vergence to low error rates occurs faster and yields (in the average) better models. If the mean vectors of the multivariate Gaussian density functions are placed according to the clusters organized by SOMs, only a couple of iterations of maximum likelihood estimation is required to set suitable values to the other CDHMM parameters. The LVQ was used to get more discriminative clustering but it seems that the Baum-Welch algorithm cannot preserve this discriminativity very well. However, in the segmental K-means algorithm the lowest average speech recognition error rate was obtained when the mean vectors of the mixed Gaussians were initially created from the reference vectors by using LVQ.

References

[1] G. D. Forney. The Viterbi algorithm. In *Proceedings of the IEEE*, volume 6, pages 268–278, 1973.

[2] Biing-Hwang Juang. Maximum likelihood estimation for mixture multivariate stochastic observation of Markov chains. *AT&T Technical Journal*, 64:1235–1249, 1985.

[3] Biing-Hwang Juang and Lawrence R. Rabiner. The segmental K-means algorithm for estimating parameters of hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38:1639–1641, 1990.

[4] Teuvo Kohonen. Clustering, taxonomy, and topological maps of patterns. In *Proceedings of the 6th International Conference on Pattern Recognition*, volume 1, pages 114–128, Munich, Germany, 1982.

[5] Teuvo Kohonen. The Self-Organizing Map. In *Proceedings of the IEEE*, pages 1464–1480, 1990.

[6] Teuvo Kohonen, Jari Kangas, Jorma Laaksonen, and Kari Torkkola. LVQPAK: A program package for the correct application of learning vector quantization algorithms. In *Proceedings of the International Joint Conference on on Artificial Neural Networks*, Baltimore, June 1992.

[7] Mikko Kurimo and Kari Torkkola. Training continuous density hidden Markov models in association with self-organizing maps and LVQ. In *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, Copenhagen, Denmark, August 1992.

[8] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of Fifth Berkeley Symposium on Math. Statist. and Prob.*, pages 281–297, 1967.

[9] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. 77(2):257–286, 1989.

[10] Kari Torkkola, Jari Kangas, Pekka Utela, Sami Kaski, Mikko Kokkonen, Mikko Kurimo, and Teuvo Kohonen. Status report of the finnish phonetic typewriter project. In *Proceedings of the International Conference on Artificial Neural Networks*, volume 1, pages 771–776, Espoo,Finland, June 1991.

data. The weighting coefficient is then the probability of being in the state for which the parameters are estimated computed by the old parameter values. The difference in the segmental K-means compared to the Baum-Welch is that only data points assigned to the current state in the most probable state sequences are used in the estimation. The most probable state sequence for each training word is efficiently computed using the Viterbi algorithm based on dynamic programming [1].

In both of the training algorithms the amount of training data and the number of mixture components affect heavily the amount of required memory and calculations. In preliminary experiments we found that it is profitable to simplify the form of mixture densities by using global and diagonal covariance matrices for all mixture components. This also allows to use considerably more mixture components without increasing the computational load excessively. In [7] we determined the suitable number of mixture components for our system to be about 25. The problem with covariance terms is that their proper estimation requires vast amount of well representative training data [9] and poor estimates seems to be fatal for the recognition ability.

5 EXPERIMENTS

Experiments were carried out using the speech recognition system in the Laboratory of Information and Computer Science at Helsinki University of Technology [10]. The system works like a phonetic typewriter transcribing spoken Finnish utterances into phoneme sequences.

The tests were performed for three male Finnish speakers. For each speaker, four repetitions of a set of 311 Finnish words were available. The acoustical features used as observations of speech signal are 20-component cepstral vectors computed every 10 ms from a 20 ms window. The cepstral coefficients are weighted (liftered) with raised sine. The energy of the signal is concatenated to the vector.

The HMMs of 20 Finnish phonemes were trained by extracting the phoneme samples from three word sets spoken by the same speaker. The fourth set was then used for testing. The recognition performance was determined by calculating the average of 12 recognition tests containing all 4 combinations of training and testing data for each of the 3 speaker. The resulting error percentage plotted in figures 4 and 5 is the sum of changed, missing and extra phonemes in decoded phonetic transcriptions divided by the correct number of phonemes.

6 RESULTS

From figures 4 and 5 it can be observed that SOMs offer good initial values for the mixture density functions of the states in CDHMMs. This leads to a low recognition error rate using only a couple of iterations by the Baum-Welch or the segmental K-means estimation. When random initial values were used, significantly more iterations were required to achieve similar recognition rates.

