

# Optimizing the Number of States, Training Iterations and Gaussians in an HMM-based Handwritten Word Recognizer

Simon Günter and Horst Bunke

Department of Computer Science, University of Bern

Neubrückestrasse 10, CH-3012 Bern, Switzerland

E-mail: {sguenter, bunke}@iam.unibe.ch}

## Abstract

*In off-line handwriting recognition, classifiers based on hidden Markov models (HMMs) have become very popular. However, while there exist well-established training algorithms, such as the Baum-Welch procedure, which optimize the transition and output probabilities of a given HMM architecture, the architecture itself, and in particular the number of states, must be chosen “by hand”. Also the number of training iterations and the output distributions need to be defined by the system designer. In this paper we examine some optimization strategies for an HMM classifier that works with continuous feature values and uses the Baum-Welch training algorithm. The free parameters of the optimization procedure introduced in this paper are the number of states of a model, the number of training iterations, and the number of Gaussian mixtures for each state. The proposed optimization strategies are evaluated in the context of a handwritten word recognition task.*

**Keywords:** handwritten word recognition, hidden Markov model (HMM), training strategy, state number optimization.

## 1 Introduction

The field of off-line handwriting recognition has been a topic of intensive research for many years. First only the recognition of isolated handwritten characters was investigated [12], but later whole words [11] were addressed. Most of the systems reported in the literature until today consider constrained recognition problems based on vocabularies from specific domains, e.g. the recognition of handwritten check amounts [3] or postal addresses [4]. Free handwriting recognition, without domain specific constraints and large vocabularies, was addressed only recently in a few papers [5, 7].

Hidden Markov Model classifiers (HMMs) have become very popular in the domain of handwritten word recognition

[4, 7, 14]. Given an HMM with a predefined architecture, there exist some well-established training algorithms to automatically optimize the parameters of that architecture. An example is the Baum-Welch training procedure [9] which uses the Maximum Likelihood Estimation (MLE) criterion. However, the architecture of an HMM as well as the number of Gaussian mixtures per state and the number of training iterations are usually empirically determined. In the speech recognition domain there are some papers that address the optimization of the free parameters of an HMM architecture. In [8] the number of Gaussian mixtures of the states is optimized by repeatedly splitting Gaussian distributions according to the Maximum Mutual Information Estimation (MMIE) criterion, which is also used for training in this paper. Also in [10] the number of Gaussian mixtures is optimized. Here in each step the Gaussian distribution with the highest variance is split. Before splitting, Gaussian distributions that have too few training samples and are similar to each other are merged to avoid the problem of overfitting. In [13] a combined optimization strategy for the number of states and the number of Gaussian mixtures per HMM is presented. A disadvantage of this method is that it doesn't preserve the linearity of the models.

In the present paper we propose some optimization strategies for HMM classifiers using Baum-Welch training. These optimization strategies are evaluated in a number of experiments with a handwritten word recognizer. The parameters of the HMM subject to optimization are the number of training iterations and the number of Gaussian mixtures per state. In addition we address the problem of finding the optimum number of states per HMM.

The rest of the paper is organized as follows. Section 2 describes the handwritten word recognizer and the recognition task. In Section 3 methods for finding the number of states of an HMM are discussed, and in Section 4 a description of the strategies to jointly optimize the number of Gaussian mixtures and training iterations is given. Experiments are then discussed in Section 5 and, finally, conclusions are drawn in Section 6.

## 2 Handwritten word recognizer

The application considered in this paper is the off-line recognition of cursively handwritten words. As basic classifier an HMM-based recognizer is used. This recognizer is similar to the one described in [7]. We assume that each handwritten word input to the recognizer has been normalized with respect to slant, skew, baseline location and height (for details of the normalization procedures see [7]). A sliding window of one pixel width is moved from left to right over the word and nine geometric features are extracted at each position of the window. Thus an input word is converted into a sequence of feature vectors in a 9-dimensional feature space. The geometric features used in the system include the fraction of black pixels in the window, the center of gravity, and the second order moment. These features characterize the window from the global point of view. The other features give additional information. They represent the position of the upper and lowermost pixel, the contour direction at the position of the upper and lowermost pixel<sup>1</sup>, the number of black-to-white transitions in the window, and the fraction of black pixels between the upper and lowermost black pixel. In [7] a more detailed description of the feature extraction procedures can be found.

