Published in the proceedings of ICANN-91, Espoo, Finland, June 24-28, 1991, pp. 771-776

STATUS REPORT OF THE FINNISH PHONETIC TYPEWRITER PROJECT

Kari Torkkola, Jari Kangas, Pekka Utela, Sami Kaski, Mikko Kokkonen, Mikko Kurimo and Teuvo Kohonen

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

Abstract

In connection to a speech recognizer, the aim of which is to produce phonemic transcriptions of arbitrary spoken utterances, we investigate the combined effect of several improvements at different stages of phoneme recognition.

The core of the basic recognition system is Learning Vector Quantization (LVQ1) [1]. This algorithm was originally used to classify FFT-based short-time feature vectors into phonemic classes. The phonemic decoding stage was earlier based on simple durational rules [2] [3].

At the feature level, we now study the effect of using mel-scale cepstral features and concatenating consecutive feature vectors to include context. At the output of vector quantization, a comparison of three approaches to take into account the classifications of feature vectors in local context is presented. The rule-based phonemic decoding is compared to decoding employing Hidden Markov Models (HMMs). As earlier, an optional grammatical post-correction method (DEC) is applied.

Experiments conducted with three male speakers indicate that it is possible to increase significantly the phonemic transcription accuracy of the previous configuration. By using appropriately liftered cepstra, concatenating three adjacent feature vectors, and using HMM-based phonemic decoding, the error rate can be decreased from 14.0 % to 5.8 %.

1 Introduction

The speech recognition task we have concentrated on has been to transcribe spoken Finnish into text. We have chosen phonemes as the basic recognition units, which can be justified by the following characteristics of the Finnish language: 1) it contains only 21 phonemes, 2) the correspondence of written and spoken forms is almost identical, and 3) the words are highly inflected, for example, a single verb can have over 1000 different inflected forms. The choice of phonemes has further been favorable in this work, since our main objective has been to write out text from *arbitrary* but carefully articulated dictation, whereby the prosodic features and coarticulation effects can be kept at minimum. The methods described in this paper are also applicable to other languages having similar characteristics, e.g., Japanese [4].

Our earlier system [4] [5], which operated in real time, was designed around a simple but reasonably effective co-processor board of our own construction. Its main features were: extraction of spectral features by FFT, their conversion into quasiphonemes (phonemic labels) every 10 ms, decoding of the quasiphoneme sequences into phonemes by simple merging rules (cf. Sec. 4.1), and compensation of the remaining coarticulation effects and other systematic errors by a grammar-like algorithm (DEC, cf. Sec. 5).

The system used in the present work was built for a testbed, in which various algorithms and their combinations can be benchmarked flexibly. The computing time has been of no primary concern; the tests were run in Silicon Graphics Iris workstations. In order to implement on-line demonstrations, we have used for preprocessing (including feature extraxtion) the Loughborough Sound Images PCS/320C30 co-processor board. All demonstrations can now be performed in a few times real time. Fig. 1 illustrates the system configuration; its blocks are described in due context.