The clustering of the training data can also be performed by, for example, the K-means algorithm [8], which resembles SOMs, except that it does not conserve the topology because the neighborhood includes only one vector in all cases. Comparing with the SOMs, it, however, appears to be inferior having about 5% higher average error rate after every iteration by the both maximum likelihood algorithms (for example, 8.2 against 7.8 after seven iterations of the segmental K-means).

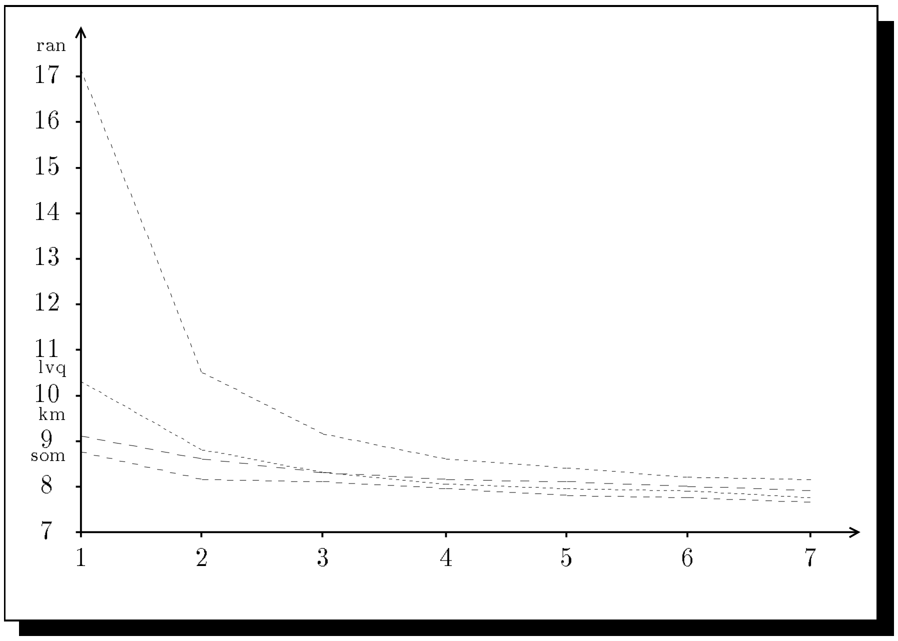

Figure 4: *Development of error rate during Baum-Welch iterations with random initial values "ran" compared to initialization done by SOM, K-means and LVQ. Number of iterations is shown in horizontal axis and percentage of recognition errors for independent test data in vertical axis.*

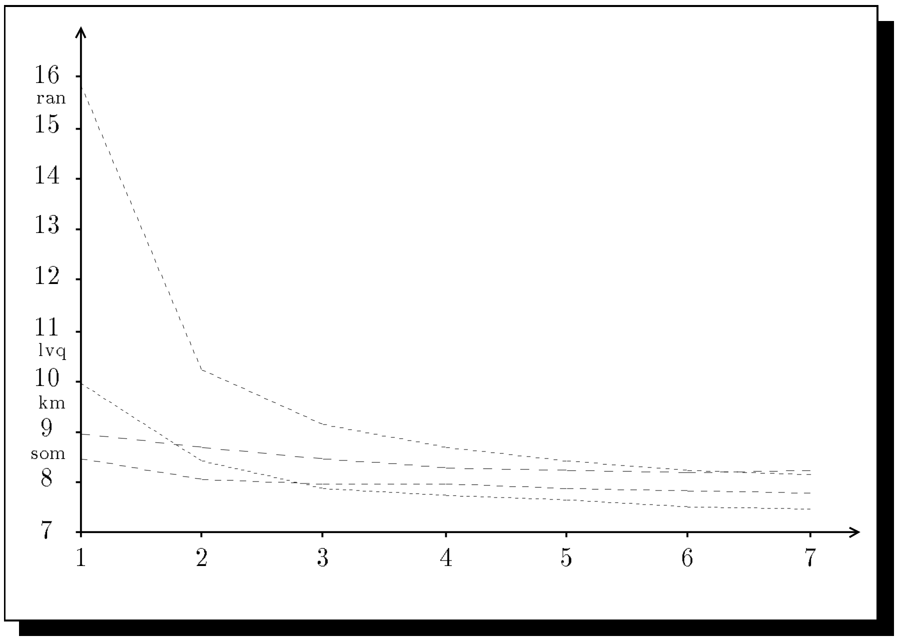

Figure 5: *Development of error rate during segmental K-means iterations with random initial values compared to initialization done by SOM, K-means and LVQ.*

Using LVQ algorithms to get more discriminative initialization produced low recognition error rates especially in the segmental K-means training (figure 5).

