18 Speech Recognition

Mikko Kurimo and Panu Somervuo

18.1 The Recognition System

The recent projects in automatic speech recognition (ASR) are aimed both to use the recognition system as a test bench for the neural network algorithms developed in the laboratory and to develop the system itself as a pilot application of the neural networks. To produce respectable results, the best modeling and learning methods are applied with our own latest developments to fully exploit the available computer technology so that the recognition can still operate online in real time with high-dimensional input features. The results show that by the improved methodology and hardware, reductions in recognition error rate have been successful.
The speech recognition by the system developed in our laboratory occurs in five successive phases (see Figure 28). The most significant improvements have lately been introduced to the second and third phases. Some new inventions have also been tested for the spectrum analysis and for the phoneme string corrections.

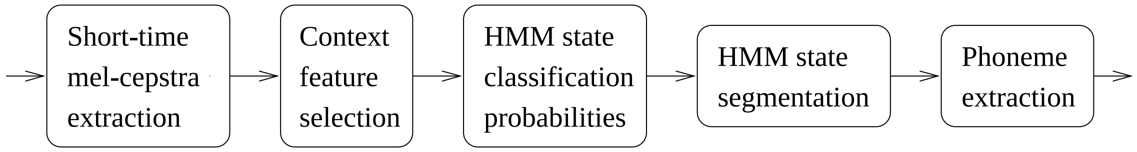

Figure 28: The main phases of the ASR by HMMs.

The recognition task used as a test bench for the new developments is the speaker dependent, but vocabulary independent ASR. The recognition is based on connecting the hidden Markov models (HMMs) of the phonemes to decode the phoneme sequences of the spoken utterances [5]. The HMM parameters can be automatically trained by neural network based methods using only a set of training words for each speaker. The output density function of each state in each model is a mixture of multivariate Gaussian densities.

18.2 Determination of the Error Rate

In the speech database collected here mostly in 1995, there are currently data of 20 speakers and at least four recording sessions of 350 Finnish words for each speaker. The speaker dependent recognition models are trained using three word sets and tested by the remaining one. The error rate given as the result is the number of all phoneme errors (inserted,deleted and changed phonemes) divided by the total number of phonemes. To gain statistical significance for the model comparisons, the tests are normally made for seven different speakers and the error rates are averaged. For verifying the robustness of the models for slightly different speech data also an older database (from 1990) is sometimes used. In general, the older database gives lower average error rates, probably because of the more experienced speakers.

For comparisons of the models the post-processing by the Dynamically Expanding Context (DEC) [1] is not applied in order to extract all the differences of the results. The long phonemes like /AA/ are separated from their short counterparts by using phoneme dependent duration limits learned iteratively during the model training. This simple separation do not take the word context into account and produces some errors which, in addition to some minor mismatches between the written and spoken format of the words, affect to the lowest obtainable value for the error rate. The acoustic features used throughout this work are the mel-cepstrum coefficients and RMS value of the signal. The basic feature vectors for the experiments are 20 component cepstra, but also extended feature vectors like averaged, concatenated and delta cepstra were tested and for those sometimes only 10–15 first coefficients were used.

18.3 Selection of Context Vectors and Multiple Feature Streams

The implementation of HMM is usually a first-order model and the context of short time feature vectors is thus not fully used. By using the context of the short-time feature vectors (see Figure 29), the coarticulation effect can be taken into account already in the feature extraction stage. The problem is how to define a suitable context. When the context is added to the recognition process there are two alternatives: whether to concatenate it to the short time feature vector or to use parallel feature streams in HMM so that the context is in its own feature stream.

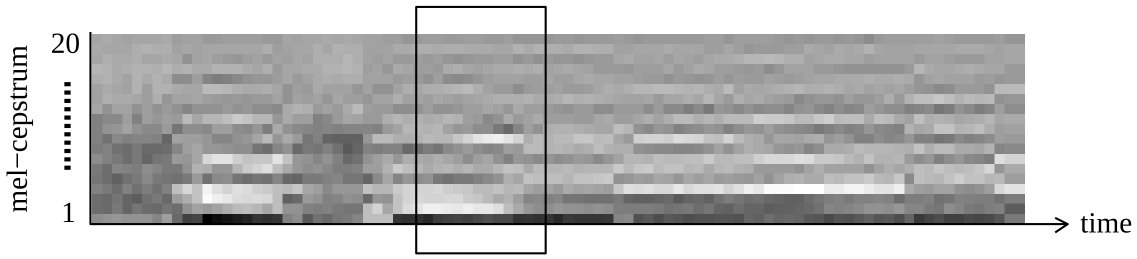

Figure 29: The features of the spoken word "OTANIEMI". The context vectors are combinations of several successive short-time features. The context window is 0.1s wide.

