[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*, August 1992. To be published.

[8] Louis A. Liporace. Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Transactions on Information Theory*, 5:729–734, 1982.

[9] 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.

[10] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of ICASSP*, volume 1, pages 267–295, 1989.

[11] 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.

comparison between different initialization methods produced quite similar results (fig. 4). When random initial values were used, significantly more iterations were required both in the case of the Baum-Welch and of the segmental K-means algorithms to achieve good recognition rates.

The SOMs were trained in this experiment 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. 1). 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.

Using LVQ algorithms to get more discriminative initialization produced low recognition error rates for the Baum-Welch training (figure 3) and especially for the segmental K-means training (figure 4). The codebook vectors of LVQ were 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 was selected by finding the closest Gaussian mean vector in the group of all mixture components in all state output density functions. For adjustments the learning laws LVQ1, LVQ2, LVQ3 [5] and the optimized learning rate OLVQ1 [6] were tried with almost equal recognition rates. In figures 3 and 4 the recognition error rates are illustrated for LVQ1 where the teaching gain was decreased monotonically from 0.05 to 0 and the whole training data set was used 2 times.

# 6 CONCLUSIONS

It is shown by experiments that a careful initialization of the parameters determining the observation density functions of the states in CDHMMs speeds up the convergence and leads to better models (in average). The criterion by which the models are compared is the performance in speech recognition. The improvements due to better initialization occur both in the iterative Baum-Welch and in the segmental K-means algorithms. The increased speed of convergence allows the use of more accurate and complex models which require more training data and iterations in estimation.

The new methods introduced in this paper for training the CDHMMs are combinations of iterative maximum likelihood estimation algorithms and different vector quantization methods. The quantization methods were used to select suitable initial parameter values in order to reduce the number of iterations of the numerically more complicated maximum likelihood methods. The clustering of the training data determines initial placements for the means of the multivariate Gaussian density functions which approximate the continuous observation density of each state in HMMs. The LVQ was used to get more discriminative clustering but it seems that the Baum-Welch algorithm cannot preserve this discriminativity very well. However, the segmental K-means algorithm converged to the best results when combined with the LVQ. The best results with the iterative Baum-Welch were obtained by a combination with the Self-Organized Maps.

# References


[1] X.D. Huang and M.A. Jack. Unified techniques for vector quantization and hidden Markov modelling using semi-continuous models. In *Proceedings of ICASSP*, volume 1, pages 639–642, New York, 1989.

[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.