The best average recognition rates (correct phonemes about 94.3% and error rate of 7.4% including changed, missing and extra phonemes) in the iterative Baum-Welch were obtained by a combination with SOMs and in the segmental K-means with LVQ.

7 CONCLUSIONS

The combinations of maximum likelihood training and neural network based vector quantization by SOMs and LVQ offer attractive two phased training methods for CDHMMs. These combinations reduce the amount of computationally heavy operations in the parameter estimation, which enables the use of more training data and more components in mixture densities.

The speech recognition experiments show that through good initial values obtained by vector quantization methods the con-

In this paper we suggest to enhance the maximum likelihood training methods for CDHMMs by splitting the training into two phases (fig. 1). First, we determine the placement of the mean vectors of the Gaussian densities used in the mixtures by clustering the observations of each state. The second phase is to estimate mixture weights, covariance matrices and transition probabilities. To obtain the best possible parameter combinations also the mean vectors are allowed to be adjusted in the second phase. The initialization already gives rather good Gaussian densities so that most iterations of the maximum likelihood estimation can be neglected. The clustering can be done adaptively by LVQ and SOMs and is thus quite a fast procedure compared to Baum-Welch maximum likelihood estimation, when there is a large amount of training data.

In addition to speeding up the training, carefully chosen initial parameter values seem to lead, in average, towards better models as indicated by our experiments (see Sec. 5).

2 INITIALIZATION BY SOMS

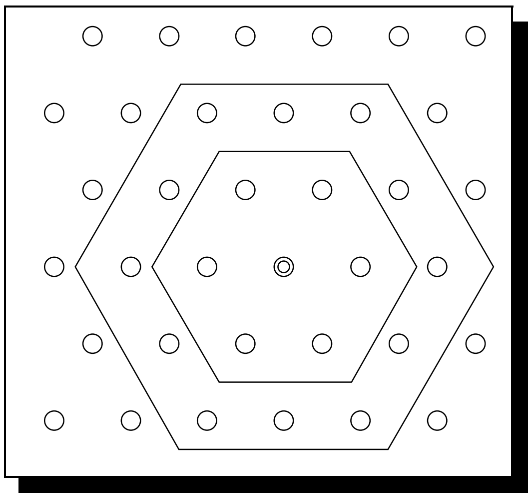

Figure 3: *In SOMs the reference vectors are organized to a two-dimensional array. The neighborhood can be defined using, for example, a hexagonal grid. Two neighborhoods which have a different radius are illustrated for the unit in the middle.*

When the mixture density function of a state has many components the initialization becomes especially important. Estimation of the mixture weights and the covariance matrices is based on the initial placement of each component's mean vector. The initialization of the mixture weights and the covariance matrices is more straightforward because adequate estimates can be obtained by analyzing separately the clustered observations of each state when the mean vectors of mixture components are first determined. The quality of the initial estimates depends naturally on the quality of the achieved clustering.

Self-Organizing Maps (SOMs) [4] are applied to produce adaptively a representative clustering of the observed feature vectors of in a state. In SOMs the reference vectors are organized to a two-dimensional array (fig. 3). For each training vector x the best-matching reference vector and its topological neighbors are adjusted closer to x. The size of the neighborhood and the relative adjustments are monotonically decreased during teaching. The best-matching unit m_c is determined by

$$c = \arg \min_i \{||x - m_i||\}. \qquad (2)$$

The SOMs for the experiments in figures 4 and 5 were trained as follows: The size of the SOM for each state was 5 times 5. Training data contains 5211 phonemes. Each phoneme sample was divided into 4 groups of feature vectors; one for each state in the HMM (see fig. 2). The feature vectors were then used in random order to update the corresponding originally random valued SOMs. The training data was used 5 times during which the teaching gain was decreased monotonically from 0.2 to 0 and neighborhood radius from 3 to 0.

3 DISCRIMINATING DIFFERENT STATES BY LVQ

We used also *Learning Vector Quantization* (LVQ) to place the mixture means in hope to enhance the discriminatory power of phoneme models. Training by any of the LVQ-variations [5] is adaptive and contains only simple operations. The objective is to approximate the decision boundaries in Nearest Neighbor classification by moving the nearest reference vectors to better positions according to the training vectors. In HMMs the classes correspond to the states. After the vector quantization the reference vectors are used as the mean vectors of the Gaussian density functions.

