Unsupervised HMM adaptation Using Page Style Clustering

Huaigu Cao, Rohit Prasad, Shirin Saleem, Premkumar Natarajan
BBN Technologies, Cambridge MA 02138, USA
{hcao, rprasad, ssaleem, pnataraj}@bbn.com

Abstract

In this paper we present an innovative two-stage adaptation approach for handwriting recognition that is based on clustering of similar pages in the training data. In our approach, we first perform page clustering on training data using features such as contour slope, pen pressure, writing velocity, and stroke sparseness. Next, we adapt the writer-independent Hidden Markov models (HMMs) to each cluster in the training data. While decoding a test page, we first determine the cluster the test page belongs to and then decode the page with the model associated with that cluster. Experimental results with the two-stage adaptation show significant gains on a held-out validation set.

1. Introduction

Hidden Markov Model (HMM) based speech and handwriting recognition systems, trained on large and diverse data-sets can be improved by adapting models to a small set of samples displaying similar characteristics (such as speech or handwriting by the same person). In speech recognition, speaker-dependent HMM systems trained on sufficient training data outperforms speaker-independent systems. However, it is usually impractical to collect and annotate large amounts of speech from the same speaker. Therefore, speaker-independent models are often trained using all available data from multiple speakers, and the model parameters are then re-estimated for each speaker using Maximum Likelihood Linear Regression (MLLR) or Maximum *a* Posteriori (MAP) approaches [6, 3]. Similarly for handwriting recognition, one can adapt character models to the pages written by the same author.

In real-world handwritten data, the identity of the writers is mostly unknown. Therefore, we typically perform unsupervised adaptation of the writer-independent model using the decoding results for each test page. We can also adapt the HMMs to clusters of similar pages in the test set. However, it is rarely the case that there are enough homogeneous pages in the test data. Thus, both approaches lack sufficient data for adaptation. Furthermore, the decoding results used as transcripts for adaptation are prone to recognition errors. Specifically, the word error rate of handwriting recognition is significantly higher than that of speech recognition. To mitigate the impact of recognition errors on adaptation performance, one can use a lattice for adaptation instead of using the single-best output from the first pass recognition output [13]. However, lattice-based MLLR also suffers from a lack of significant amount of adaptation data.

In speech recognition, several speaker selection methods have been explored to obtain more adaptation data [9, 5, 14, 4]. These approaches rely on selecting a subset of speakers from the training data, that are acoustically similar to a test speaker, and then adapting the HMMs on the selected speakers. The speaker similarity metrics can be defined using the likelihood of the data of the test speaker on the models of each training speaker [9, 5] and the distance between the adapted model parameters of different speakers [14, 4]. The adaptation methods used are either MAP or MLLR depending on the amount of training data available. In essence, the use of speaker clusters from training increases the amount of adaptation data and improves the recognition performance. In handwriting recognition and word spotting applications, similar techniques are used [2, 10].

This paper presents a two-stage unsupervised HMM adaptation technique for improving handwriting recognition. In the first stage, inspired by the success of writer-identification [12, 1, 11], we use features such as handwriting velocity and stroke directions to cluster similar handwritten document images. Given the page clusters from the training data, we adapt the writer-independent HMMs to each of the clusters. During recognition, each test page is mapped to the nearest cluster in training and then decoded using the corresponding adapted models for that cluster. We call the first stage "page style adaptation" instead of simply writing style adaptation because, in addition to the difference between writers, the global features also capture some characteristics of the paper (pre-printed rule-lines, noise) and different writing devices. Unlike existing approaches ([9, 5, 14, 4, 2, 10]), our clustering and writer selection

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.77

methods do not rely on the trained HMM recognizers because 1) the feature vectors in an HMM system which are usually computed per frame do not contain adequate writer-discriminative information, and 2) feature transforms like LDA and Hetroscedastic LDA eliminate the components that show the difference between writers. The second stage in our adaptation approach involves page-wise adaptation to the test pages. The "page style" adapted HMMs for each test page are adapted to the test page using the decoding hypothesis obtained from the first stage. Unlike page style adaptation, the second stage adapts more to the allograph of the test writer. In Section 4, we present experimental results that show significant reduction in word error rate (WER) for each stage of adaptation.

