Experiments on the use of LVQ in phoneme-level segmentation of speech

Jyri Mäntysalo†, Kari Torkkola‡, and Teuvo Kohonen†

†Helsinki University of Technology,
Laboratory of Information and Computer Science,
Rakentajanaukio 2 C, SF-02150, Espoo, FINLAND

‡IDIAP (Institut Dalle Molle d'Intelligence Artificielle Perceptive),
CP 609, CH-1920 Martigny, SWITZERLAND
email: karit@idiap.ch

Abstract

The problem that we address in the present paper is producing phonemic transcriptions of spoken utterances. We have applied the Learning Vector Quantization (LVQ) methods, including the latest developments [3, 5, 4, 6] in a novel way to derive information helpful to the task at hand. We have studied the effect of radically increasing the dimensionality of the context vectors classified into phoneme classes using LVQ. The high-dimensional feature vectors in our work represent the whole phoneme and they are formed by both averaging and concatenating short-time feature vectors within a time domain window. We report excellent results for classification of Finnish phonemes. Moreover, we have succesfully applied the same method to obtain additional information about phoneme center and border locations. Both phoneme classification and segmentation information are combined, together with codebook quantization error, using Hidden Markov models (HMM) with multiple parallel input streams. We obtained as low as three per cent phoneme level error rates for a speaker-dependent recognition task.

1 Introduction

In certain languages, such as Finnish, recognition of phonemes from the acoustic signal is a good basis for further processing, mainly due to a small number of phonemes, and due to a direct correspondence between orthography and phonemic transcriptions. A further attraction in producing a stream or a lattice of phonemes

lies in the enormously reduced bandwidth for later processing stages in speech recognition.

However, being abstract linguistic units, phonemes do not directly correspond to clearly distinct acoustic segments. The phonemes have varying lengths, they may consist of several partly overlapping acoustic segments, and their acoustic appearances vary according to their context, and also across speakers. Even the same speaker is not able to produce exactly the same acoustic signal for the same utterance. The task of transcribing speech into phonemes is thus anything but trivial. In this paper we address the aforementioned task for the Finnish language.

Theoretically, a single centralized classification decision is expected to be more accurate than any fusion of distributed local decisions due to information quantization effects. That is, in order to get best recognition performance, all the relevant information should be included into each recognition stage. Within the task of separate phoneme classification using the Learning Vector Quantization methods [3, 5, 4, 6], a natural way of applying this principle is to combine successive short-time feature vectors to form "context vectors". The high-dimensional feature vectors might then represent a whole phoneme. In this way, however, we obtain a very large number of different patterns that represent the same "phoneme". Their classification needs high computing capacities, which are provided by neural-network methods.

In this work, the pattern vectors are formed by both averaging and concatenating short-time feature vectors within a time domain window. Since there exist no known theoretical way of deciding the combination that will give the most accurate classification results with the given data, we have tried to find the best form experimentally.

Because classifying pre-segmented phoneme locations is a problem of an academic interest only, we have also extended our phoneme recognition experiments into combined labeling and segmentation of continuous speech. Our solution is to try to exploit vector quantization errors in addition to the phonemic classification results. Since the codebook vectors have been trained by samples from the phoneme centers, the locations of minima and maxima of quantization errors contain valuable information for segmentation. We also applied the LVQ to the segmentation task, whereby the context vectors are classified into two classes only, representing the stationary regions and transitions, respectively.

To combine the above mentioned information we have concentrated on an approach, where the classification results are interpreted as output symbols of discrete-observation HMMs with multiple observation streams [2]. The models can also include quantization errors as continuos-density probability distributions. It should be emphasized that in this work the objective has not been to create a real-time speech recognition system, since the high-dimensional context vectors require much more computation time than, for example, the system described in [9].

The rest of this paper is organized as follows. In Section 2 we describe the speech data and its parametrization. Section 3 presents the experiments with classification of known phoneme center locations. In Section 4 we extend the results into continuous speech, that is, into phonemic segmentation (or phoneme spotting) and labeling. Finally, in Section 5 we draw the conclusions.