Several experiments were done in order to find discriminative features and a suitable context vector. Compared to the [4], new elements were delta features and the investigation of a proper time span for both static short-time feature concatenation and delta computations. One objective was to keep the dimension of the final feature vector suitably low. The effect of kernel width in SCHMM using Gaussian kernels was experimented with several feature vectors. All context vectors were found more tolerable to the change of kernel width than a single short-time feature. Compared to the case of using only one single mel-cepstrum, the use of three concatenated mel-cepstra dropped the phoneme recognition error from 6.7% to 3.5% and from 7.6% to 4.7% for two test speakers having different speaking rates. It was interesting that good recognition results were obtained even by using only delta features. Two concatenated differences of mel-cepstra as a feature vector gave similar or even better results than one static mel-cepstrum vector. This is remarkable because when only difference features are used, the long time average of static features (channel bias) is automatically removed from the feature vector. The phoneme recognition errors for two test speakers using two concatenated mel-cepstra differences as a feature were 4.9% and 7.6%. When a static mel-cepstrum was concatenated to this feature vector, the corresponding errors were 3.1% and 6.0%. In general, significant improvements in recognition results were achieved when one or more additional static or difference mel-cepstra was concatenated to the single static or single difference mel-cepstrum. This shows the importance of using context of a single short-time feature vector.

Two alternatives, whether to concatenate the context vector into one feature stream or use parallel feature streams in HMM were experimented. When one static mel-cepstrum was used in one HMM input stream and two concatenated mel-cepstrum differences were used in another input stream, the phoneme recognition errors were 3.0% and 5.4% for test speakers. Here equal stream weightings were used. The latter error dropped to 5.0% when more weight was given to the static mel-cepstrum stream. The average phoneme error rate for six independent speakers using equal stream weighting was 5.6% when the feature selection was done according to the results obtained for two test speakers with different speaking rates. Compared to the baseline system, which had used only one static mel-cepstrum as a feature the phoneme error rate being 8.5%, the additional context stream gave the error reduction of 35%.

As a conclusion, the context of a single short-time feature vector is important and this study has shown that significant improvements in the recognition rate can be obtained by forming the context using only a few short-time feature vectors.

18.4 Scaling the Recognition System Up by using Extended Feature Vectors

Despite that the ASR systems should be able to operate online, it is also vital to study what will happen to the developed modeling and training method, when the dimensions are doubled or trebled. Actually, the computational capacity of the workstation used for the ASR demonstrations of the laboratory is now about five times than three years ago.

In the tests reported in Table 7 the dimensions are increased by adding delta features and concatenating averaged successive feature vectors into a high-dimensional context vector. The objective of these extended feature vectors is to provide the HMMs more freedom to create component-wise sequential dependencies by giving the observation densities of the states information on variable length features.

The recognition times in the Table 7 includes simple optimizations such as the partial distance computation and the ordered search mentioned in [3]for efficient computation with high-dimensional vectors. The HMMs in test were mixture density HMMs (MDHMMs) with 70 Gaussians per phoneme trained by SOMs and the segmental LVQ3. The demonstration system developed in 1994 with 24 Gaussians per phoneme gave the performance values 6.9 %, 9.8 % and 0.5 for the last three columns of the Table 7, respectively.

A completely different preprocessing approach has also been studied to develop

Feature vector	Cepstra		RMS		Context	Total dim.	Error rate%		Recognition time factor
		Δ		Δ			Data 90	Data 95	
basic	20		1		no	21	5.5	7.7	1.0
delta21	10	10	1		no	21	4.7	6.8	1.3
delta42	20	20	1	1	no	42	4.0	6.4	1.8
context80	15		1		5	80	3.6	5.3	1.7
context105	20		1		5	105	3.4	5.3	1.8

Table 7: The contents of the alternative feature vectors in the tests and the average test set error rates. The MDHMMs are trained by SOMs and the segmental LVQ3. The recognition time factor is the average recognition time per word divided by that of the baseline system ("basic" features).

feature extraction methods that correspond better the subjective voice observation of a human. By visual inspection the phonemes seem to be more distinct by the obtained auditive spectra than by the conventional mel-cepstra. One project is currently going on to test the auditive spectra for ASR.