For each uppercase and lowercase character, an HMM is build. For all HMMs the linear topology is used, i.e. there are only two transitions per state, one to itself and one to the next state. The character models are concatenated to word models. There is exactly one model for each word from the underlying dictionary. This approach makes it possible to share training data across different words. That is, each word in the training set containing character  $x$  contributes to the training of the model of  $x$ . Thus the words in the training set are more intensively utilized than in the case where an individual model is build for each word as whole, and characters are not shared across different models.

The implementation of the system is based on the Hidden Markov Model Toolkit (HTK), which was originally developed for speech recognition [15]. This software tool employs the Baum-Welch algorithm for training and the Viterbi algorithm for recognition [9]. The output of the HMM classifier is the word with the highest rank among all word models together with its score value.

The increase of the number of Gaussian mixtures in a state is done in HTK by splitting the Gaussian distribution with the highest weight, i.e. the most training samples [15]. The mean vectors of the two new Gaussian distributions are the mean of the original Gaussian plus, respectively minus, 0.2 times the standard deviation of the original distribution. If the number of Gaussian mixtures is to be increased by more than one then several sequential splittings are ex-

<sup>1</sup>To compute the contour direction, the windows to the left and to the right of the actual window are used.

cuted where the distributions with more training samples are split first.

For the experiments, words from the handwritten sentence database described in [6, 17] were used. Training and test sets contain 9861 and 3850 words, respectively, and a validation set of 1066 words was used. The considered recognition task is quite challenging. The number of writers contributing to training, validation and test set is larger than 80. The sets of writers of the training, validation set and test set are pairwise disjoint, i.e. all experiments described in this paper are writer independent. The vocabulary underlying the data set used in the experiments is 2296. That is, a pattern recognition problem with 2296 classes is considered. A few sample words used in the experiments are shown in the Appendix.

## 3 Finding the number of states

The basic version of the system described in Section 2 employs a single Gaussian distribution for each state. This system is trained with four iterations of the Baum-Welch procedure. This number of training iterations has been found to be optimal for a large range of state numbers. To find the optimum number of states of the HMMs two approaches were examined in this paper:

**Constant number:** The number of states of each HMM is set to a constant  $s$ . The best value of  $s$  is determined using the validation set. For the system described in Section 2 the optimal value of  $s$  is 14<sup>2</sup>. The performance of the base system using 14 states per HMM is 66.23 % on the validation set. This approach is denoted as *const-state* in the following.

**Flexible number:** The Bakis method introduced in [2] is applied where the number of states of each HMM is set to the average length of the corresponding sequence of feature vectors times a constant  $f$ . For the classifier described in the last section this average length is equal to the average width of a character. The average width of the characters is determined by running the HMM recognizer in “forced alignment” mode on the training set. In this mode the recognizer uses the label information of the data to find the character boundaries. For further details see [16]. Using the system described in Section 2 a value of 0.36 was found to be optimal for  $f$ . A recognition rate of 70.92 % on the validation set was achieved with this value. This approach is denoted as *var-state* in the following.

Because parameters  $s$  and  $f$  were optimized for a single Gaussian distribution for each state, it is not guaranteed that the *var-state* method is also superior to *const-state* when Gaussian mixtures are used instead of single Gaussian distributions. Therefore, both methods have been tested in con-

<sup>2</sup>In HTK always two non-emitting states are included in the HMM, so the optimal number of states is 16 in HTK.

junction with the optimization methods introduced in the next section.

#### 4 Joint optimization of the number of Gaussian mixtures and training iterations

Given an HMM with a number of states optimized by means of one of the two methods described in Section 3, we now turn to the problem of jointly optimizing the number of training iterations and the number of Gaussians per state. Three strategies for this task are introduced in this section:

**Strategy 1:** The algorithm underlying the first strategy is shown in Table 1. After each increase of the number of Gaussian mixtures a fixed number,  $C$ , of training iterations are performed. When the desired number of Gaussian mixtures is reached, the classifier is tested on the validation set immediately after each training cycle to find the optimal number of training iterations for the final system, i.e. the system with the desired number of Gaussians. The idea of this approach is that the system should adapt itself after an increase of the number of Gaussian mixtures to the new configuration. For the adaption a fixed number of training cycles are executed.