2 Speech data and preprocessing

In the following experiments the same small Finnish speech database has been used as in [9]. It contains four repetitions of a set of 311 words spoken by three male Finnish speakers. Each set contains 1737 phonemes. In the separate phoneme recognition experiments the unvoiced plosives /k,p,t/ have been divided into two parts, representing the closure part and the plosive release, so that the actual number of pre-segmented phonemes in each set is now 1903. Three of the repetitions were used each time for training, and the remaining one for testing. Four independent runs were made by leaving one set at a time for testing, so that all the following recognition results are averages of 12 runs. As features computed every 10 ms we use 20-component mel-scale cepstra liftered with raised sine [9].

There are only 21 different phoneme classes in the original Finnish language. Four additional phonemes have been adopted with loan words from other languages, but our database did not contain any of those. It must be also noted that we consider each diphthong to be a concatenation of two vowels, rather than to be a phoneme of its own. This is in contrast to the common practice in phonemic speech recognition with, for example, the English language.

3 Applying LVQ for phoneme center locations

3.1 LVQ training procedure

We have first run an excessive amount of preliminary experiments to find good parameters to the Learning Vector Quantization methods [3, 5, 4] for this specific task. We used the public domain software package *Lvq_Pak* [6] throughout these experiments. The following learning scheme proved to give the best results on the average:

1. (Lvq_Pak: program 'eveninit') The codebook vectors are initialized with samples from teaching data, so that each class contains an identical number of codebook vectors. All the candidates are tested against all the other samples;

if the classification with the K-nearest-neighbor –method is correct, then the candidate will be accepted as a codebook vector initial value. This guarantees, that all the initial values are situated inside the correct class decision borders.

2. (Lvq_Pak: program 'balance') The medians of the shortest distances between the initial codebook vectors of each class are first computed. If the distances turn out to be very different for the different classes, new codebook vectors are added to or old ones deleted from the deviating classes, and a tentative training cycle based on the optimized-learning-rate LVQ1 algorithm is run once.

3. (Lvq_Pak: program 'olvq1') The codebook vectors are trained iteratively with the OLVQ1-algorithm using 10000 iterations and learning-rate initial values $\alpha_i(0) = 0.3$.

4. (Lvq_Pak: program 'lvq1') The codebook vectors are trained iteratively with the LVQ1-algorithm using learning-rate initial value $\alpha(0) = 0.2$ and decreasing it linearly during the learning process. Total number of iterations is 100 times the number of codebook vectors.

Four different codebook sizes, namely 200, 500, 1000 and 2000 units, were used.

3.2 Averaging experiments

We first studied experimentally the effect of pure averaging of the successive feature vectors starting from the pre-segmented phoneme centers. The recognition results are plotted in Fig. 1. The best results are obtained when the number of averaged feature vectors is approximately six. Thus the average width of the stable region of a phoneme can be approximated to be around 60 ms.

3.3 Concatenation experiments

Next, we have tested the effect of pure concatenation of successive short-time feature vectors into one context vector. Fig. 2 displays the results.

We have restricted our context vector dimension to be less than or equal to 240 because of computation time limitations. To include more context in time, we next try to concatenate every second short-time feature vector (Fig. 3).

Excluding every second short-time feature vector does not significantly decrease the accuracies. Even better results can be expected if we concatenate averages of two or more successive feature vectors.