18.5 Continuous Density Phoneme Models

The HMM structure has been a subject for a continuous development throughout the history of this work. The basic idea has been the simple temporal structure of uni-directional chains without skips (see Figure 30) and the principle of using one HMM for each of the 22 common Finnish phonemes including the silences directly before and after the word. For the output density of the states the building blocks have been Gaussians with a shared diagonal covariance matrix. The currently best performing version (Table 8) applies phoneme-wise tied Gaussian codebooks (PWMHMM), where the mixture densities are shared so that the states representing the same phoneme use the same codebook [2]. Thus there are as many sets of Gaussians as there are HMMs, which is a kind of intermediate for the traditional continuous HMMs (CDHMM) (different set for each state) and semi-continuous HMMs (SCHMM) (only one large set of Gaussians). The tied Gaussians resembles the vector quantization codebook of DHMMs, except that the densities are smoothly overlapped rather than partitioned.

By the PWMHMMs (and MDHMMs in general) the recognition occurs so that for the feature vector of every time window, the K-best matching Gaussians are extracted for every codebook and used to determine the HMM state classification probabilities. The codebooks were estimated from the training data by SOM and LVQ based training methods to ensure both the smoothness of the mapping and the efficient discrimination between phonemes. The most probable state segmentation for the feature sequence is then revealed using the HMM state structure and finally the phonemes are extracted from the path. When comparing the performance of the different continuous density HMMs the Table 8 shows that the PWMHMMs provide clearly the most appealing configurations, when the number of parameters, the recognition time and the error rate are compared.

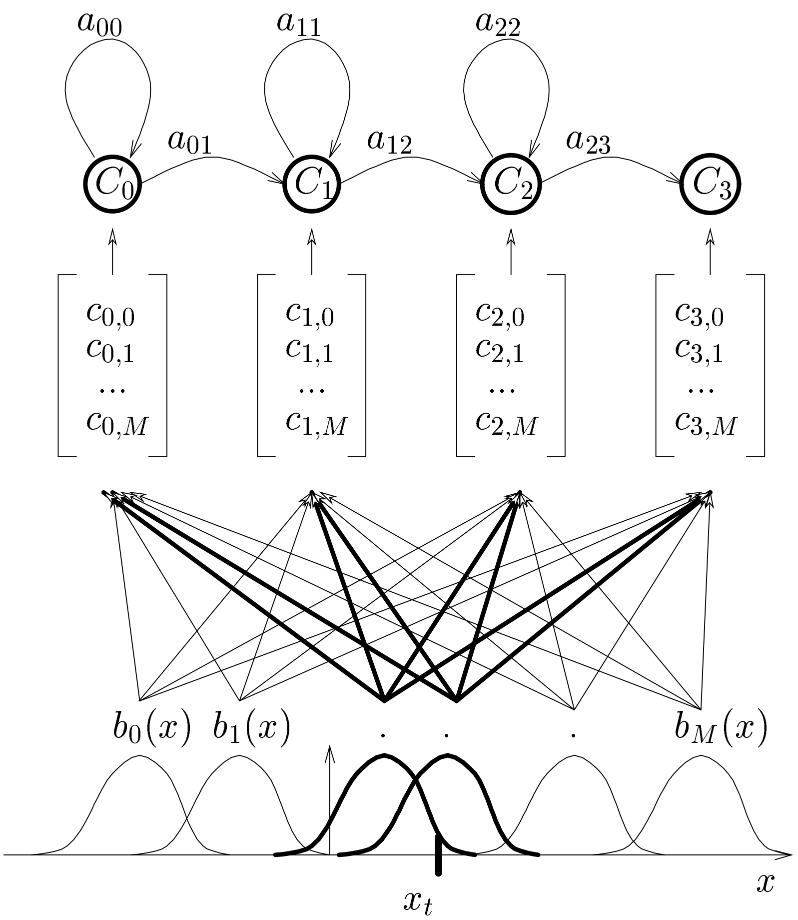

Figure 30: In phoneme-wise tied MDHMMs the same mixtures are used for the states representing the same phoneme. The model is defined by the set of transition probabilities a_{ij}, mixture weights $c_{i,k}$ and mixture densities $b_k(x)$. The output probability of state C_i at time t is approximated by using only the best matching mixture densities for the current observation vector x_t.

18.6 Discrete HMMs and Long Context Vectors using CNAPS

The experiments with high-dimensional context vectors [4] showed that the static recognition accuracy of separate phoneme tokens can be increased from 87% to 99% by substituting the single 20 dimensional cepstra by a 140 dimensional context vector and increasing the LVQ codebook size from 500 to 2000 units.
Since the winner search for the extended system was to slow to perform online in a normal workstation, the context vectors and the codebooks were sent to CNAPS parallel computer with 512 processing nodes. The search results are returned to the workstation for the HMM probability computations and the decoding of the most probable state sequence. To improve the HMM performance during the unstable transition parts between the phonemes, information from another LVQ codebook is also fed to the HMMs. The input for this codebook is the same as for the other codebook, but the classification task is to discriminate between phoneme centers and transition parts.