2 Overview of the HMM Handwriting Recognition System

We use the HMM-based Handwriting Recognition (OHR) system described in [8]. HMM OHR systems in general are well suited for modeling Arabic because they perform character segmentation implicitly as a byproduct of recognition. The OHR system can be sub-divided into two basic functional components: training and recognition. Both training and recognition have the same pre-processing and feature extraction stages.

The images are first pre-processed to remove any skew due to rotation of the text during scanning. They are then segmented into lines of text. For each line of text, percentile of the intensities, angle, correlation and energy features are computed from a sequence of overlapped windows called frames. Several features are computed for a single frame. Linear Discriminant Analysis (LDA) is used to reduce the number of features per frame. The resulting vector of LDA features is a compact numerical representation of the data in the frame. The detailed description of the feature extraction procedure is described in [7].

Handwritten text is modeled as the output of a continuous density 14-state, left-to-right HMM. The HMM for a word is formed by concatenating the HMMs for the characters in the word, and model for a line of text is formed by concatenating the models of the words in the line. Each state has an associated output probability distribution over the features, modeled by a mixture of diagonal Gaussians. The parameters of the HMM - the means and variances of the Gaussians, the weights of components of the Gaussian mixture, the state output probabilities and the transition probabilities are estimated from training data with the Expectation Maximization (EM) algorithm. The maximum likelihood estimate of the parameters of the HMM are obtained by iteratively aligning the sequence of feature vector with the sequence of character models. We typically run three iterations of the EM algorithm. For tying the Gaussians, we use a decision-tree based clustering, where the questions are based on different characteristics of the characters, e.g. whether the character is an ascender or a descender, a dotted character, punctuation, *etc*.

Depending on the amount of available training data, it may not be possible to achieve robust estimates of all the HMM parameters for all the characters. Hence, in order to reduce the total number of parameters in the system, Gaussians are shared across different character models. We train a separate set of Gaussians for each state of all the context-dependent character HMMs associated with a particular character. The configuration, referred to as Position Dependent Tied Mixture (PDTM) offers a greater number of free parameters and the possibility of better performance, subject to the availability of sufficient data for training all the parameters.

The trained character HMMs and a trigram language model (LM) are used for recognition. The use of a word lexicon during recognition generally results in a lower error rate. The lexicon is estimated from a suitably large text corpus. The language model, which provides the probability of a word sequence, is also typically estimated from the same corpus. Recognition is performed using a two-pass search strategy to find the most likely sequence of characters or words. The forward pass is an approximate but efficient procedure for generating a small list of word sequences that are possible candidates for being the most likely sequence. The backward pass is a more detailed search for the most likely word sequence within this small list. The output of recognition is an n-best list or lattice of hypotheses. The n-best list is re-ranked using a combination of the acoustic scores, and a language model score which does not model the ”white space” token. The weights for re-ranking were optimized on the development set. The optimizer works by running a non-gradient search to find the set of weights that have the lowest word error rate for a set of n-best recognition hypotheses.

3 Two-Stage Unsupervised Adaptation

As shown in Figure 1, our adaptation algorithms has two stages. The first stage performs unsupervised clustering of pages followed by adaptation of the global character HMMs on each of the page clusters. Each test page is mapped to a page cluster and decoded by the models adapted to that cluster. In the second stage, the models obtained from the first stage are further adapted to the allographs of each test page using the decoding hypothesis from the first stage.

We compute the following 9 features for each page:

- *Average width and height of the bounding boxes of the connected components in the image.* We normalize the

