LM is a character or word *n*-gram estimated from a text corpus. The first pass in search uses a bigram LM to generate a lattice. The second pass uses an approximate trigram LM and optionally more detailed character HMMs to generate a 1-best hypothesis, an N-best list, or a lattice. The N-best or the lattice can be rescored with other knowledge sources.

3. Baseline Arabic OCR Experiments

For our baseline experiments, we used data from the DARPA Arabic Machine Print (DAMP) document corpus collected by SAIC [6]. The corpus consists of 297 images scanned from newspapers, books, magazines, etc. The corpus was partitioned into three sets: 60 images for development, 60 images for testing, and 177 images for training the OCR system. In addition to the 177 images from the DAMP corpus, we used 380 synthetically generated images. These 380 images were created by printing 100 newswire text passages in multiple font types and font sizes.

Arabic is a connected script where each character can be rendered in a different graphical form based on its position within a word. However, the underlying sequence of characters in the ground truth transcripts is a sequence of Unicode characters. These Unicode characters are inherently *context independent* and typically the browser or the editor renders the character in the “presentation-form” based on the neighboring characters. We refer to the context independent Unicode character as the “base-form” representation.

For training glyph HMMs, we first apply a set of transformation rules to convert the base-form character transcripts into presentation-form transcripts. Next, we estimate a separate HMM for each presentation-form character. In our baseline experiments, all HMM states of a glyph element share a single mixture of Gaussians, but the mixture weights associated with each state of a particular character is different.

From the training data, we estimated HMMs for a set of 162 presentation-form characters. This set includes Arabic characters, Arabic numerals, and some additional non-Arabic numerals and characters. A total of 68K Gaussian mixtures were estimated with a maximum of 512 Gaussian components assigned to each character in the training lexicon.

Next, we estimated both character and word LMs from transcriptions of the images from the training set as well as from 2.6 million words of Arabic newswire data. The character lexicon used for character *n*-grams consisted of all 162 characters observed in the training images. The word lexicon was restricted to the 65K most frequent words in the LM training data. Both the character and word LMs were trained using Witten-Bell discounting.

We performed recognition experiments on the test set to compare character LM to word LM. The configuration used for recognition has two steps. First, the two pass decoding described in Section 2 was used to generate an N-best list (character or word N-best depending on the type of language model). Next, we rescored the N-best list using a trigram word or character LM). The weights for combining different knowledge source in the N-best rescoring were estimated on the development set.

In Table 1, we summarize the word error rate (WER) for both the word and character LM measured on the test set. As shown in the table, the character trigram LM resulted to a WER of 12.3% compared to 15.9% obtained by using a word LM. Although a word trigram models wider context than a character trigram, in our recognition experiments the WER for the word trigram is higher than the character trigram LM. We believe this is because of the high out-of-vocabulary (OOV) rate of the word LM. The word lexicon results in an OOV rate of 12.6% on the test set, therefore the errors are dominated by OOVs.

Language Model – Trigram	%WER
Word	15.9
Character	12.3
PAW	10.1

Table 1: Comparison of different LMs for recognition on the DAMP test set.

4. Language Modeling Improvements

Higher-order LMs: For the same *n*-gram order, a character *n*-gram LM models a much narrower context than the word LM. Therefore, to model a wider *n*-gram

Character N-gram Order	%WER
n=3	12.3
n=5	11.5
n=7	11.7

Table 2: N-best rescoring experiments with character LMs with n>3.

context using characters, we estimated *n*-gram character LMs with $n > 3$. Since our two-pass decoder only supports LMs with $n<=3$, we used the higher order LMs only for rescoring the N-best list generated by decoding with a trigram character LM.

As shown in Table, 2 increasing the n-gram order to 5 decreased the WER by 0.8% absolute over using the trigram LM. However, increasing the n-gram order beyond 5 did not yield additional WER reduction.