**Strategy 2:** The second strategy is identical to the first one when  $C=0$ . This means that the number of Gaussian mixtures is increased up to the desired number without any training iterations after an increase. The second part, i.e. finding the optimal number of training iterations, is the same as under the first strategy. In this approach it is assumed that it is not necessary to adapt the system immediately after each increase of the number of Gaussian mixtures.

**Strategy 3:** The algorithm for the third strategy is shown in Table 2. In this strategy the optimal number of training iterations is searched after each increase of the number of Gaussian mixtures, which means that the system is optimized immediately after a new Gaussian has been added.

In each strategy the desired number of Gaussians, GAUSSES, must be defined. To find the best training method the algorithms must be executed with a suitable range of values for GAUSSES, which is application dependent. For the application considered in this paper a range from 5 to 9 was used. The value of GAUSSES with the best recognition result is chosen for the final classifier. In addition to GAUSSES the constants MINITER and MAXITER must be set. For the first and second strategy MAXITER was 60, whereas it was set to 20 for the third strategy. The value of MINITER was equal to 5. For the first strategy there is also the parameter  $C$ . In the experiments described in the next section the strategy was tested with  $C = 2, 3, 4$ . In Fig 1 the structure of the training methods corresponding to each strategy is shown.

```

train the base classifier  $B$ ;
for( $i = 2; i < \text{GAUSSES}; i++$ )
    increase number of Gaussian mixtures of  $B$  by 1;
    train  $B$  with  $C$  iterations;
increase number of Gaussian mixtures of  $B$  by 1;
 $B'$  is a copy of  $B$ ;
for( $i = 1; i \leq \text{MAXITER}; i++$ )
    train  $B'$  with 1 iteration;
    test  $B'$  on the validation set;
    if( $B'$  is the best tested classifier yet)
         $\text{best\_iterations} = i$ ;
train  $B$  with  $\text{best\_iterations}$  iterations;
return  $B$ ;

```

**Table 1. First strategy for optimizing of the number of training iterations**

```

train the base classifier  $B$ ;
for( $i = 2; i \leq \text{GAUSSES}; i++$ )
    increase number of Gaussian mixtures of  $B$  by 1;
     $B'$  is a copy of  $B$ ;
    for( $j = \text{MINITER}; j \leq \text{MAXITER}; j++$ )
        train  $B'$  with 1 iteration;
        test  $B'$  on the validation set;
        if( $B'$  is the best tested classifier yet)
             $\text{best\_iterations} = j$ ;
    train  $B$  with  $\text{best\_iterations}$  iterations;
return  $B$ ;

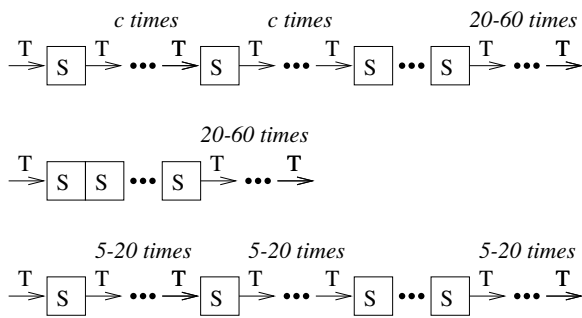
```

**Table 2. Third strategy for optimizing of the number of training iterations**

The above mentioned limitations of the search space were based on results of previous experiments using smaller sets of words.

## 5 Experiments

The three optimization strategies of the last section were applied to find the optimal combination of the number of Gaussians per state and training iterations for each of the two procedures described in Section 3. Classifiers trained under any of the three strategies were then tested on a test set of 3850 words, which is disjoint to the training and validation set. The base system was built as described in [6]. It uses a single Gaussian distribution for each state of all HMMs and is trained with four training iterations. The recognition rate of this system is 70.91 % for *const-state* and 72.46 % for *var-state*. Note that the performance difference between *var-state* and *const-state* has dropped from 4.76 % on the validation set to 1.55 % on the test set. The



**Figure 1. Structure of the training methods corresponding to the first (top), second (middle) and third (bottom) strategies. “T” denotes a training and “S” a splitting operation.**

reason for this drop is that parameter  $c$  was optimized on the validation set and is therefore suboptimal for the test set.