For example, the adaptation by LVQ1 occurs as follows:

- If the training vector x and its nearest reference vector m_c belong to the same class

$$m_c(t+1) = m_c(t) + \alpha(t)[x(t) - m_c(t)]. \qquad (3)$$

- If they belong to different classes, the adaptation occurs to the opposite direction

$$m_c(t+1) = m_c(t) - \alpha(t)[x(t) - m_c(t)]. \qquad (4)$$

The teaching gain $\alpha(t) \in (0,1)$ is decreased monotonically.

In LVQ2 the two best-matching reference vectors representing different classes are adjusted, if the sample vector appears at the border between two classes. LVQ3 is a combination of LVQ1 and LVQ2 having good long run behavior. OLVQ1 [6] is LVQ1 with optimized learning rate $\alpha_i(t)$ defined individually for each m_i.

The reference vectors of LVQ can be initialized by finding a group of vectors which satisfy the K-nearest neighbor (KNN) criterion as suggested in [6]. The KNN criterion states that from the K-nearest neighbors in the training data the majority must belong to the same class as the tested vector.

In this application of LVQ the vector to be updated corresponding to a training feature vector is selected by finding the closest Gaussian mean vector in the group of all mixture components in all state output density functions. In figures 4 and 5 the LVQ1 learning law was used with teaching gain decreasing monotonically from 0.05 to 0. The whole training data set was used 2 times. In this application all the learning laws LVQ1, LVQ2, LVQ3 and OLVQ1 gave almost equal recognition rates.

4 CDHMM TRAINING

When the initial parameter estimates are obtained using the predescribed methods, the final values are computed using only a couple of iterations of the two alternative maximum likelihood methods the *Baum-Welch re-estimation* [2] and the *segmental K-means* algorithm [3]. Objective in these algorithms is to find a model that would have produced the training state sequences with the highest probability.

In the Baum-Welch re-estimation the new parameter values can be interpreted as the weighted averages from all training

Application of Self-Organizing Maps and LVQ in training continuous density hidden Markov models for phonemes

Mikko Kurimo†and Kari Torkkola‡

†Helsinki University of Technology
Laboratory of Information and Computer Science
Rakentajanaukio 2 C, SF-02150, Espoo, FINLAND
tel: +358 0 451 3266, fax: +358 0 451 3277
email: mikko.kurimo@hut.fi
‡IDIAP, CP 609, CH-1920 Martigny, SWITZERLAND

Abstract

We present experiments in using neural network based methods to initialize continuous observation density hidden Markov models (CDHMMs). Proper initialization provides an easy way to avoid excessive amount of iterations, when maximum likelihood algorithms are used to estimate the parameters of CDHMMs. This is important in, for example, phoneme based automatic speech recognition, where the output density functions of the states of HMMs are complex and a lot of training data must be used. In our work CDHMMs are used as phoneme models in the task of transcribing speech into phoneme sequences. The probability density function of the output distribution for a state is approximated by mixture of a large number of multivariate Gaussian density functions (typically 25). We present experiments of initializing the means of mixture Gaussians by Self-Organizing Maps (SOMs) and Learning Vector Quantization (LVQ). The results of the experiments indicate that initialization by SOMs speeds up the convergence in ML-parameter estimation, when error rate is used as a measure. The same applies to LVQ especially combined with segmental K-means algorithm.