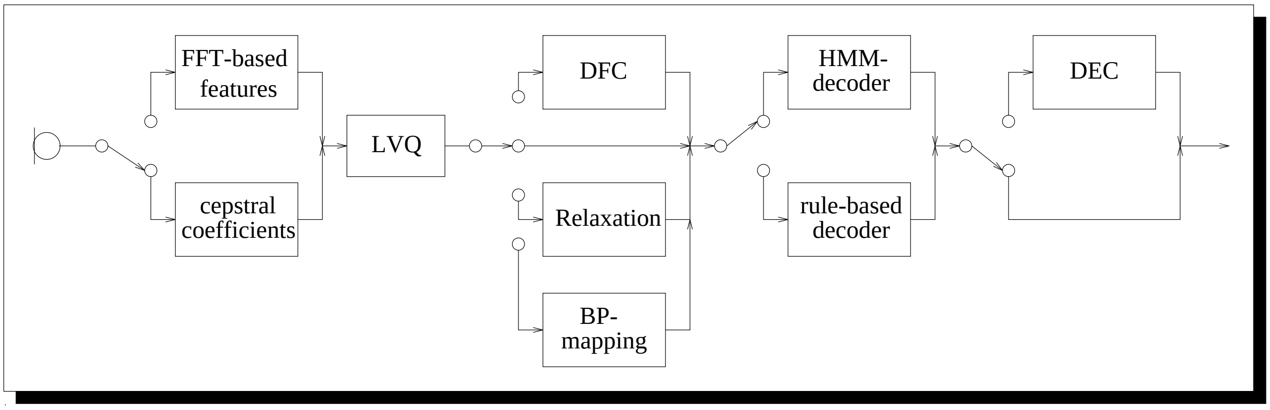

Figure 1: *The block diagram of the system.*

2 Feature Computation

For feature computation, the speech signal is first digitized at a rate of 12.8 kHz and transformed by a 256-point FFT every 10 ms using a 256-point (20 ms) Hamming window. 128-component logarithmic power spectra are then computed from the output of FFT.

For the *FFT-based features*, the outputs from the 128 channels are grouped into 15 feature components with 12 equally spaced points in the range from 200 Hz to 3 kHz and three points in the range from 3 to 5 kHz. This feature set was used in the original system [5].

The *mel-scale cepstral coefficients (MFCC)* are computed in the following way. First, 22 spectral components X_k are collected from the above 128 channels using triangular-shaped overlapping band-pass filters in the *mel-scale* [6] (here: 10 linearly spaced features below 1 kHz, 12 exponentially spaced features between 1 kHz and 5 kHz). Then

$$MFCC_i = \sum_{k=1}^{N} X_k \cos\left[i\left(k-\frac{1}{2}\right)\frac{\pi}{N}\right], i = 1, 2, \ldots, M, \tag{1}$$

where N is the number of filters and $M < N$. The coefficients were weighted in order to reduce undesired variability in the speech spectrum (called liftering) [7]. 20-coefficient cepstra liftered with raised sine yielded the highest accuracies.

3 Feature Vector Classification

The original feature vector classification method described in [3] and [5] was retained in this work. It is a supervised adaptive vector quantizer based on the LVQ1-algorithm [1], which produces a quasiphoneme label every 10 ms. In reality there are two "codebooks" for phonemes: one for all phonemes including the closure parts of /k/, /p/, and /t/ as one class /#/, and another "codebook" for the bursts of /k/, /p/, /t/ and the glottal stop separately. In the simple decoding method described in Sec. 4.1, the latter overwrites its outputs to the places /#/ marked by the former.

Since speech is dynamic in nature, classification of short-time feature vectors cannot be expected to yield the best possible quasiphoneme sequences. We made experiments in concatenating several consecutive feature vectors to longer "context vectors", and classified such vectors into phoneme classes. Table 1 shows the preliminary experiments for three speakers with manually centered context "windows". These experiments were performed using the same vector quantizer (LVQ1) as in Section 6.

In the continuous-speech recognition mode, a concatenated vector was formed every 10 ms on the basis of the adjacent feature vectors.

features	concatenation	jk	kt	pu	average
FFT-based spectral	1	88.8	91.2	88.6	89.5
Mel-scale	1	90.0	93.5	90.9	91.5
cepstral	3	91.6	94.8	92.0	92.8
features	5	89.0	92.7	90.1	90.6

Table 1: *Isolated phoneme recognition accuracies for three speakers (jk, kt, and pu). Unvoiced plosives were treated as one class; there were thus 19 different phonemic classes.*

4 Phonemic Decoding

4.1 Decoding Based on Heuristic Durational Rules

In the original decoding scheme presented in [2] and [3], each phoneme c has a window of length n_c sliding over the quasiphoneme sequence symbol by symbol. If m_c quasiphoneme labels in the window are equal to c, this phoneme c will be detected. The parameters n_c and m_c are phoneme and speaker dependent, but certain average default values may already yield a reasonably good recognition accuracy.