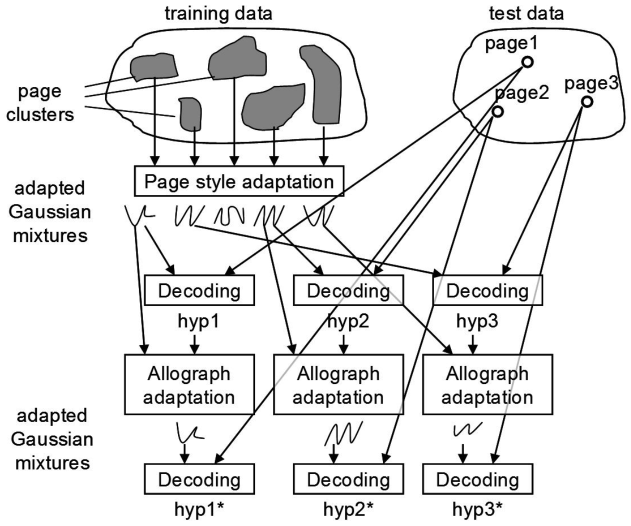

Figure 1. Diagram of the two-stage adaptation

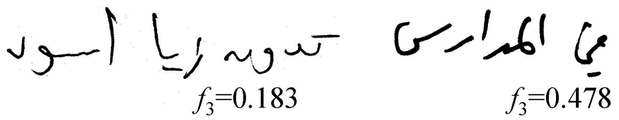

Figure 2. Handwriting samples of different pen pressures

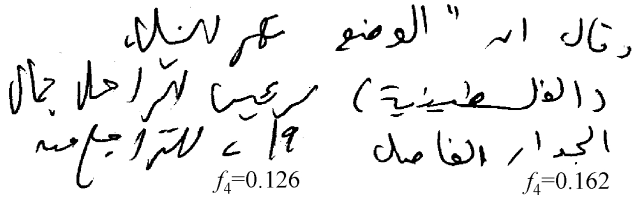

Figure 3. Handwriting samples of different text densities

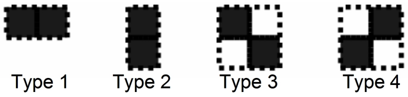

Figure 4. Four types of connectivity patterns showing different slope angles. Each black cell represents a contour pixel

2 features as follows:

$$\begin{aligned} f_1 &= \mathbf{average\_width}/(33\% \times \mathbf{dpi}) \\ f_2 &= \mathbf{average\_height}/(33\% \times \mathbf{dpi}) \end{aligned} \quad (1)$$

where **dpi** (dot per inch) is the image resolution.

- *Pen pressure and width of strokes* is represented by the following feature (Figure 2):

$$f_3 = 1 - \frac{\textbf{\# contour pixels}}{\textbf{\# black pixels}} \quad (2)$$

- *Text density*. We compute text density as follows:

$$f_4 = \frac{\textbf{\# contour pixels}}{\textbf{\# pixels in bounding boxes of connected components}} \quad (3)$$

Handwriting with low text density tends to have more exaggerated ascenders and descenders (Figure 3).

- *Contour slope features and external/internal contours [12]* The overall slope along the contours of text shows writer-specific features of slope and round/sharp turns in the handwriting. It also detects ruled-lines in the image. We first count the numbers of occurrences of the four connectivity patterns of contour pixels shown in Figure 4 and then normalize the four numbers by the total number:

$$\begin{aligned} f_5 &= \frac{\textbf{\# Type 1 connectivity patterns}}{\textbf{\# Types 1-4 connectivity patterns}} \\ f_6 &= \frac{\textbf{\# Type 2 connectivity patterns}}{\textbf{\# Types 1-4 connectivity patterns}} \\ f_7 &= \frac{\textbf{\# Type 3 connectivity patterns}}{\textbf{\# Types 1-4 connectivity patterns}} \\ f_8 &= \frac{\textbf{\# Type 4 connectivity patterns}}{\textbf{\# Types 1-4 connectivity patterns}} \end{aligned} \quad (4)$$

The writing movement characteristics such as velocity and connectedness are indicated by the number of external and internal contours (Figure 5). In general, cursive handwriting has fewer internal contours than neat handwriting. We compute the ratio between the number of internal contours and the total number of contours as a feature.

$$f_9 = \frac{\textbf{\# internal contours}}{\textbf{\# internal and external contours}} \quad (5)$$

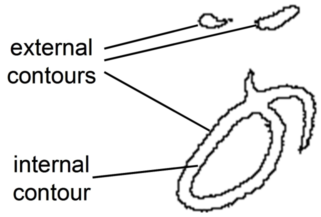

Figure 5. External and internal contours

We compute the nine features $f_1, ..., f_9$ for each page in the training set and run K-means clustering using these features. Each test page is mapped to the cluster with the nearest mean value. We test both MAP and MLLR algorithms for page-cluster-level adaptation (stage I) and MLLR for page-level adaptation (stage II).

