

SCHMMS IN DIGIT DIALING

Janne Suontausta, Marcel Vasilache, and Kari Laurila

Nokia Research Center, Speech and Audio Systems Laboratory,
P.O. Box 100, FIN-33721 Tampere, Finland

Email: {janne.suontausta, marcel.vasilache, kari.laurila}@research.nokia.com

ABSTRACT

Hidden Markov models are a powerful tool for modeling speech signals. In the literature, several types of HMMs with their own advantages have been proposed. In this paper, the semi-continuous hidden Markov models (SCHMMs) are compared to the continuous density hidden Markov models (CDHMMs) in a noise robust connected digit recognition task. In the paper, we use a two-step training approach for SCHMMs. First, the codebook of densities and the initial model parameters are estimated, and finally discriminative training (DT) is applied. The CDHMMs are trained with the maximum likelihood (ML) method followed by DT. The experimental results indicate that the SCHMMs have similar performance as the CDHMMs with the same amount of parameters.

1. INTRODUCTION

Continuous density hidden Markov models have proven to be a powerful tool for modeling speech signals. In CDHMMs, each state of the hidden Markov model is associated with a continuous output probability distribution. The output probability distributions are usually composed of Gaussian mixtures. In CDHMMs the Gaussian mixtures are given for each state independently of the other states.

Semi-continuous hidden Markov models can be regarded as continuous density hidden Markov models, where a codebook of output distribution functions is shared by all the states in all the SCHMMs [1,2]. SCHMMs have certain advantages over CDHMMs. Due to the codebook approach, SCHMMs can have a large number of mixture densities in one state and they can be trained with a limited amount of training data.

In [1] SCHMMs and their training algorithms are discussed. SCHMMs are compared against the discrete hidden Markov models (DHMMs) and CDHMMs in a phoneme recognition task when the complexity of all the model types is comparable. SCHMMs were applied to speaker-independent continuous speech recognition in [2] and the comparison against CDHMMs and DHMMs favored SCHMMs. In [3] SCHMMs were successfully used in connected digit recognition. In [1,2,3], the recognition tests were done in the clean environment only.

In the case of multi-mixture CDHMMs, the computational complexity increases as the number of mixtures per state increases. For SCHMMs we have always a large number of mixtures per state, but the computational complexity is primarily determined by the size of the density codebook. In this paper, the objective is to achieve good noise-robust recognition performance with low complexity by using SCHMMs. The training of SCHMMs is discussed and the performance of SCHMMs is compared to the performance of CDHMMs in the connected digit recognition task.

This paper is organized as follows. The structure of SCHMMs is briefly discussed in Section 2. In Section 3 the training algorithms for SCHMMs are presented. The training methods include both the initial training and the discriminative training algorithms. The simulation experiments are given in Section 4. The performance of SCHMMs is tested on the noise corrupted TIDIGITS database. The simulations are done with clean data and with noise corrupted data with two SNR values (0 and -10 dB). The performance of SCHMMs is compared to the performance of CDHMMs. Finally, conclusions are given in Section 5.

2. SEMI-CONTINUOUS HMMS

The CDHMM λ having N states is specified by the state transition matrix $A = (a_{ij})_{i,j=1,\dots,N}$, the continuous output probability density functions $b_j(\mathbf{x})$, where $j = 1, 2, \dots, N$, and the initial state probability distribution π_{s_0} . For a sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ of acoustic observations (feature vectors) the probability of observing the sequence given the model λ is computed from

$$P(\mathbf{X}|\lambda) = \sum_S \pi_{s_0} \prod_{t=1}^T a_{s_{t-1}s_t} b_{s_t}(\mathbf{x}_t), \quad (1)$$

where the summation is taken over all possible state sequences. The output probability density function of the CDHMM having K Gaussian mixtures per state is given by

$$b_j(\mathbf{x}_t) = \sum_{k=1}^K c_{jk} N(\mathbf{x}_t | \mu_{jk}, \Sigma_{jk}), \quad (2)$$

where $N(\mathbf{x}_t|\mu_{jk}, \Sigma_{jk})$ is the Gaussian density function having mean vector μ_{jk} and covariance matrix, Σ_{jk} and c_{jk} is the weight of the k 'th Gaussian density in state j .