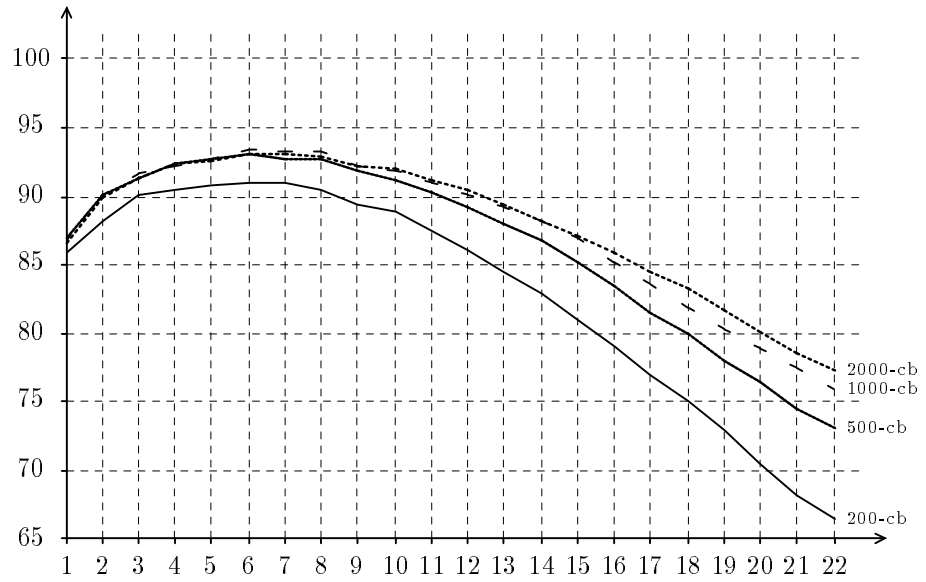

Figure 1: *Recognition accuracies as a function of the number of averaged short-time feature vectors.*

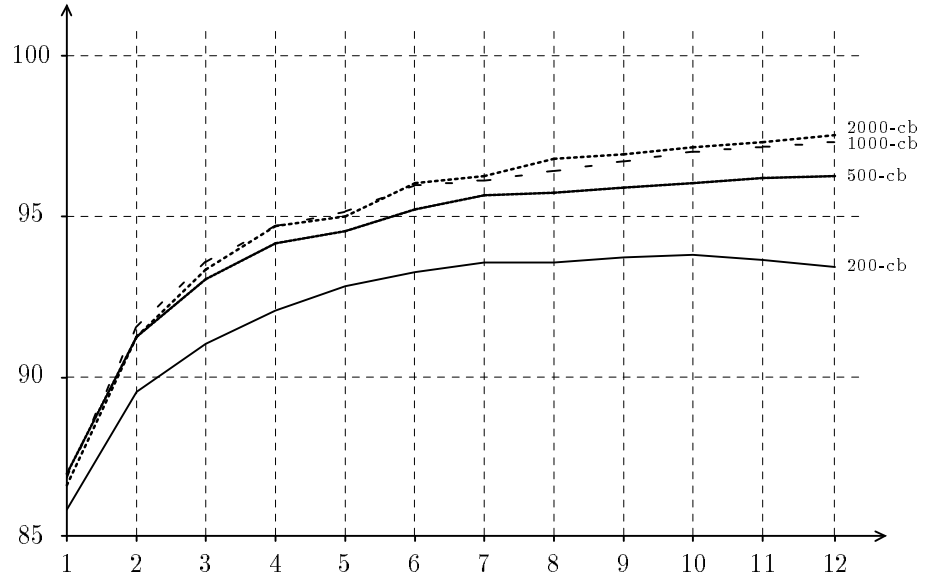

Figure 2: *Recognition accuracies as a function of the number of concatenated successive short-time feature vectors.*