Table 3 shows a summary of the experimental results. The entry in column *state* indicates which method was used to determine the number of states. The optimization strategy for the number of training iterations and Gaussians is shown under *strategy*. In the column *training iterations* the number of training iterations between each increase of the number of Gaussian mixtures are given (separated by commas). The final number of Gaussian mixtures is shown in column *gauss*. The performance of the optimized classifier on the validation and test set is shown in columns *valid* and *test*, respectively.

For all strategies the system with a variable numbers of states did better on the validation and also on the test set than the system with a fixed number of states per HMM. The overall best result on the test set of 82.13 % was achieved using the first training strategy. Yet, the problem with this strategy is that the best value for the constant  $C$  is a priori not known. Considering the results on the validation set, the best value for  $C$  is 3, yet for that value the results on the test set are the lowest. For the second and third strategy no such constants must be determined. The results produced by the third strategy are quite good: Among all systems with a fixed state number it reached the highest recognition rate, and the version with a variable number of states was better than the second strategy. In summary it may be stated that the first strategy is able to produce superior results when the optimal value for parameter  $C$  is known, whereas the third strategy doesn't have such a parameter and also produces quite good results.

## 6 Conclusions

Two methods to optimize the number of states and three strategies to find the optimal number of training iterations and Gaussian mixture models for an HMM classifier are

proposed in this paper. They were evaluated in the context of a handwritten word recognition task and compared to a base system that uses only a single Gaussian distribution for each state of the HMM. For the experiments three sets were used: a training set, used to find the transition and output probabilities of the individual HMMs, a validation set, used to find the optimal number of states, training iterations and Gaussian mixture models, and a test set on which the optimized classifiers were evaluated.

The results of the experiments show that the more sophisticated strategy for setting the number of states of the HMMs works well together with the training strategies. Out of the three training strategies, two were found to be promising. The strategy which needs the setting of an additional parameter produced the best result. An improvement of the recognition rate from 72.46 % to 82.13 % was achieved on the test set under this method. The other promising strategy is easier to apply because it doesn't have such a parameter. Nevertheless it was able to produce a classifier with a performance of 80.93 %. From the experimental results it can be concluded that the optimization methods proposed in this paper are potentially useful to improve the performance of HMM-based handwriting recognition systems.

An open problem is the rather restricted ability of the validation set to represent the test set. In most of the tests the performance of a classifier on the validation set didn't correspond well with the performance on the test set. We will address this problem in our future research by applying cross-validation schemes.

## Acknowledgment

The research was supported by the Swiss National Science Foundation (Nr. 20-52087.97). The authors thank Dr. Urs-Victor Marti for providing the handwritten word recognizer and Matthias Zimmermann for the segmentation of a part of the IAM database. Additional funding was provided by the Swiss National Science Foundation NCCR program “Interactive Multimodal Information Management (IM)<sup>2</sup>” in the Individual Project “Scene Analysis”.

## References

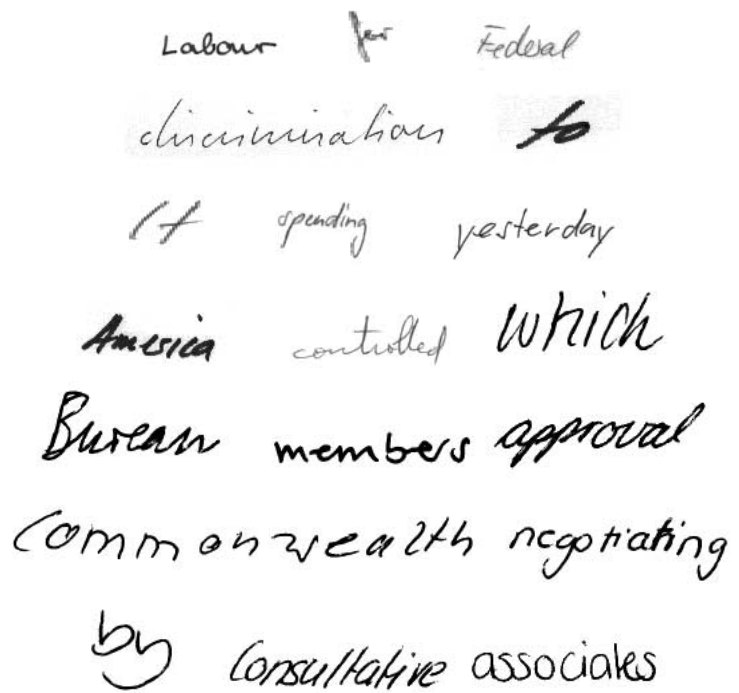
- [1] *Proc. of the 8th International Workshop on Frontiers in Handwriting Recognition*, Niagara-on-the Lake, Ontario, Canada, 2002.
- [2] R. Bakis. Continuous speech recognition via centisecond acoustic states. In *91. Meeting of the Acoustic Society of America*, 1976.
- [3] S. Impedovo, P. Wang, and H. Bunke, editors. *Automatic Bankcheck Processing*. World Scientific Publ. Co, Singapore, 1997.
- [4] A. Kaltenmeier, T. Caesar, J. Gloger, and E. Mandler. Sophisticated topology of hidden Markov models for cursive