In the case of SCHMMs [1,2], the Gaussian density functions $N(\mathbf{x}_t|\mu_{jk}, \Sigma_{jk})$ are shared by all the states in all the models. Thus, we have a codebook of density functions $N(\mathbf{x}_t|\mu_k, \Sigma_k)$, where the total number of density functions is $M \gg K$. In the case of SCHMMs, the index k in equation (2) goes from 1 to M . Only the weights c_{jk} of the Gaussian density functions in equation (2) are model dependent parameters.

3. TRAINING OF SCHMMs

In [1] the training of SCHMMs is done by starting with a VQ codebook for observation vectors and Baum-Welch trained DHMMs. The training consists of mutual estimation of the VQ codebook and SCHMMs which were constructed from DHMMs.

In this paper the training of SCHMMs proceeds in two steps. First, the parameters of SCHMMs are initialized with a maximum likelihood clustering method, and in the second step the parameters are re-estimated with a discriminative training algorithm.

3.1. Initial training

Initial training is done in two parts. First the codebook of density functions is generated, after that only the model specific parameters are estimated.

The training of SCHMMs starts with the generation of the codebook of density functions. When generating the codebook, the data are assumed to be generated by a mixture of Gaussian density functions, see [4] for detailed discussion. Thus, the probability density function of the data sample \mathbf{x} , $p(\mathbf{x}|\mu, \Sigma)$, is given by

$$p(\mathbf{x}|\mu, \Sigma) = \sum_{j=1}^M N(\mathbf{x}|\mu_j, \Sigma_j) P(\omega_j), \quad (3)$$

where M is the codebook size, $\mu = [\mu_1, \dots, \mu_M]$, $\Sigma = [\Sigma_1, \dots, \Sigma_M]$, and $P(\omega_j)$ is the a priori probability of the Gaussian density function $N(\mathbf{x}|\mu_j, \Sigma_j)$.

The parameters $\mu_j, \Sigma_j, P(\omega_j)$ are estimated according to the maximum likelihood principle. Given the observations $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ the likelihood function that is to be maximized is

$$p(\mathbf{X}|\mu, \Sigma) = \prod_{t=1}^T p(\mathbf{x}_t|\mu, \Sigma). \quad (4)$$

The maximum likelihood estimates are [4]

$$\hat{P}(\omega_j) = \frac{1}{T} \sum_{t=1}^T \hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma}) \quad (5a)$$

$$\hat{\mu}_j = \frac{\sum_{t=1}^T \hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma}) \mathbf{x}_t}{\sum_{t=1}^T \hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma})} \quad (5b)$$

$$\hat{\Sigma}_j = \frac{\sum_{t=1}^T \hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma}) (\mathbf{x}_t - \hat{\mu}_j)(\mathbf{x}_t - \hat{\mu}_j)^T}{\sum_{t=1}^T \hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma})}, \quad (5c)$$

where the conditional probability $\hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma})$ is

$$\hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma}) = \frac{N(\mathbf{x}_t|\hat{\mu}_j, \hat{\Sigma}_j) \hat{P}(\omega_j)}{\sum_{k=1}^M N(\mathbf{x}_t|\hat{\mu}_k, \hat{\Sigma}_k) \hat{P}(\omega_k)}. \quad (6)$$

The estimation algorithm is iterative. First, initial values for parameters $\mu_j, \Sigma_j, P(\omega_j)$ are needed in order to compute the conditional probabilities $\hat{P}(\omega_j|\mathbf{x}_t, \hat{\mu}, \hat{\Sigma})$. Knowing these conditional probabilities, the parameter estimates $\hat{\mu}_j, \hat{\Sigma}_j, \hat{P}(\omega_j)$ can be updated according to equations (5). These two steps, the computation of the conditional probabilities and the updating of the parameter estimates, are repeated until convergence is reached.

The maximum likelihood clustering yields the density codebook that is shared by the word models. In addition to the density codebook, we need to estimate the model state dependent weights c_{jk} for every Gaussian density. The weights are found by dividing all the training utterances for a model into a given number of segments. The segments correspond to the states in the model. The probabilities of the Gaussian densities in the model states are estimated based on the state segmentations.

The estimation of these probabilities is similar to the estimation of the probabilities of the Gaussian density functions of the density codebook. Based on the initial estimates, the conditional probabilities in each state of the word model are first computed according to equation (6). In the second step, the state probabilities for the Gaussians can be found from equation (5a). The iteration, the computation of the conditional probabilities in each state of the word model and the updating of the state probabilities, is repeated until the probabilities have converged.