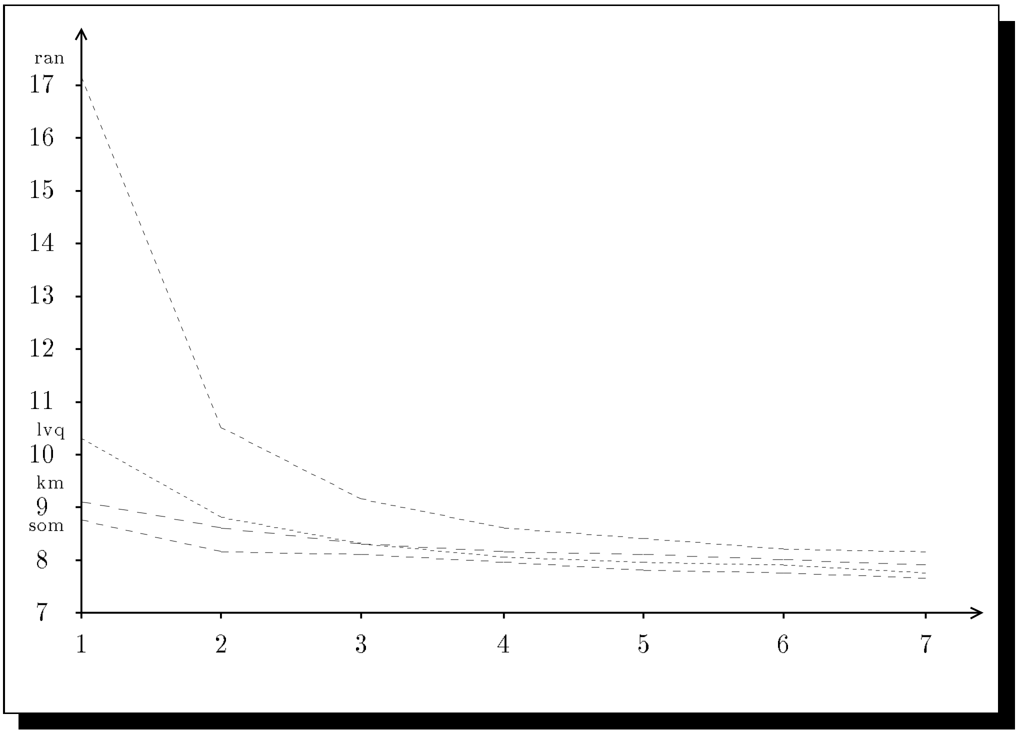


Figure 3: *Development of speech recognition error rate during Baum-Welch iteration with random initial values "ran" compared to initialization done by SOM, K-means and LVQ. Number of iterations is shown in horizontal axis and percentage of recognition errors for independent test data in vertical axis.*

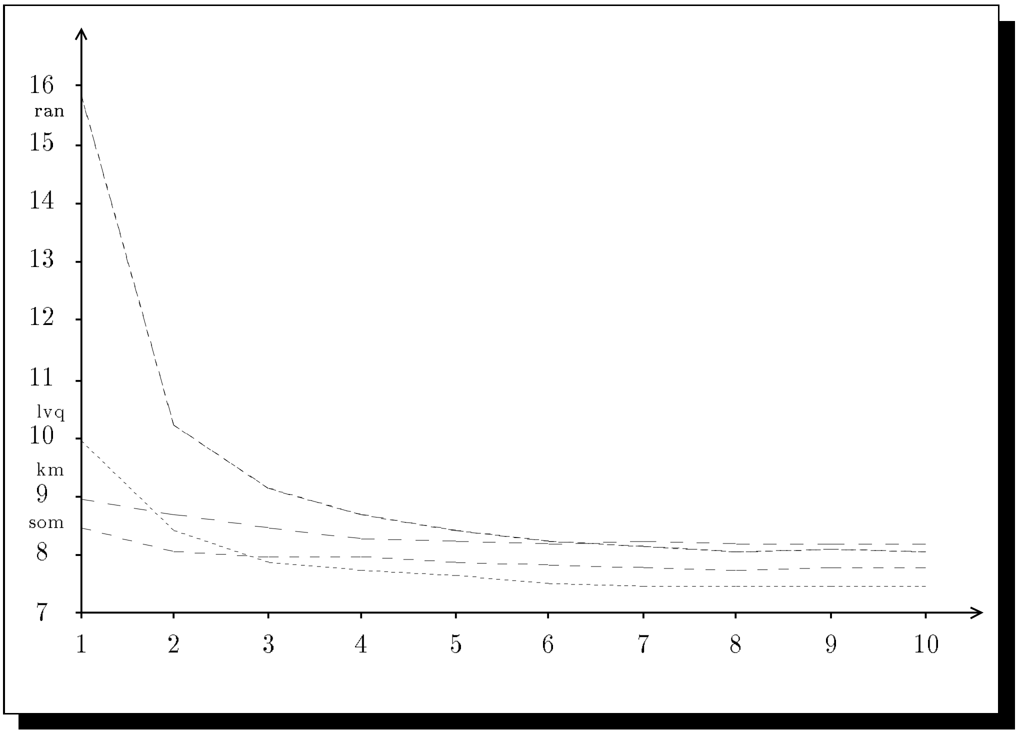


Figure 4: *Development of speech recognition error rate during segmental K-means iteration with random initial values compared to initialization done by SOM, K-means and LVQ.*

# 4 TRAINING

The final objective in training the HMMs is to estimate the parameters of the model $\lambda = (A, B, \pi)$ using the training data available so that the recognition accuracy of the test data is maximized. There has been many efforts to develop such a training method but unfortunately the generally optimal method is still (and probably also will be) unreached.