state	strategy	training iterations	gauss	valid	test
const.	1, $C = 2$	4,2,2,2,2,2,13	7	78.33 %	80.30 %
var.	1, $C = 2$	4,2,2,2,2,2,11	7	79.74 %	81.02 %
const.	1, $C = 3$	4,3,3,3,3,32	6	78.71 %	79.04 %
var.	1, $C = 3$	4,3,3,3,19	5	80.58 %	79.18 %
const.	1, $C = 4$	4,4,4,4,4,57	6	78.14 %	79.07 %
var.	1, $C = 4$	4,4,4,4,4,10	6	80.21 %	82.13 %
const.	2	4,0,0,0,0,45	6	76.17 %	77.35 %
var.	2	4,0,0,0,0,34	6	79.27 %	80.47 %
const.	3	4,12,14,15,15,8	6	78.14 %	80.43 %
var.	3	4,11,12,17,11,16,12	7	80.86 %	80.93 %

**Table 3. The optimized training methods and the performance of the trained classifiers**

- script recognition. In *Proc. of 2nd Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan*, pages 139–142, 1993.
- [5] G. Kim, V. Govindaraju, and S. Srihari. Architecture for handwritten text recognition systems. In S.-W. Lee, editor, *Advances in Handwriting Recognition*, pages 163–172. World Scientific Publ. Co., 1999.
- [6] U. Marti and H. Bunke. A full English sentence database for off-line handwriting recognition. In *Proc. of 5th Int. Conf. on Document Analysis and Recognition, Bangalore, India*, pages 705–708, 1999.
- [7] U.-V. Marti and H. Bunke. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. *Int. Journal of Pattern Recognition and Art. Intelligence*, 15:65–90, 2001.
- [8] Y. Normandin. Optimal splitting of HMM Gaussian mixture components with MMIE training. In *Proceeding of the International Conference on Acoustic Speech Signal Processing*, pages 449–452, Detroit, 1995.
- [9] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–285, 1989.
- [10] A. Sankar. Experiments with a Gaussian merging-splitting algorithm for HMM training for speech recognition. In *Proceedings of the 1997 DARPA Broadcast News Transcription and Understanding Workshop*, pages 99–104, 1998.
- [11] J.-C. Simon. Off-line cursive word recognition. *Proc. of the IEEE*, 80(7):1150–1161, 1992.
- [12] C. Suen, C. Nadal, R. Legault, T. Mai, and L. Lam. Computer recognition of unconstrained handwritten numerals. *Proc. of the IEEE*, 80(7):1162–1180, 1992.
- [13] M. Takano. Unified state/mixture topology design via multi path HMM by ML greedy state split. In *International workshop on Automatic Speech Recognition and Understanding (ASRU)*, 1999.
- [14] W. Wang, A. Brakensiek, and G. Rigoll. Combination of multiple classifiers for handwritten word recognition. In [1], pages 117–122.
- [15] S. J. Young, J. Jansen, J. J. Odell, D. Ollason, and P. C. Woodland. *The HTK Hidden Markov Model Toolkit Book*. Entropic Cambridge Research Laboratory, <http://htk.eng.cam.ac.uk/>, 1995.
- [16] M. Zimmermann and H. Bunke. Hidden markov model length optimization for handwriting recognition systems. In [1], pages 369–374.
- [17] M. Zimmermann and H. Bunke. Automatic segmentation of the IAM off-line database for handwritten English text. In *Proc. of 16th Int. Conference on Pattern Recognition*, volume 4, pages 35–39, Quebec, Canada, 2002.

## Appendix



**Figure 2. Sample of words from the underlying database.**