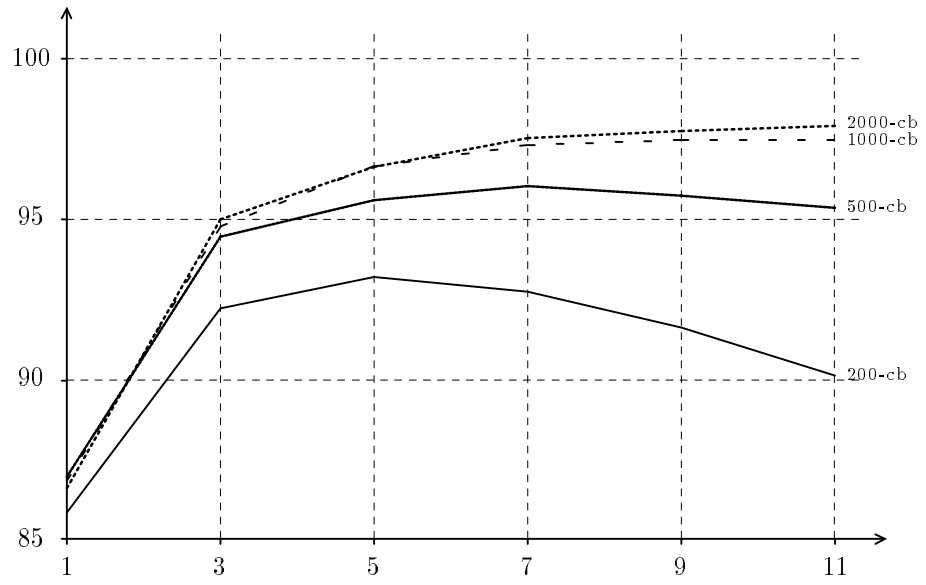

Figure 3: *Recognition accuracies as a function of the number of concatenated short-time feature vectors – every second vector is excluded.*

3.4 Experiments with combined concatenation and averaging

Next we have tested the effect of combined concatenation and averaging of successive short-time feature vectors into one context vector. The first graph (Fig. 4) shows the concatenation results, when the number of averaged feature vectors is two.

In the next graph (Fig. 5) we have used averages of three feature vectors.

It is obvious, that outside the stable regions of the phonemes, speech signal contains more irrelevant information than in the phoneme centers due to coarticulation effects. Also, different durations of phonemes decrease the possibilities to exploit this information effectively. Thus it seems reasonable to use wider averages outside the stable regions. Table 1 lists the numerical values of the new experiments, where averages of four have been tried. For comparison, some previous results are included.

3.5 Experiments with less training data

It would be interesting to know, whether the previous results could be still improved by increasing the amount of training data. Since no more data has been currently

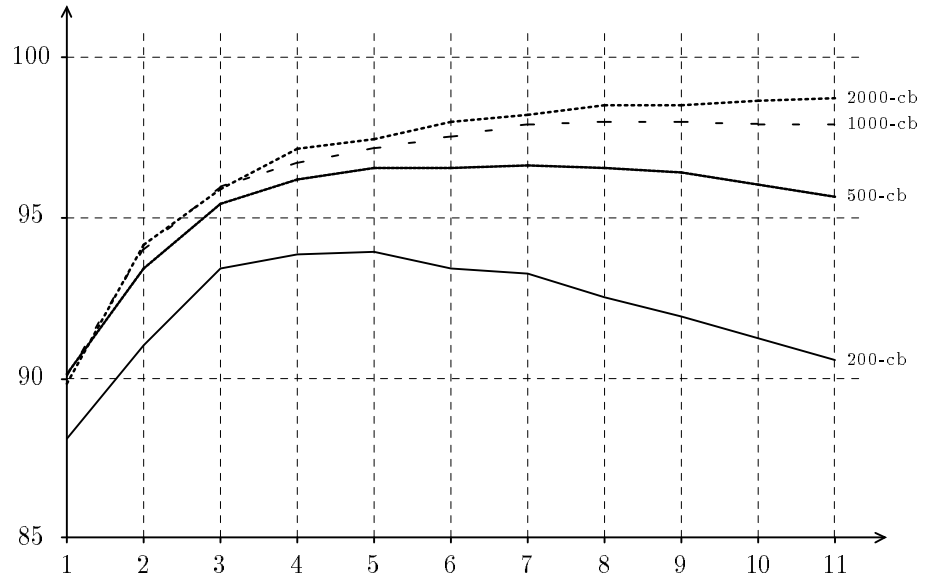

Figure 4: *Recognition accuracies as a function of the number of concatenated averages of two short-time feature vectors.*