3.2. Discriminative training

After the initial training, discriminative training is applied. The training scheme that was used is based on a

combination of corrective training and Generalized Probabilistic Descent (GPD) [5,6]. We use a similar misclassification measure as in GPD and a frame level weighting. The general idea is that the utterances and frames that resulted in a lower recognition confidence are used with an increased weight in training. Contrary to GPD the training is applied to the correct models only. By this restriction we intend to address the problem of a too rapid convergence in the training set at the expense of generalization power.

At word level the misclassification measure and loss functions are given by

$$d(X) = -P_{\log}(X|\lambda_c) + \log \left[\frac{1}{N-1} \sum_{\lambda, \lambda \neq \lambda_c} \exp(\delta P_{\log}(X|\lambda)) \right]^{\frac{1}{\delta}}$$

and

$$l(d(X)) = \frac{1}{1 + \exp[-\alpha(d(X) + \beta)]}. \quad (7)$$

In the previous formulas $P_{\log}(X|\lambda)$ represents the log-likelihood in the Viterbi sense for the model λ on utterance X , N is the total number of models, $\alpha > 0, \beta, \delta > 0$ are optimization control parameters and λ_c represents the correct model for X .

The objective function is defined as

$$Obj = E[l(X)], \quad (8)$$

and the target is to minimize the expected loss over all the utterances from the adaptation data.

Gradients are taken according to GPD and a gradient descent like algorithm is used in the optimization. The gradients are transformed into weights at utterance and frame level which are then used to update the models parameters. With the frame level weights we aim at an enhanced control over the training region within a word model. The weight is given by

$$w(x) = l_w \left(-P_{\log}(x|\lambda_c, s_x) + \max_{\lambda, \lambda \neq \lambda_c} P_{\log}(x|\lambda, s_x) \right) \quad (9)$$

where l_w is a frame level loss function as in (7) but with different values for α, β . $P_{\log}(x|\lambda, s)$ is the log likelihood for generating the frame x by model λ in state s . The indexing of the states denote that they are assigned by Viterbi alignment to frame x when recognizing the model λ .

4. EXPERIMENTAL RESULTS

Tests were done using the male section of the TIDIGITS database. In addition to the clean utterances, the training set contained also noisy utterances. The noisy utterances were obtained by mixing car noise with the SNRs of 0 and -10 dB. In training the number of the noisy utterances for both SNRs was half the number of utter-

ances from the clean environment. The test set contained all the clean five digit strings and their noise corrupted versions with the SNRs of 0 and -10 dB.

We used feature vectors consisting of 12 FFT-based MFCCs, log-energy and their first and second order time derivatives which were then subjected to a normalization scheme as in [7].

The vocabulary was composed of digits '1' to '9' plus 'oh' and 'zero'.

SCHMMs were tested against CDHMMs with similar model structures. The selected model structure was the traditional left-to-right state sequence with self loops and no skips.

The number of states in the word models were fixed based on the simulations in the clean environment only. In these initial simulations models with increasing number of states were tested and it was found that the optimum state counts for each digit model were close to the corresponding minimum word durations.

We aimed at obtaining a good performance with a number of parameters feasible for low complexity real-time recognition. From this point of view SCHMM with different codebooks having 200, 250, 300 and 350 densities were tried. On the CDHMMs part three sets of models having one, two and three mixtures per state were trained. In terms of total number of densities they correspond to 187, 374 and 561 respectively.

In order to speed up the recognitions only the scores along the best path were used in (1) and the sum from (2) was limited to the best term only.

4.1. Initial training

Initial SCHMM models were constructed with the training algorithm presented in Section 3.1. Table 1 presents the test set string error rates for the different SCHMM codebooks. In Table 2, the similar results for the CDHMMs can be found.

Table 1 String error rates of the SCHMMs on the test set for different codebook sizes.

CB size	Test set error rates			
	Clean	0 dB	-10 dB	Average
200	9.11	13.50	27.80	16.80
250	9.27	11.22	24.39	14.96
300	8.94	12.36	22.60	14.63
350	8.78	10.24	20.98	13.33

Table 2 String error rates of the CDHMMs on the test set for different mixture counts.

Mitures	Test set error rates
---------	----------------------

(at state)	Clean	0 dB	-10 dB	Average
187 (1)	8.29	8.29	20.98	12.52
374 (2)	7.32	6.18	14.31	9.27
561 (3)	5.37	5.04	15.45	8.62

By comparing the results from Table 1 with Table 2 we notice that after the initialization SCHMMs are in a clear disadvantage compared with CDHMMs. This comes to no surprise as the model dependent parameters and the codebook densities were separately trained.