PAW based LMs: A PAW [7] is a character sequence that is typically rendered as a single connected component in an image. PAWs can be sub-words or, words and are derived from the morpho-lexical rules of the Arabic language. An LM trained with PAWs as lexical units is likely to provide wider context at the same n-gram order than a character LM, while still preserving the unlimited vocabulary coverage property of the character LM.

We estimated a PAW trigram LM from the same training data used for character and word LMs. A total of 9K PAWs including individual characters were used as lexicon units in the PAW LM. The number of PAWs was restricted to 9K by imposing a length cutoff of 6 characters for the PAWs. The length cutoff was empirically determined on the development set by decoding with different cutoffs. Next, we decoded the test set using the PAW LM and the glyph HMMs described in Section 3. Finally, the N-best was rescored using the PAW trigram LM. As shown in the last row of Table 1, the PAW trigram LM resulted in a WER of 10.1%, a significantly lower WER than the word and the character trigram LM.

We also explored using a higher order PAW LM for N-best rescoring. However, experimentations with *n*-gram order > 3 using a PAW LM did not show any additional improvement.

5. Glyph Modeling Improvements

Context-dependent HMMs: In Arabic script the shape of a character glyph often varies based on the position of the character within a word. As described in Section 3, one approach for modeling this context dependency is to use presentation-form characters as modeling units for training context independent (CI) HMMs. Another alternative is to use base-forms to model glyphs, but instead of using CI HMMs use context-dependent (CD) HMMs.

We performed context-dependent (CD) training with both presentation form characters and base form characters. We also used two different configurations for tying HMM parameters. In the first configuration, referred to as character-tied mixture (CTM), a single mixture of Gaussians is shared by all contexts of a particular character. In the second configuration, which we refer to as position-dependent tied mixture (PDTM), a separate set of Gaussians is estimated for each state of *all* the context-dependent HMMs associated with a particular character. For example, we estimate a set of '*K*' Gaussians for the first state of all HMMs associated with the character 'Alif', and a separate set of '*K*' Gaussians for the second state of all HMMs for "Alif", and so forth.

In addition to sharing the Gaussians, we used a decision-tree based clustering of mixture weights for both the CTM and PDTM configuration. The decision-tree uses a set of questions based on different characteristics of the characters, e.g. whether the character is an ascender or a descender.

In Table 3, we compare the performance of different CD configurations with the baseline configuration described in Section 3. For fair comparison all models were configured to have approximately the same total number of Gaussian mixtures as the baseline configuration.

For comparing the different CD models, we decoded the test set using the word LM described in Section 3. As shown in Table 3, CD training using presentation form characters as modeling units did not yield any improvement over the baseline CI configuration. A possible reason for this result is that the contextual form characters by definition model glyph variations depending on the relative position of the character within a word. Thus, any additional attempt at modeling context merely fragments the training data.

Training Configuration	%WER
Pres. form, CI (baseline)	15.9
Pres. form, CD, CTM	16.9
Pres. form, CD, PDTM	16.9
Base form, CD, CTM	16.2
Base form, CD, PDTM	15.2
+ MCE training	14.1

Table 3: Decoding the test set with context-dependent HMMs and word LM.

Unlike presentation form characters, context-dependent modeling using base forms characters with PDTM tying resulted in a 0.7% absolute reduction in WER over the baseline result of 15.9% obtained using CI modeling with presentation form characters.

Discriminative Training: Traditionally, HMM parameters are estimated using ML criterion. However, in recent years discriminative training has been shown to outperform phonetic HMMs estimated using ML for speech recognition [8]. Standard ML estimation attempts to find model parameters that maximize the likelihood of the training data. In contrast, discriminative training attempts to make the correct hypothesis more probable while simultaneously making incorrect hypotheses less probable.