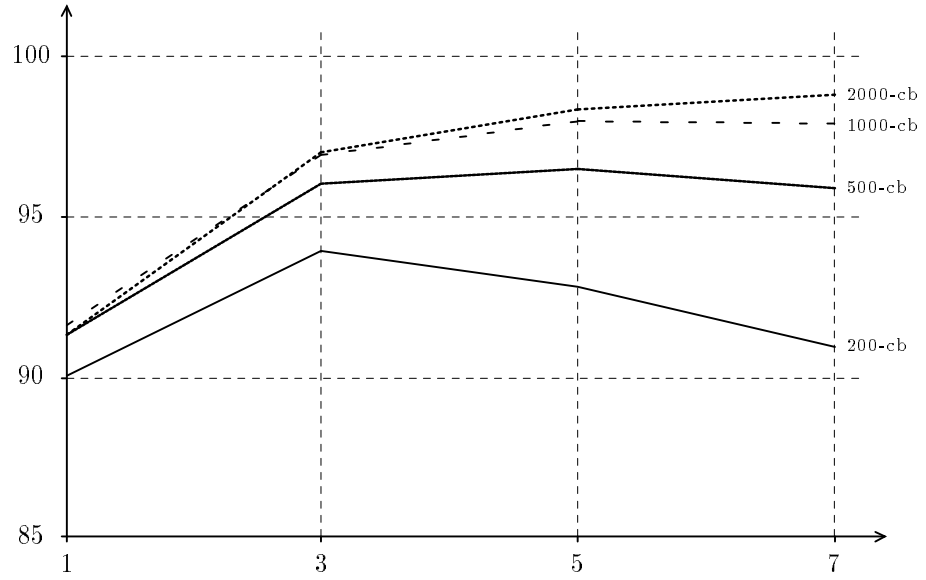

Figure 5: *Recognition accuracies as a function of the number of concatenated averages of three short-time feature vectors.*

Context vector	dim.	cb:500 correct%	cb:1000 correct%	cb:2000 correct%
I	20	86.96	86.87	86.61
[....]	20	93.01	93.43	93.12
IIIIII	120	95.23	95.97	96.04
[][][]	60	95.49	95.96	95.93
[..][][][][..]	100	97.03	97.93	98.22
[..][..][][][][..][..]	140	97.13	98.50	98.95
[][][][][][][][][][][]	220	95.71	97.91	98.76
[.][.][.][.][.][.][.]	140	95.89	97.90	98.81

Table 1: *Numerical separate phoneme recognition results.* `I` *is a single short-time feature vector,* `[]` *means average of two,* `[.]` *average of three,* `[..]` *average of four, and so on. For example,* `[..][]` *means an average of four feature vectors concatenated with an average of two.*

available, we tested the effect of reduced set of training data to see if it affects to the accuracies. The results show an approximate decrease of a half percent unit, when two thirds of the original training data was used. A rough extrapolation of these results to the case, in which more training data would be available, gives hope for some improvements.

4 Continuous speech recognition

4.1 What information is available for segmentation?

In this section we address the obvious question that follows the previously described successful experiments: Although high accuracies can be obtained by classifying phonemes when their locations are known beforehand, are these context vectors applicable in combined segmentation and labeling, that is, transcribing speech into phoneme sequences?

Our solution to this question is to slide the time domain window, that covers a period of of 220 milliseconds, (Fig. 6) over all of the speech signal and to compute phoneme classification by LVQ for its center position. This results in a so-called quasiphoneme sequence with a phoneme symbol every 10 ms. The idea of our segmentation procedure is based on the following two facts:

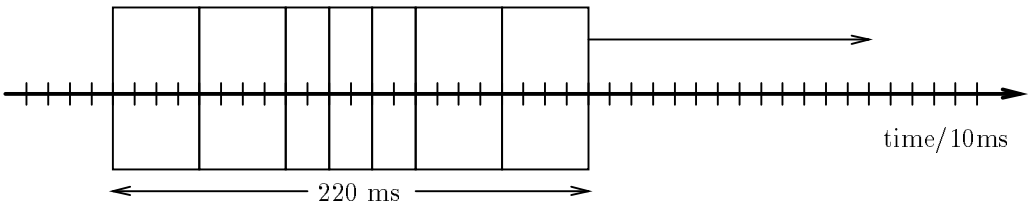

Figure 6: *The selected 140-dimensional context vector*

- The context vector covers a relatively long period of time domain.
- The LVQ-codebook is trained using phoneme centers only.

We can thus assume that transitions, since they are represented as long context vectors, will appear substantially different from center parts of phonemes. This will in turn result in high quantization errors at phoneme borders, since the codebook vectors represent phoneme centers. This error can then be used as an aid to segmentation in addition to the quasiphoneme sequence obtained as the stream of classification results from the codebook.