4 Experimental Results

4.1 Data Corpus

For our experiments, we used scanned images of handwritten Arabic text of newswire articles, web log posts and newsgroup posts. A total of 8250 pages by 58 different authors were used for training the HMM system. Additionally, a set of 218 pages comprising of 109 pages written by 6 authors in the training set and 109 pages written by 6 new authors who were never seen in training were used as the development set. A separate set of 403 images by 26 authors never seen in training were used as the validation set. The number of words in the validation set is 42762.

The OCR recognition engine was trained using all available training data. The ground truth annotations for the images included the coordinates of bounding boxes around individual words and the corresponding tokenized transcriptions. Features were extracted on individual words in the image, and then concatenated to form line-level features which were used for training and recognition. Therefore, only the top and bottom locations from the word segmentation were used for feature extraction for a piece-wise linear approximation of the envelope for the text line. PDTM context dependent HMM models were trained for a total of 176 unique characters. This set includes Arabic characters, Arabic numerals, punctuations and some additional non-Arabic numerals and characters. A total of 254K Gaussian mixtures were estimated. The trigram language model was trained on 90 million words of Arabic newswire data. The size of the decoding word lexicon is 92,000. The ground-truth transcriptions were processed such that all punctuations and digits were stripped from the preceding/following character for the computation of the word error rate.

We obtained 4 groups of page clusters by selecting different numbers of clusters ($N = 8, 32, 128, 512, 1024$) in the K-means clustering step. On each group of page clusters, we ran page cluster adaptation on our training data using two standard adaptation algorithms (MLLR and MAP).

4.2 Writer Identification Results

First, we evaluated the features presented in Section 3 by performing a writer identification experiment. We randomly selected 80% of the pages in the OCR training corpus as the training set for writer identification. The remaining 20% of images was used as the test set for writer identification. We trained one 2-class Support Vector Machine (SVM) classifier with RBF kernels for every two writers. Next, the test pages were classified using all the SVM classifiers. Each time a classifier was used, the winning class got a vote. The class that got the most votes was taken as the identified writer class. The described approach resulted in a 77.8% top-choice accuracy on 58 writers. This demonstrates that the features for clustering are effective in identifying writers.

4.3 Adaptation Results

We ran decoding experiments on the development set using the models obtained from stage I adaptation. The performance of the un-adapted and adapted models are compared in Table 1. Consistent improvement of WER was obtained for both writers in the training set and writers not in the training set using MLLR adaptation. Thus the MLLR adaptation is not biased toward the writers in the training set. The MAP adaptation of 8 clusters and 32 clusters shows better performance on the writers in the training set than the MLLR approach but shows worse performance on writers not in the training set. This indicates that the MAP adapted models are over-fitting to the adaptation data. As one would expect, the amount of adaptation data decreases as more page clusters are used (128, 512 and 1024), resulting in worse performance of MAP adaptation.

Table 1. Word error rate (WER) on the development set using unadapted models and models adapted to page clusters

Author Set	In Training	Not in Training
Unadapted	44.9%	36.7%
8-Cluster/MLLR	43.0%	34.6%
32-Cluster/MLLR	42.7%	34.2%
128-Cluster/MLLR	42.2%	33.7%
512-Cluster/MLLR	42.0%	34.0%
1024-Cluster/MLLR	42.2%	33.9%
8-Cluster/MAP	42.2%	34.8%
32-Cluster/MAP	41.9%	34.9%
128-Cluster/MAP	42.1%	35.2%
512-Cluster/MAP	43.7%	36.4%
1024-Cluster/MAP	44.1%	37.1%

We also evaluated the performance of combining stages I and II adaptation on the held-out validation set. The n-best lists of hypotheses from the decoder for each test page were re-ranked using the weights estimated from the development set. For comparison we also tested our system