If the objective is formulated as maximization of the probability $P(O|\lambda)$ which means finding a model that would have produced the training state sequences with the greatest probability, the iterative Baum-Welch maximum likelihood re-estimation algorithm (in e.g.[10]) can be applied. In this method the new parameter estimates are found by computing the expectation values as weighted averages of the training data where the state probabilities calculated with the old parameters are the weighting factors.

As shown in [8] and [2] the iterative Baum-Welch method generates models that approach the maximum of $P(O|\lambda)$ and good results reported in the literature verify the power of this method in many speech recognition applications. The Baum-Welch re-estimation has, however, some practical drawbacks. In real applications the amount of training data required to obtain accurate models is relatively large and makes re-estimation cycles computationally heavy and memory consuming. When longer feature vectors and increased number of mixture components are used, the increased distribution complexity seems to require more iteration cycles before the models can be successfully applied for speech recognition purposes.

The computational difficulties can be reduced by using only the most probable state sequence for each training word instead of weighted averages of all possible sequences. In this method the new parameter estimates are expectation values computed from observations classified to the state. The optimization criterion is now the so called state-optimized likelihood of a state sequence $Q$ defined by

$$L_\lambda(Q^*) = \max_Q P(O, Q|\lambda). \tag{8}$$

This method is called the segmental K-means algorithm and its proof of convergence is given in [3]. The most probable state sequences are calculated using the Viterbi algorithm based on dynamical programming.

Another way to lighten the training is to reduce the number of iterations by speeding up the convergence. The convergence occurs faster and better models are achieved (in average), if the re-estimation is started from suitable initial values both in the Baum-Welch (fig. 3) and in the segmental K-means algorithm (fig. 4).

# 5 EXPERIMENTS

We have made experiments with the proposed modeling and training methods using the speech recognition system in the Laboratory of Information and Computer Science at Helsinki University of Technology. The tests were performed for three male Finnish speakers. For each speaker, four repetitions of a set of 311 words were available.

The hidden Markov models 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 system works like a phonetic typewriter writing the phonetic transcriptions of spoken Finnish words which are then compared with the correct transcriptions.

The recognition performance for each differently trained model 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 3 and 4 is the sum of changed, missing and extra phonemes in decoded phonetic transcriptions divided by the total number of phonemes.

Use of SOMs to give good initial values for the mixture distributions of the states in HMMs led to the fastest convergence in Baum-Welch re-estimation when comparing to the other clustering methods (fig. 3). Also the final error rate with SOMs was lower than the error rate with K-means. With the segmental K-means algorithm the

The initialization of mixture weights and 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. The neural network based methods, SOMs and LVQ, which are used in experiments to place the centers of mixture components, are described below.

## 3.1 SOMs

The SOM [4] for the feature vectors produced by one state in a HMM is trained by selecting the sample feature vectors $x$ one at a time in a random order and updating the SOM according to each vector. The update of the SOM is done by adjusting the best-matching unit and its neighbors closer to $x$. The best-matching unit $m_c$ is determined by

$$c = \arg\min_i \{||x - m_i||\}. \tag{5}$$

The adaptation occurs as follows

$$m_i(t+1) = \begin{cases} m_i(t) + \alpha(t)[x(t) - m_i(t)] & \text{for } i \in N_c(t), \\ m_i(t) & \text{for } i \notin N_c(t). \end{cases} \tag{6}$$

The neighborhood $N_c(t)$ around the best matching unit is wide in the beginning of the training and shrinks monotonically with time. The teaching gain $\alpha(t) \in (0,1)$ is also monotonically decreased during teaching.

## 3.2 LVQ

In the LVQ the codebook vectors are adaptively adjusted using sample vectors $x$ randomly chosen from training data. The adjustments are made according to the supervised learning laws LVQ1, LVQ2, LVQ3 [5] and OLVQ1 [6] which modify the nearest codebook vector $m_c$ determined by equation (5).