The quasiphoneme strings computed from a *new speaker's* collected speech material are first converted into phonemic transcriptions using the default rules. The obtained transcriptions are aligned with the correct ones, and a confusion matrix is constructed to display the frequencies of different replacement, insertion, and deletion errors. If, for example, the frequency of deletion errors is high for phoneme c, its durational constraints are relieved. This means that either n_c is increased or m_c is decreased. On the other hand, if insertion errors are very frequent for a phoneme, its durational constraints are tightened. This process is repeated a few times. There are additional durational rules to decide when a long (double) phoneme has been pronounced, and rules to determine how close to each other the detected phonemes are allowed to be.

4.2 HMM-Based Decoding

The quasiphonemes can be interpreted as output symbols of a discrete-observation hidden Markov model. The present system first models each phoneme as a separate Markov source (cf. [8] and [9]). The simple topology shown on the left-hand side of Fig. 2 was found to give good results. Three separate codebooks were used, specially in order to enhance the recognition of unvoiced plosives. The main LVQ1-codebook is trained using all phonemes except unvoiced plosives. An additional LVQ1-codebook is trained by unvoiced plosives only using three concatenated feature vectors. The third codebook represents the power of the speech signal. Thus each transition in the model has three distributions of discrete symbols. These models are trained using the standard maximum-likelihood approach [9]. Phoneme models are then connected in parallel with empty transitions (i.e. transitions that do not produce output symbols). Initial and final states are also added to the combined model. The decoding algorithm is allowed to loop back from the final state of a phoneme model to the initial state of any other model. Thus the number of consecutive phonemes is not fixed, and the model can be used to decode a sequence of any length. Probabilities for the transitions between phonemes were estimated separately from a large text corpus.

The Viterbi-search [9] is used in decoding quasiphoneme strings. This algorithm finds the most probable state path in the combined model. As a result, the phoneme models along this path give the most likely phoneme sequence. Durations of states in phoneme models are also recorded during the search. They are used in distinguishing long phonemes from the short ones by comparing the durations to individual threshold values. This is an important distinction because in Finnish about ten percent of the phonemes appear as prolonged, and in the orthography there are two identical successive symbols to denote it.

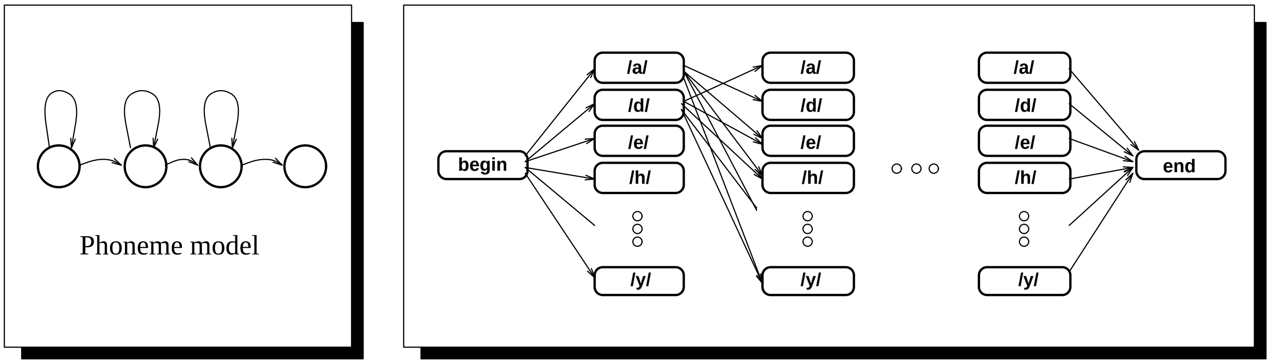

Figure 2: The structure of the phoneme model and the combined model in HMM decoding.

5 Taking Classification of Local Context into Account

At phoneme level the coarticulation effects and other systematic errors can be handled by the *Dynamically Expanding Context (DEC)* [10]. The central idea is to derive unique symbol to symbol(s) transformation rules from two streams of symbols: the source stream, and the desired target stream. In speech recognition the source stream is the phonemic transcription produced by either of the phonemic decoders, and the target stream is the correct transcription, or the orthographic form of the utterance.