4.2. Discriminative training

With the 2nd level of training we performed the joint optimization of model and codebook parameters. In this phase, the discriminative training methods presented in Section 3.2 were applied to SCHMMs. The training data was the same as in the initial training.

The joint parameter training improved the results significantly. DT resulted in the reduction of average error rates by 42.9%, 42.0%, 47.8% and 43.9% for the codebooks of 200, 250, 300, and 350 densities. The string error rates on the test set for the discriminatively trained SCHMMs are given in Table 3. The SCHMMs with codebook size 250 perform similarly with the most complex CDHMMs. The 300 and 350 SCHMMs are better than any CDHMM set after ML training.

Table 3 String error rates of the discriminatively trained SCHMMs on the test set.

CB size	Test set error rates			
	Clean	0 dB	-10 dB	Average
200	5.53	7.32	15.93	9.59
250	5.37	5.69	14.96	8.67
300	4.88	4.39	13.66	7.64
350	5.04	3.90	13.50	7.48

It is well known that also for CDHMMs ML training results can be improved with DT. The results for the discriminatively trained CDHMMs are given in Table 4. At the string level DT reduces the average error rates by 16.9%, 19.3% and 20.2% for 1, 2, and 3 mixture CDHMMs. We have less relative improvement in this case but the absolute results turn the balance in the favor of CDHMMs. By comparing the tables 3 and 4 we can notice that now the 350 SCHMMs are at the level of 2 mixtures CDHMMs (374 densities). The 3 mixture HMMs provide clearly the best results.

Table 4 String error rates of the discriminatively trained CDHMMs on the test set.

(at state)	Test set error rates			
	Clean	0 dB	-10 dB	Average
187 (1)	7.15	7.48	16.59	10.41
374 (2)	6.83	4.23	11.38	7.48
561 (3)	5.69	4.55	10.41	6.88

5. CONCLUSIONS

In this paper the training of SCHMMs was discussed and the performance of SCHMMs was compared to the performance of the CDHMMs.

The training of SCHMMs was composed of initialization followed by DT. The initialization of SCHMMs had two parts. First, the codebook of densities was estimated, and finally only the model dependent parameters were initialized. During DT, the codebook and the model dependent parameters were mutually estimated in order to obtain optimum recognition performance. The performance of SCHMMs was compared against the performance of CDHMMs that were trained with ML training followed by DT.

The performance comparison was done in a noise robust connected digit recognition task on the TIDIGITS database. Tests were done in clean and in noisy conditions with two SNR values (0 dB and -10 dB). For low complexity real-time recognitions SCHMMs and CDHMMs produced similar results in noise-robust digit dialing with the same amount of parameters. Since with the same density count SCHMMs are computationally more expensive, our conclusion at this point is that CDHMMs are more feasible for implementation.

6. REFERENCES

- [1] X.D. Huang, and M.A. Jack, "Semi-continuous hidden Markov models for speech signals," *Computers Speech and Language*, vol. 3. pp. 239–251, 1989.
- [2] X.D. Huang, K.-F. Lee, and H.-W. Hon, "On semi-continuous hidden Markov modeling," in *Proc. ICASSP90*, Albuquerque, New Mexico, USA, April, 1990, vol. 2, pp. 689–692.
- [3] R. Cardin, Y. Normandin, and R. De Mori, "High performance connected digit recognition using maximum mutual information estimation," in *Proc. ICASSP91*, Toronto, Ontario, Canada, 1991, vol. 1 pp. 533-536.
- [4] R.O. Duda, and P.E. Hart, *Pattern Classification and Scene Analysis*, New York, John Wiley & Sons, 1973.

- [5] B.-H. Juang, S. Katagiri, "Discriminative Learning for Minimum Error Classification", *IEEE Trans. on Signal Processing*, Vol. 40, No. 12, pp.3043-3054, 1992
- [6] W. Reichl, G. Ruske, "Discriminative Training for Continuous Speech Recognition", in *Proc. EUROSPEECH*, Madrid, Spain, 1995, pp. 537-540.
- [7] O.Viikki, K. Laurila, "Noise Robust HMM-Based Speech Recognition Using Segmental Feature Vector Normalization", in *Proc. of ESCA-NATO Workshop on Robust Speech Recognition for Unknown Communication Channels*, Pont-a-Mousson, France, April, 1997, pp. 107-110.