To obtain even more explicit information about phoneme locations, we have experimented with training another codebook for two classes only, one for the phoneme centers and the other for transitions between phonemes. Figure 7 illustrates all this information available for segmentation. For this codebook, the context vector form and the number of codevectors were kept the same as in the most successfull phoneme recognition experiment, i.e., the context vector of Fig. 6, and 2000 codevectors. The codebook is trained using exactly the same procedure, but with examples from phoneme centers and phoneme borders. The accuracy we obtained in this phoneme/border classification problem was 94.6%. This segmentation task really brings forward the advantage of using context windows. As expected, using only short-time cepstral vectors, the accuracy was as low as 75.4%. Table 2 displays these results with different context vector forms and codebook sizes.

The remaining problem, how to combine all this information to produce phonemic transcriptions of uttered speech, will be addressed in the next subsection.

4.2 HMM-Based Decoding

In our approach, the above mentioned classification results are interpreted as output symbols of a discrete observation HMM. The details of the basic configuration can be found in [9]. The system models each phoneme as a separate Markov source. A simple four-state left-to-right topology has been used. Output symbol distributions

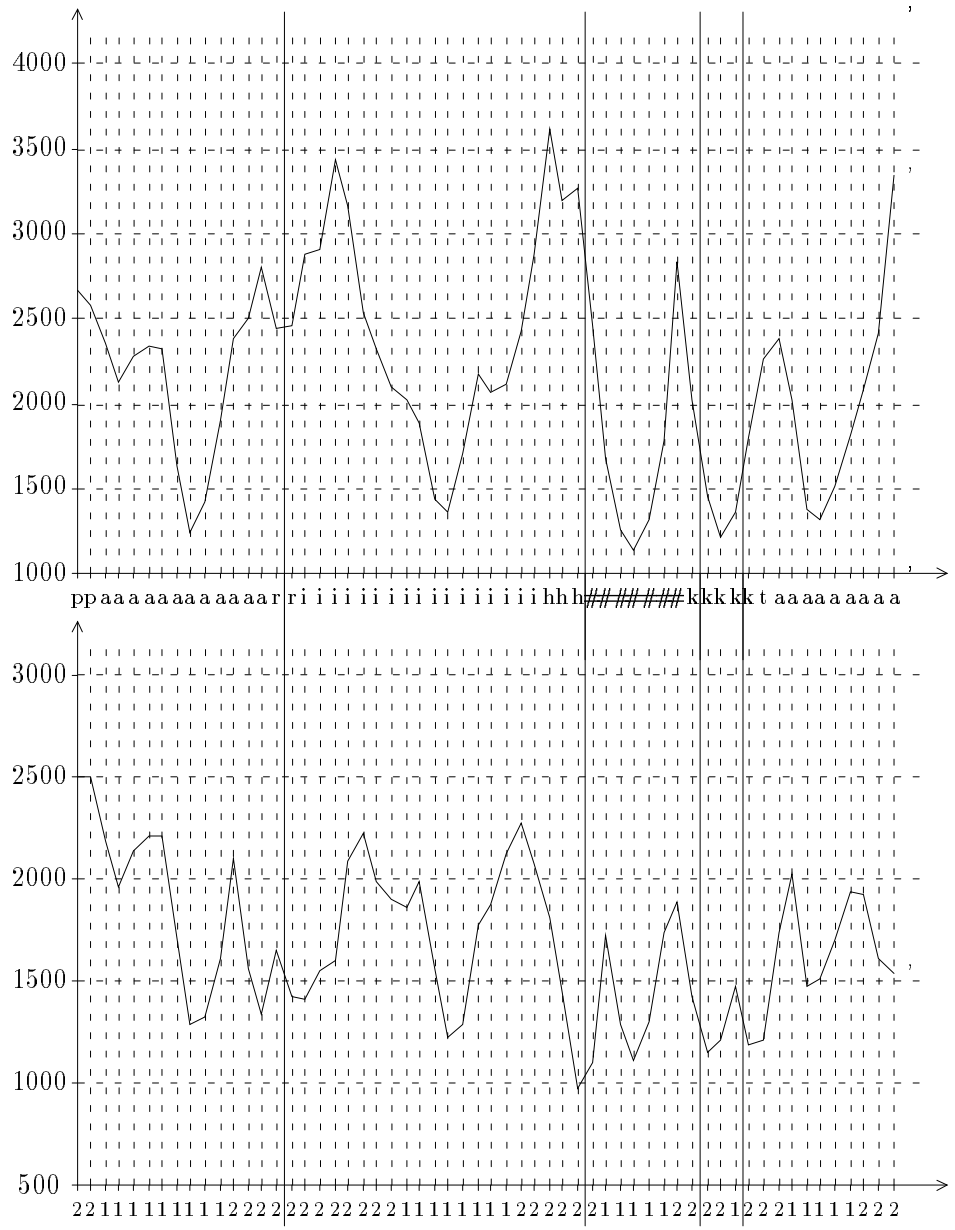

Figure 7: *Segmentation information for the Finnish word 'aika'. The top graph displays the recognized quasiphonemes together with their quantization errors. The manual segmentations (not used in the experiments) are shown as vertical lines (the phonemes /a/, /i/, /k/, /a/). The sign '#' denotes silence. Respective segmentation classifications are shown in the lower graph. '1' and '2' stand for stationary regions and transitions, respectively. Note that the diphthong /ai/ is here considered as a concatenation of two vowels.*

Context vector	time span (ms)	dim.	cb:500 correct%	cb:2000 correct%
I	10	20	75.44	73.83
[]	20	20	77.02	76.05
[.]	30	20	80.78	80.52
[][][]	60	60	89.84	91.84
[.][.][.]	120	60	90.53	92.87
[][][][][]	100	60	91.52	93.93
[.][.][.][.][.]	150	100	91.27	93.97
[..][][][][..]	140	100	92.34	94.46
[..][..][][][][..][..]	220	140	91.94	94.63