Similar to Minimum Phone Error (MPE) [8] used in speech, recognition, we define the following objective function to maximize for performing *Minimum Character Error* (MCE) training for glyph HMMs:

$$F_{MCE}(\lambda) = \sum_{i=1}^{N} \log \frac{\sum_h P_\lambda(O_i | M_h) P(h) CharAccuracy(h)}{\sum_h P_\lambda(O_i | M_h) P(h)}$$

In the equation above, the *CharAccuracy(h)* is a measure of the number of characters accurately generated in hypothesis h, O_i are the feature vectors, λ are the HMM parameters, is $P_\lambda(O_i|M_h)$is the glyph model score, and $P(h)$ is the LM probability.

Next, we used character lattices generated using ML glyph models and a unigram character LM to perform MCE training with base form CD HMMs with PDTM parameter tying. Extended Baum-Welch algorithm [8] was used for updating parameters and I-smoothing [8] was applied to avoid over-training. As shown in Table 3, decoding the test set with the MCE models and the word trigram LM resulted in a WER of 14.1%, a 1.1% absolute reduction in WER over the ML models.

Finally, to make use of the best glyph HMM and the best LM, we implemented the following *hybrid* recognition setup. First, we used the CI presentation form glyph HMMs estimated using ML and the PAW LM to generate an N-best list. Next, the PAW N-best was converted into a word N-best. Finally, the base form CD PDTM glyph models estimated using MCE was used to rescore the word N-best list.

Recognition Configuration	%WER
Decoding + N-best rescoring with word LM and ML CI pres. form HMMs	15.9
Decoding with PAW LM and ML CI pres. form HMMs + N-best rescoring with CD PDTM base form HMMs	9.6

Table 4: Summary of improvements in WER.

In Table 4, we compare the results from the hybrid recognition configuration above to the baseline recognition configuration with ML CI glyph HMMs and word LM. As shown in the table, we have achieved a 40% relative reduction in WER over our baseline configuration.

6. Conclusions and Future Work

In this paper, we presented a suite of techniques for significantly improving OCR performance on Arabic machine-printed documents. For glyph modeling the combination of context-dependent HMMs and discriminative training resulted in a 12% relative reduction over ML context-independent training using presentation form characters. The largest improvement for language modeling comes from the use of PAWs as lexical units for recognition. The PAW LMs outperformed both the word and language models. Since PAWs may also be better for modeling glyph characteristics for Arabic, we will explore the use of PAWs as modeling units for training glyph HMMs.

8. References

[1] J. Makhoul, R. Schwartz, C. LaPre, and I. Bazzi, “A Script-Independent Methodology for Optical Character Recognition,” *Pattern Recognition*, Vol. 31, No. 9, 1285-1294, 1998.

[2] P. Natarajan, Z. Lu, I. Bazzi, R. Schwartz, and J. Makhoul, “Multilingual Machine Printed OCR,” *International Journal of Pattern Recognition and Artificial Intelligence*, 15:1, 2001, pp. 43–63.

[3] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, K. Subramanian, “Multi-lingual Offline Handwriting Recognition Using Hidden Markov Models: A Script-Independent Approach ,” *Springer Book Chapter on Arabic and Chinese Handwriting Recognition*, ISSN: 0302-9743, Vol. 4768, pp. 231-250, March 2008.

[4] L. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” *Proc. IEEE*, Vol. 77, 257-286, 1989.

[5] A. Vinciarelli, S. Bengio, and H. Bunke, “Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models,” *IEEE PAMI*, Vol. 26, No. 6, pp. 709-720, June 2004.

[6] R.B. Davidson and R.L. Hopley, “Arabic and Persian OCR Training and Test Data Sets,” *Proc. SDIUT*, pp. 303-307, Annapolis, MD, 1997.

[7] V. Märgner, M. Pechwitz, H. El Abed, “ICDAR 2005 Arabic Handwriting Recognition Competition,” 8th ICDAR 2005, Aug. 29-Sep. 01, Seoul, Korea, 2005.

[8] D. Povey and P.C. Woodland, “Minimum Phone Error and I-Smoothing for Improved Discriminative Training,” *Proc. IEEE ICASSP’02*, Orlando, 2002.