For example, in the LVQ1 learning law the direction of the adjustment depends on the class of the nearest codebook vector $m_c$.

$$\begin{aligned} m_c(t+1) &= m_c(t) + \alpha(t)[x(t) - m_c(t)] \quad \text{if } x \text{ and } m_c \text{ belong to the same class,} \\ m_c(t+1) &= m_c(t) - \alpha(t)[x(t) - m_c(t)] \quad \text{if } x \text{ and } m_c \text{ belong to different classes,} \\ m_i(t+1) &= m_i(t) \quad \text{for } i \neq c. \end{aligned} \tag{7}$$

The teaching gain $\alpha(t) \in (0,1)$ is monotonically decreased during teaching. The LVQ2 and LVQ3 differ from LVQ1 by adjusting the two best-matching codebook vectors representing different classes if the sample vector appears at the border between two classes. OLVQ1 is LVQ1 with optimized learning rate $\alpha_i(t)$ defined individually for each $m_i$. In HMMs the classes correspond to the states of the HMMs.

## 3.3 Other methods

Other algorithms can also be used for clustering, for example, the K-means algorithm [9], which is somehow similar to SOM, except that it doesn't conserve the topology because the neighborhood includes only one vector in all cases. It produces more recognition errors than SOM (see fig. 3) after the same Baum-Welch training, however. The situation was same when the segmental K-means method was used instead of Baum-Welch (fig. 4).

density function

$$b_i(O_t) = \sum_{m=1}^{M} c_{im} b_{im}(O_t) \ , \tag{4}$$

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 $i$. The suitable number of distribution components $M$ for our system was determined to be about 25 [7].

To be able to estimate and to use this kind of huge mixtures certain generalizations have to be made with the parametric distributions. Because poor estimation of covariance terms seems to be fatal for the recognition ability the increase of the number of mixture components requires vastly more well representative training data [10]. The generalization and diagonalization of the covariance matrices increase the accuracy of which they can be estimated when the amount of training data is limited as there remains more data for the estimation of different covariance parameters. This also allows to use considerably more mixture components without increasing the computational load excessively.