1 INTRODUCTION

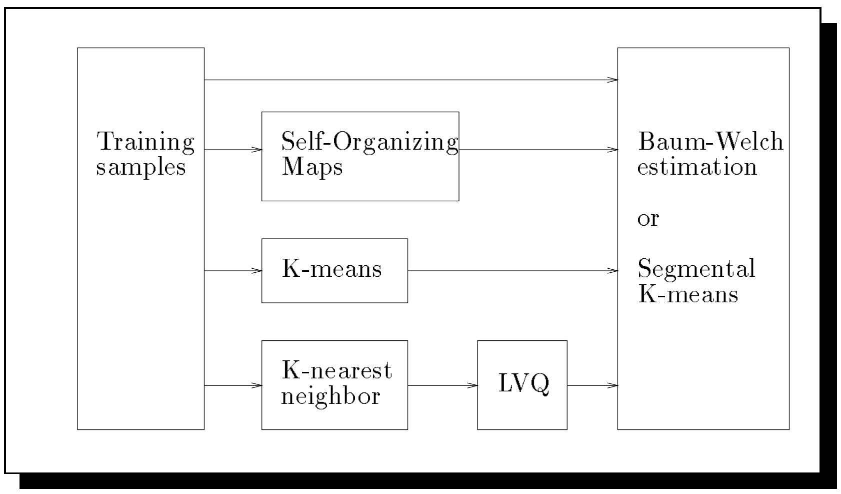

Figure 1: *The training combinations experimented with CDHMMs.*

Accurate modeling of the feature vectors produced by a phoneme is difficult but vital for the successful operation of a speech recognition system based on phonemes. In CDHMMs the phonemes are modeled as a sequence of states. A state of a phoneme is assumed to produce observations which are represented by a weighted mixture of parametric probability density functions (see fig. 2). The probability that state i can produce an observation O_t is then calculated by

$$b_i(O_t) = \sum_{m=1}^{M} c_{im} b_{im}(O_t) \quad , \tag{1}$$

where the components $b_{im}(O_t)$ are e.g. multivariate Gaussian densities and the weight factors $c_{im} < 1$ are positive real numbers which sum to 1 for each state i.

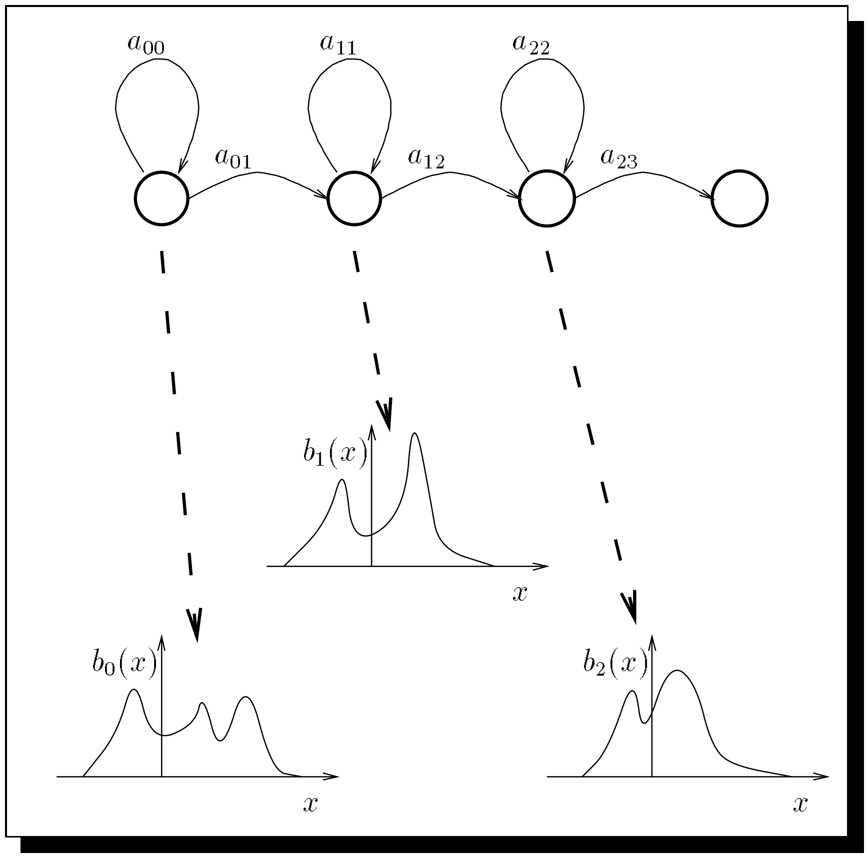

Figure 2: *The Markov model of a phoneme is 4-state unidirectional chain. Each state has its own continuous observation distribution $b_i(x)$ and discrete transition distribution a_{ij}. The observations x are projected to scalars here only for illustrative purposes. The true dimension of x used in experiments is 21.*

A big problem in using CDHMMs is the estimation of the model parameters effectively in order to obtain the best possible speech recognition performance. The acoustic features of a phoneme vary considerably depending its context and the manner it is pronounced. Due to this variability it is important to use a substantial amount of well representative training data. There must also be quite many components in the mixture density functions (1) to cover the necessary output variations in a state. Another solution would be the use of context dependent models. The estimation of the continuous observation density models by the maximum likelihood principle involves considerable computational complexity and several successive iterations. The training is also quite sensitive to the initialization of the model parameters [9].