Type of HMM	Number of mixtures	Parameters ($x10^3$) weights	means	Recognition time factor	error%
CDHMM	4	0.4	9	0.7	7.9
PWMHMM	24	3	11	0.7	6.9
SCHMM	494	54	10	2.6	8.1
CDHMM	24	3	55	3.1	5.8
PWMHMM	70	8	32	2.1	5.7

Table 8: Some comparisons between different continuous density HMM structures. PWMHMM refers to the phoneme-wise tied mixture density HMMs. The CDHMM and SCHMM experiments are made using the same speech database (Data 90) and corresponding training methods. The recognition time factor is the average recognition time per word divided by that of the baseline system (24 mixture PWMHMM).

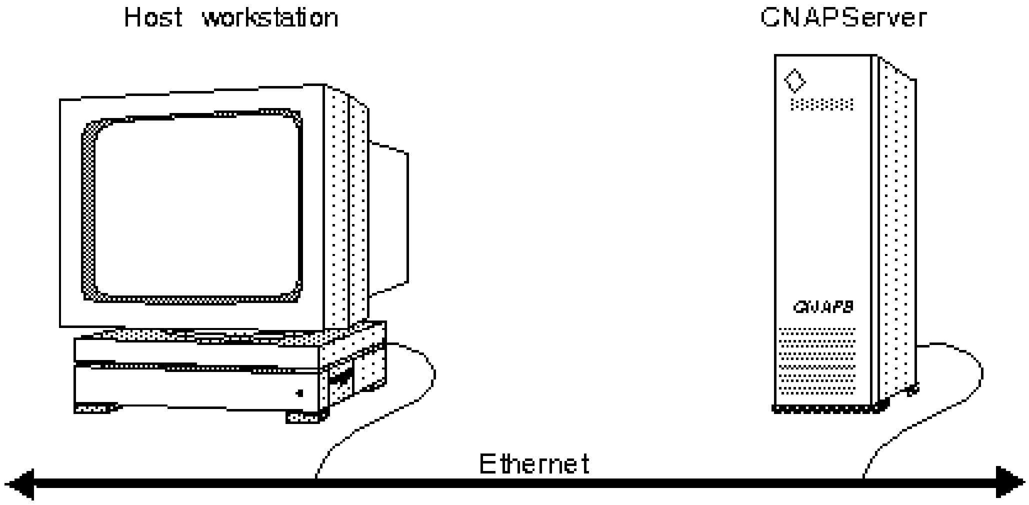

Figure 31: The CNAPServer System.

In the tested system the CNAPServer performed parallel computations under the control of a host workstation as shown in Figure 31. The context vectors are computed in the normal workstation and then transferred in buffers to the CNAPS. The information about the winner nodes is returned back to the workstation, which computes the HMM state classification probabilities and finally shows the decoded phoneme strings. However, despite the excellent off-line recognition results with DHMMs and the two large LVQ codebooks [4], the system working with the CNAPS had some problems in the fluent online operation.

18.7 CNAPS/PC Board in the Continuous Density System

While it was observed that the high-dimensional context vectors can also improve the performance of the new MDHMM system (see section 18.4), a CNAPS/PC board was tested to overcome the bottle neck occurring in the search of K-best match for all phoneme codebooks. The CNAPS/PC board is an ISA bus board that implements the CNAPS architecture for 128 processing nodes. The motivation for this work was also to see, whether there would be any major problems in transferring the workstation based system into the PC with a special board. At least the data communication was expected to be much simpler and faster than between the workstation and the CNAPServer.

The computations between the PC and the CNAPS are divided so that after the collection and formation of each context vector, the CNAPS immediately finds out the responses and the indexes of the K-best matching Gaussians. The PC then computes the state-dependent weighted sums that approximate the HMM state classification probabilities and takes care of the remaining phoneme decoding. So the CNAPS/PC board actually performs only a part of the third phase from whole the recognition process (see Figure 28), but this is the part that would otherwise take over 50% of total recognition time.

As a result of the study, a demonstration system operating with Linux PC using 80-dimensional context vectors processed by the CNAPS/PC board was able to perform the current ASR task smoothly online. The otherwise excessive codebook transfers were reduced by performing the parallel search on all phoneme codebooks in one operation, so that there is no need to change the codebooks in the processor memory. The size of the memory restricts, however, the use of larger codebooks and feature vectors. Due to the rapid capacity improvements in the general-purpose workstations, the corresponding ASR task is now processed online also in the 1997 recognition system without any special hardware.