# 3 INITIALIZATION

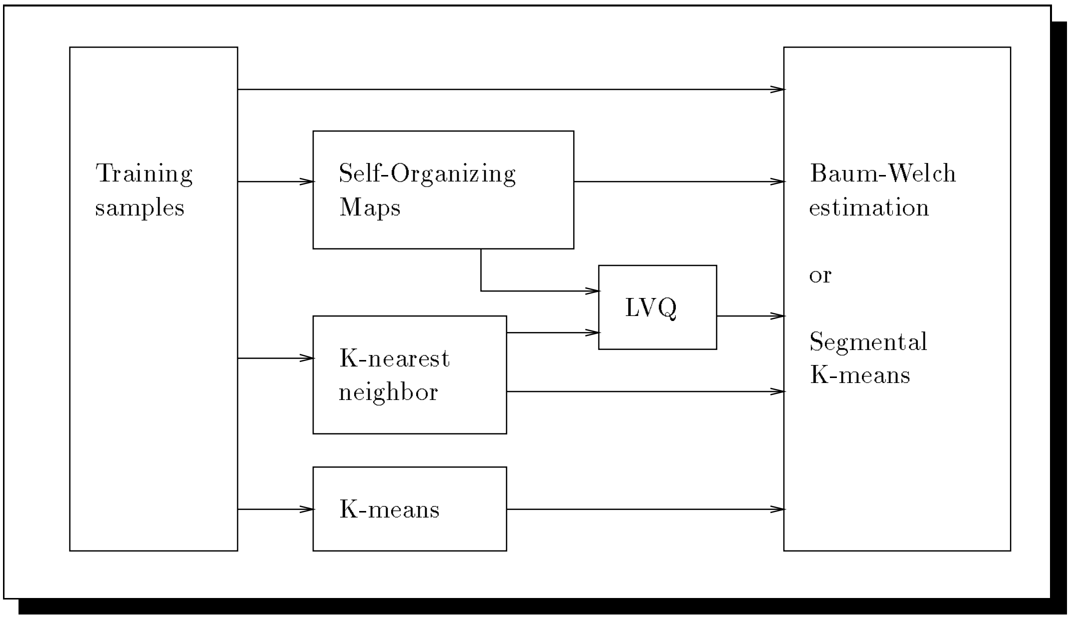


Figure 2: *Different training combinations for observation distributions.*

Choosing the initial values for the continuous observation distributions randomly is often adequate because the Baum-Welch iteration tends to converge relatively fast, at least in the case of simple mixture distributions. There is also no generally applicable optimal and well justified initialization method which would guarantee the best possible initial values for the Baum-Welch iteration. However, when large mixtures of high-dimensional feature vector distributions are required advanced initialization methods are profitable.

Increasing the number of mixture components makes the proper initial placement of the components nontrivial. Clustering of training samples would assure the best possible exploitation of all component distributions. The clusters are later replaced by multivariate Gaussian density functions having mean vectors identical to the centers of the corresponding clusters. If the vectors representing each cluster were chosen to maximize the discrimination between states, the resulting observation distributions might also be well discriminative. The different training combinations experimented with are presented in figure 2.