Table 2: *The accuracy of phoneme center/border classification. Context vector forms are axplained in the caption of Table 1.*

have been defined for state transitions within the phoneme models and they are trained using the standard maximum-likelihood approach [8].

Phoneme models are connected in parallel with empty transitions. The decoding algorithm is allowed to loop back from the final state of a phoneme model to the initial state of any other model. Probabilities for the transitions between phonemes were estimated separately from a large text corpus.

The main differences between a standard discrete observation (vector quantization) HMM–approach and our approach here are the following:

- LVQ-codebook is used for frame-synchronous phoneme *classification*, not to *represent* the speech signal with minimum distortion.
- Codebook quantization errors are included in the HMMs
- An additional codebook is used for segmentation

We are thus using at most three separate parallel streams in the HMMs. For the quantization errors, both 8-level quantized values and continuos-density probability distributions with large constant variance were tried.

In principle, the information of the separate observation streams ought to be statistically independent to allow observation probability combination by multiplication. Strictly speaking this is not the case in our experiments. In spite of this theoretical problem improvements were observed.

Method	correct%	errors%
HMM(STF)	93.86	9.50
HMM(FON)	97.07	4.58
HMM(FON, dQE)	97.03	4.14
HMM(FON, cQE)	97.64	3.78
HMM(FON, SEG)	98.43	3.15
HMM(FON, SEG, dQE)	98.09	3.15
HMM(FON, SEG, cQE)	98.43	2.94
HMM(FON/SEG)	98.50	3.02
HMM(FON/SEG, dQE)	98.04	3.18
HMM(FON/SEG, cQE)	98.49	2.84

Table 3: *Experimental results with different HMM-codebook variations. 'FON' stands for quasiphoneme codebook, 'SEG' segmentation-class codebook, 'dQE' and 'cQE' quasiphoneme quantization error codebooks in discrete and continuos case, respectively. 'FON/SEG' means that these sequences have been combined to form one larger codebook. For comparison, results obtained by using a short-time feature vector based quasiphoneme codebook ('STF') are given. The 'errors'-column includes all substitution, deletion and insertion errors.*

Table 3 shows the final results of the experiments divided in four main groups:

1. Using traditional short-time cepstral vectors as the basis of short-time phoneme classification. The codebook size was 500.