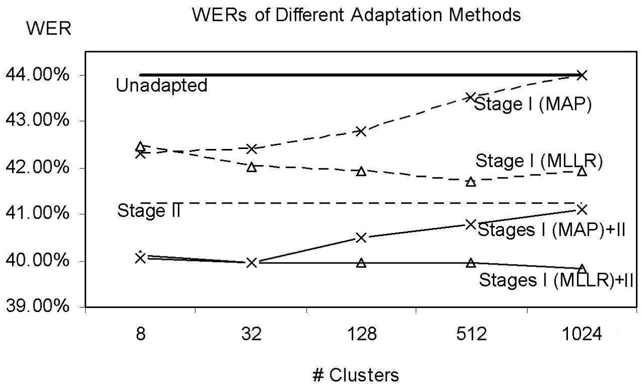

Figure 6. Adaptation results on the validation set

by using the page-level adaptation (stage II) but not using the page cluster adaptation. This is the traditional unsupervised adaptation, widely used in most systems[6]. As can be seen in Figure 6, MLLR works better than MAP with 128, 512, and 1024 page clusters and performs almost the same with 8 and 32 page clusters. This is similar to what we observed from the results on development set. The lowest overall WER of combining page cluster adaptation and page-level adaptation is 39.8%, which was obtained from MLLR page cluster adaptation on 1024 page clusters. In contrast, the WER of un-adapted decoding is 44.0% and the WER of page-level adaptation (stage II) is 41.2%. Both MLLR and MAP page cluster adaptation produce significant gains when combined with page-level adaptation.

5 Conclusion

We presented a two-stage HMM adaptation method for handwriting recognition. The first stage clusters training pages with similar page styles and adapts global HMMs to each cluster. The second stage performs adaptation on the test page itself. Our experimental results show that both stages lead to significant reduction in WER. Moreover, the overall reduction of WER when combining the two stages is close to the sum of the reduced WERs by applying two stages separately. Future work will involve investigating allograph-level clustering algorithms, applying clustering to writer adaptive training (WAT), and combining WAT and adaptation for greater overall performance.

References

[1] M. Bulacu and L. Schomaker. Text-independent writer identification and verification using textural and allographic features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):701–717, 2007.

[2] G. Fink and T. Plötz. Unsupervised estimation of writing style models for improved unconstrained off-line handwriting recognition. In *Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition*, 2006.

[3] J. Gauvain and C. Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech and Audio Processing*, 2:291–298, 1994.

[4] R. Gomez, T. Toda, H. Saruwatari, and K. Shikano. Improving rapid unsupervised speaker adaptation based on Hmm sufficient statistics. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2006.

[5] C. Huang, T. Chen, and E. Chang. Speaker selection training for large vocabulary continuous speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 609–612, 2002.

[6] C. Leggetter and P. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(2):171–185, 1995.

[7] 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):43–63, 2001.

[8] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, and K. Subramanian. Multi-lingual offline handwriting recognition using hidden Markov models: A script-independent approach. *Springer Book Chapter on Arabic and Chinese Handwriting Recognition*, 4768:231–250, 2008.

[9] M. Padmanabhan, L. Bahl, D. Nahamoo, and M. Picheny. Speaker clustering and transformation for speaker adaptation in speech recognition systems. *IEEE Transactions on Speech and Audio Processing*, 6(1):71–77, 1998.

[10] J. A. Rodriguez, F. Perronnin, G. Sanchez, and J. Llados. Unsupervised writer style adaptation for handwritten word spotting. In *Proceedings of the International Conference on Pattern Recognition*, 2008.

[11] L. Schomaker. Advances in writer identification and verification. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, volume 2, pages 1268–1273, 2007.

[12] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee. Individuality of handwriting. *Journal of Forensic Sciences*, 47(4):1–17, 2002.

[13] L. Uebel and P. Woodland. Speaker adaptation using lattice-based MLLR. In *Proceedings of the ISCA Tutorial and Research Workshop (ITRW) on Adaptation Methods for Speech Recognition*, 2001.

[14] S. Yoshizawa, A. Baba, K. Matsunami, Y. Mera, M. Yamada, and K. Shikano. Unsupervised speaker adaptation based on sufficient HMM statistics of selected speakers. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 341–344, 2001.