18.8 Post-processing of Output Strings

If the set of possible output strings is known, the recognition error rate can be considerably reduced, even if the set is very large (e.g. 100 000 words). Successful post-processing can be applied as well for an open string set, if the correct strings are given corresponding to a set of evaluation samples.

A vocabulary-independent post-processing system for HMM based recognizers is shown in Figure 32. First it extracts the N best matching result strings using the mixture density hidden Markov models (HMMs) [3] trained by neural networks. Then the strings are corrected by the rules generated automatically by the Dynamically Expanding Context (DEC) [1]. Finally, the corrected string candidates and the extra alternatives proposed by the DEC are ranked according to the likelihood score of the best HMM path to generate those strings.

The objective of the system is to improve the HMM result strings so that the final result would be the best string allowed by the DEC rules. Since it is difficult to directly take care of the DEC rule base during the HMM decoding, the task is approached by transforming all the best HMM string candidates by the DEC. The ranking of the transformed strings is obtained by using another HMM decoding pass

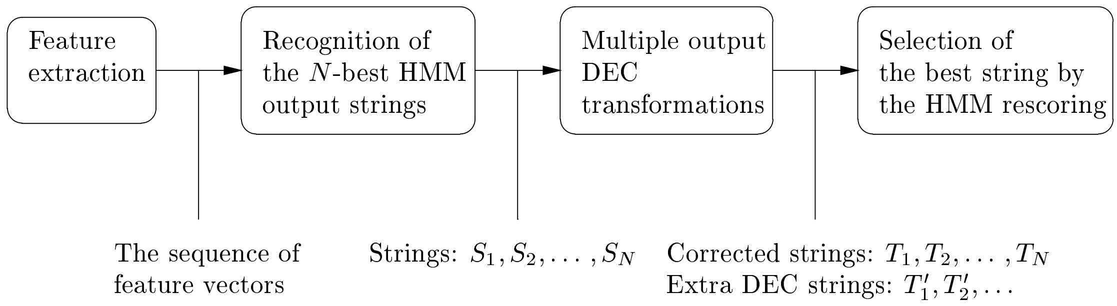

Figure 32: The stages of the N-best HMM-DEC decoding and the information that is transmitted between the stages.

which is this time restricted to the closed set of candidate strings. The experiments show that N need not be very large and the method can decrease recognition errors from a test data that even has no common words with the training data of the speech recognizer.

If the set of acceptable strings (the vocabulary) is available for the post-processing, a fast and efficient method based on Redundant Hash Addressing (RHA) [2] and two successive HMM decoding passes can be applied. First HMM decoding pass is made unconstrained and it provides one or more best-matching phoneme strings. Using these strings as keys for hashing, the large vocabulary can be quickly reduced by RHA so that only strings close enough to the keys will remain. The second, closed vocabulary HMM decoding pass can be then made in real time and a good approximation of the best matching acceptable string is found.

18.9 Other Activities

A separate report is given for the performance evaluation for a telephone based ASR application. Also the results from the developments of the neural network based training algorithms and fast density approximation methods are provided separately.

References

[1] T. Kohonen. Dynamically expanding context, with application to the correction of symbol strings in recognition of continuous speech. In *Proceedings of the 8th International Conference on Pattern Recognition*, pages 1148–1151, Paris, France, 1986.

[2] T. Kohonen and E. Reuhkala. A Very Fast Associative Method for the Recognition and Correction of Misspelt Words, Based on Redundant Hash-Addressing. In *Proceedings of the 4th International Conference on Pattern Recognition*, pages 807–809, Kyoto, Japan, 1978.

[3] M. Kurimo. Hybrid training method for tied mixture density hidden Markov models using Learning Vector Quantization and Viterbi estimation. In *Proceed-*

ings of the IEEE Workshop on Neural Networks for Signal Processing, pages 362–371, Ermioni, Greece, September 1994.

[4] M. Kurimo and P. Somervuo. Using the Self-Organizing Map to speed up the probability density estimation for speech recognition with mixture density HMMs. In *Proceedings of the International Conference on Spoken Language Processing*, volume 1, pages 358–361, Philadelphia, PA, USA, October 1996.

[5] J. Mäntysalo, K. Torkkola, and T. Kohonen. Mapping context dependent acoustic information into context independent form by LVQ. *Speech Communication*, 14(2):119–130, 1994.

[6] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.