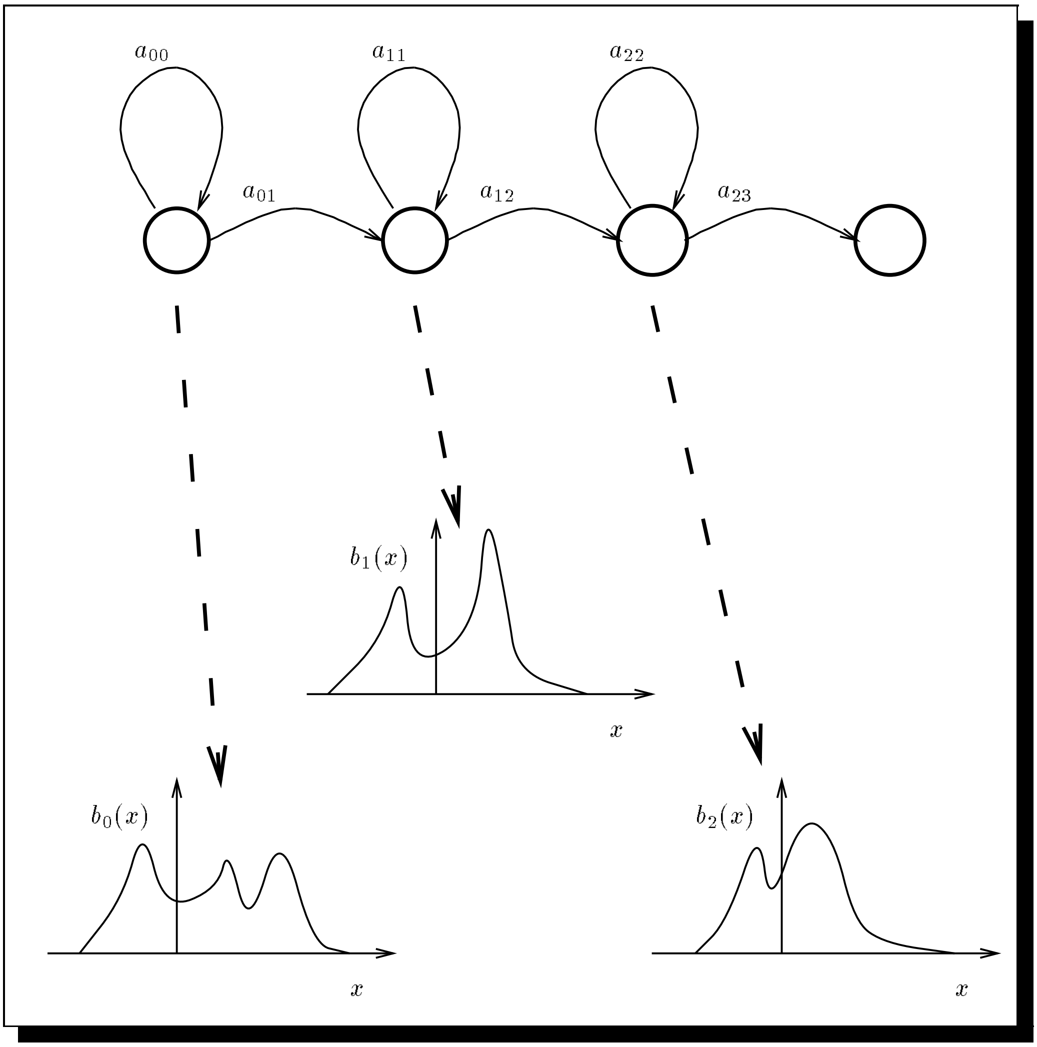


Figure 1: *The Markov model of a phoneme is 4-state uni-directional chain. Each state has its own continuous observation distribution $b_i(x)$ and discrete transition distribution $a_{ij}$. The observations $x$ are projected to scalars here only for illustrative purposes. The true dimension of $x$ used in experiments is 21.*

where

$$a_{ij} = P[q_{t+1} = j | q_t = i], \qquad 1 \leq i, j \leq N, \tag{1}$$

the observation probabilities $B = \{b_i(O)\}$ where

$$b_i(O_t) = P[O_t | q_t = i], \qquad 1 \leq i \leq N \tag{2}$$

and the initial state distribution $\pi = \{\pi_i\}$ where $\pi_i = P[q_0 = i]$. The state of the system at time $t$ is denoted by $q_t$ and the observation by $O_t$.

The stochastic process, represented by the observation sequence $O = O_1, O_2, \cdots, O_T$, is characterized by the probability of the observations having been generated by the model $\lambda$

$$P(O|\lambda) = \sum_q \pi_{q_0} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(O_t). \tag{3}$$

The probability density function of observations in one state is normally quite complex and is modeled by a mixture

of the speech signal and also some essential information for the classification task can be lost in the process. On the other hand, if the quantization codebook is trained using the LVQ algorithms [5] the reference vectors can be selected optimally in the sense of maximizing the phoneme discrimination ability. Increased differentiation properties of observation probabilities have resulted in excellent recognition accuracies, for example, in the Finnish phonetic typewriter project [11].

The estimation of the continuous observation density models involves considerable computational complexity, especially when the shape of the true observation density differs substantially from the mixture density function being adjusted. In this case the estimation procedure requires many iterations. The maximum likelihood training methods for the continuous observation density models are also quite sensitive to the initialization of the model parameters [10].

The semi-continuous HMMs have been suggested as a combination of the two most popular ways to model the distribution of observations [1]. In the SCHMMs the quantization vectors representing the discrete output symbols are replaced by Gaussian densities to avoid quantization errors. The number of free parameters is reduced by using the same Gaussian densities for all states.

In this paper we suggest to enhance the maximum likelihood training methods for CDHMMs by splitting the training in two phases, like in the case of discrete observation distributions. The first phase is to determine the placement of Gaussian densities by similar methods as in vector quantization. The second is to estimate the mixture weights which are the same as the conditional probabilities for each Gaussian of the state that they could have produced the observations (see eq. 4). During experiments it was noted, however, that also the placement of the Gaussians should be re-estimated by the maximum likelihood algorithms to obtain the best possible results. The first phase can then be considered as an initialization for the second phase. This initialization already gives quite suitable Gaussian densities so that most iterations of the maximum likelihood estimation can be neglected. This is advantageous because vector quantization using SOMs is quite a fast procedure compared to, for example, one iteration of Baum-Welch re-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 (see e.g. fig. 3).

# 2 PHONEME MODEL

The observations of speech signal used in our speech recognition system are short-time feature vectors computed every 10 ms from a 20 ms window placed over the sampled speech waveform. The feature vector contains 20 cepstral coefficients weighted (liftered) with raised sine [11]. The energy of the signal is concatenated to the vector.