2. Using high-dimensional context vectors and a large LVQ-codebook of 2000 codevectors for short-time phoneme classification. Quantization error of this codebook can be modeled as separate stream, either as a discrete or a continuous density distribution.

3. In addition to the above, using a large segmentation LVQ-codebook also with high-dimensional context vectors to produce a third parallel stream of information that consists of only two possible symbols.

4. Combine the two-stream information from the two above mentioned large codebooks into a single stream in the following way. For each phoneme, there will be two alternative symbols, a symbol for the center part, which is outputted whenever the segmentation codebook produces the center symbol, and a symbol for the transitional part, outputted whenever the segmentation codebook produces the transition symbol.

It should be mentioned that the manually segmented starting and ending points of the spoken words are supposed to be known here. In these experiments, we did not include yet another codebook for the energy of the speech signal, which would have facilitated beginning and endpoint detection.

5 Discussion

In this research we have applied the Learning Vector Quantization methods, including the latest developments [3, 5, 4, 6], to the task of Finnish speaker-dependent speech recognition. Our aim was to produce phonemic transcriptions of spoken utterances. We experimented with various ways to form a high-dimensional context vector by both averaging and concatenating short-time feature vectors within a time domain window. This method of taking the local phonemic context into account was shown to give excellent results when applied to the separate phoneme classification of Finnish speech. When compared to the case, in which one short-time feature vector was used, the classification accuracies increased from 87 % up to 99 % with the best form of context vector. It is also interesting to note the positive effect that increasing LVQ-codebook sizes has to the recognition results with high-dimensional context vectors. It seems to be obvious that this way we are able to capture some context-dependent phoneme variations in the large codebook automatically.

We also applied the LVQ methods with context vectors to phoneme border/center detection task and exploited this additional two-class time-series information successfully by using it as a separate HMM codebook, or as a combined codebook with the quasiphoneme sequence. We also tried to utilize the quantization errors, but they proved to give no significant additional improvement.

In a way, our approach resembles segmental approaches [7] [1] in which the segment (for example, phone or phoneme) classification is based on the whole duration of the segment. Speech is considered as a concatenation of such segments, which is found out by Viterbi-decoding. Segment boundaries may then be suggested to Viterbi-decoding by other means. In our case the decoding process is simpler, since we perform the segment classification frame-synchronously using fixed-time windows. However, this is where the computational burden now resides.

The lowest error percentage that we obtained was approximately 3%, when the starting and ending points of words were supposed to be known. This figure shows a significant improvement when compared to results obtained in real-time operation, based on separate classification of short-time feature vectors [9]. Naturally, the computing load is now multiplied, calling for massively parallel hardware.

References

[1] Y.M. Cheng, D. O'Shaughnessy, V. Gupta, P. Kenny, M. Lenning, P. Mermelstein, and S. Parthasarathy. Hybrid segmental-lvq/hmm for large vocabulary speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP92)*, volume I, pages 593–596, San Francisco, CA, USA, March 23-36 1992.

[2] V. N. Gupta, M. Lennig, and P. Mermelstein. Integration of acoustic information in a large vocabulary word recognizer. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP87)*, volume 2, pages 697–700, Dallas, Texas, April 1987.

[3] Teuvo Kohonen. *Self-Organization and Associative Memory.* Springer-Verlag, Berlin-Heidelberg-New York-Tokio, 3 edition, 1989.

[4] Teuvo Kohonen. Improved versions of learning vector quantization. In *Proceedings of the International Joint Conference on Neural Networks*, pages I 545–550, San Diego, June 1990.

[5] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[6] Teuvo Kohonen, Jari Kangas, Jorma Laaksonen, and Kari Torkkola. LVQ_PAK: A program package for the correct application of Learning Vector Quantization algorithms. In *Proceedings of the International Joint Conference on Neural Networks*, pages I 725–730, Baltimore, June 1992. IEEE.

[7] H.C. Leung, I.L. Hetherington, and V.W. Zue. Speech recognition using stochastic segmental neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP92)*, volume I, pages 613–616, San Francisco, CA, USA, March 23-36 1992.

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

[9] 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 (ICANN-91)*, pages 771–776, Espoo, Finland, June 24-28 1991.