The phoneme models are 4-state uni-directional Markov chains in which each state has its own continuous observation density function and a discrete state transition distribution (see figure 1 and table 1 for an example of decoding).

```
K>><><<>PP<<MAAAAAAAAAAAAAA[AAEEEIIJJJJJIJIIII><>>>>>>TKKKKOAAAAAAAAAH<<<<<<<<<<<>>K<<<<<K<<K
23323212100133222233132333001132111231313111010303033010333310001113120000101111133121002222

>>>>>>>>>AAAAAAAAAAAAAAAAAAAIIIIIIIIIIIIIIIIKKKKKKKKKKAAAAAAAAAAAAA<<<<<<<<<<<<<<<<<<<<<<<
0000122230000122222222222222300111111111222222301122222223000000011123000001111111111111222223
```

Table 1: *An example of the use of HMMs to decode observations of speech signal. In upper rows the short-time features of the speech signal from Finnish word "AIKA" are classified independently to the most probable states that could have generated the observations. Below is the same sequence of feature vectors, but the most probable path through states is found by Viterbi search which take also the state transition probabilities into account. The number (0,1,2 or 3) below each phoneme label is the number of the current state in Markov model of that phoneme. The silence preceding the word is indicated by label* $>$ *and the following by* $<$.

The Markov model can be expressed as $\lambda = (A, B, \pi)$ [10] consisting of the transition probability matrix $A = [a_{ij}]$

# Combining LVQ with continuous density hidden Markov models in speech recognition

Mikko Kurimo†and Kari Torkkola‡
†Helsinki University of Technology
Laboratory of Information and Computer Science
Rakentajanaukio 2 C, SF-02150, 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 propose the use of Self-Organizing Maps (SOMs) and Learning Vector Quantization (LVQ) [5] as an initialization method for the training of the continuous observation density hidden Markov models (CDHMMs). We apply CDHMMs to model phonemes in the transcription of speech into phoneme sequences. The Baum-Welch maximum likelihood estimation method is very sensitive to the initial parameter values if the observation densities are represented by mixtures of many Gaussian density functions. We suggest the training of CDHMMs to be done in two phases. First the vector quantization methods are applied to find suitable placements for the means of Gaussian density functions to represent the observed training data. The maximum likelihood estimation is then used to find the mixture weights and state transition probabilities and to re-estimate the Gaussians to get the best possible models. The result of initializing the means of distributions by SOMs or LVQ is that good recognition results can be achieved using essentially fewer Baum-Welch iterations than is needed with random initial values. Also in the segmental K-means algorithm the number of iterations can be remarkably reduced with a suitable initialization. We experiment furthermore to enhance the discriminatory power of the phoneme models by adaptively training the state output distributions using the LVQ-algorithm.

## 1 INTRODUCTION

The observations corresponding to a state of a phoneme-model HMM are not distributed according to any simple probability density function. In speech recognition these distributions are usually modeled by weighted mixtures of parametric probability density functions or by a set of symbols each having different discrete observation probabilities in different states. It is difficult to estimate accurately the probability density models because the acoustic features of the same phonemes vary considerably even for the same speakers. The articulation of phonemes depends on their context and speaking with different speeds and manners produces different acoustic features even in the same words. Due to this variability it is important to use flexible models and to have enough training data which covers the most frequent variations of phonemes.

Because the coarticulation and continuity, the segmentation of speech signal into phonemes is not a straightforward procedure. The stochastic methods, in which the segmentations can be ranked by their probabilities or likelihoods, perform quite well depending, of course, on how much these probabilities resemble the true situation. To compute these segmentation probabilities it is necessary to consider the observation probabilities for all phoneme states, not just to decide to which state each consequent observation most likely belongs (table 1). Hence the accurate modeling of the observation densities is very important for the success of the stochastic segmentation methods like, for example, the Viterbi algorithm.

The discrete observation distribution models are easier to estimate than the continuous density models because the selection of the vectors defining the output symbols by vector quantization can be separated from determining the observation probabilities of the symbols for each state. Vector quantization reduces vastly the information content