The algorithm starts by creating initial context-free transformation rules from the aligned two streams. The initial rules are formed by segmenting the aligned streams by symbols. One segment of the source stream acts as the condition part of a rule, and the corresponding segment of the target string constitutes the production part of the rule. Now, these initial rules may conflict each other, which means that two rules with the same condition part may have different productions. These cases are specialized by dynamically expanding the discrete symbolic context around the initial condition parts by just a sufficient amount to make the condition part of the rules unique. The rules will then have condition parts of variable length. In this way the generality of the rules will be optimum, and still, all the conflicts within the training cases are solved. For details of the algorithm, see [10]. In previous experiments, the DEC (with much greater amount of training data than used here) was able to correct 50-70% of the errors remaining in the phonemic transcriptions.

The same kind of mapping or transformation can also be done already at the LVQ1-code sequence level. The idea is to transform the sequences deteriorated by coarticulation effects and by an unideal acoustic processor closer to ideal sequences. This facilitates easier and more accurate decoding by durational rules. The method called *Dynamically Focusing Context (DFC)* [11], [12] was developed for this stage. It is a derivative of the DEC modified to better suit to data that is more fine-grained than phoneme sequences.

The other mapping method studied here employs a *feed-forward network* with one hidden layer trained using *error back-propagation*. The network uses symbolic multiresolution input derived from the code sequence [12].

Relaxation labeling is a method widely used in image analysis to improve the consistency of contradictory or inconsistent classification of adjacent pixels [13]. We have applied it to quasiphoneme data, after having derived probabilistic classification information from the LVQ1.

6 Experiments

Extensive experiments were performed for three male Finnish speakers in the speaker-dependent mode. For each speaker, four repetitions of a set of 311 words were used. Each set contained 1737 phonemes. Three of the repetitions were used for training, and the remaining one for testing. Four independent runs were made by leaving one set at a time for testing. The results displayed in Table 2 are averages of four runs for each of the three speakers, that is, twelve runs. The figures refer to the accuracy of transcribing speech into phoneme strings, not to classifying pre-segmented phoneme positions. Errors occurring in the detection of the beginning and the end of the utterance are also included. The main codebook contained 216 vectors, and the /k,p,t/-codebook 100 vectors.

Combi-	Features		Concat.			QP-mapping			Decoding		Correction	Results	
nation	FFT	Cepstra	1	3	5	DFC	BP	Relax.	rules	HMM	DEC	correct	errors
1	√		√						√			82.9	22.6
2	√		√						√		√	89.5	14.0
3	√		√							√		90.1	12.7
4		√	√							√		92.7	9.9
5		√		√						√		93.1	9.4
6		√			√					√		91.5	9.9
7		√		√					√			86.4	19.9
8		√		√		√			√			89.0	15.5
9		√		√			√		√			89.8	14.6
10		√		√				√	√			86.6	18.0
11		√		√		√			√		√	94.1	9.0
12		√		√			√		√		√	94.7	8.1
13		√		√				√	√		√	93.5	8.9
14		√		√					√		√	92.7	11.2
15		√		√						√	√	96.0	5.8

Table 2: *The effect of combining different recognition methods to the phonemic transcription rate. Column 'correct' denotes the percentage of correctly recognized phonemes and column 'errors' denotes the percentage of the sum of deletion, replacement, and insertion errors, compared to the correct number of phonemes.*

Table 2 may be self-explanatory. The percentage of errors that the DEC is able to correct can be seen to vary between 40% and 50% in all of the cases. We could not run HMM-decoding combined with the two quasiphoneme mapping methods (DFC and BP) since we did not have enough data to train them independently. Relaxation labeling combined with HMM-decoding did not introduce any improvements.

7 Conclusion

We have tried several ways to improve the performance of a phonemic speech recognizer. The best combination in our experiments turned out to be the following: 20-component liftered cepstra used as features, three such vectors concatenated into a single context vector, classification into phoneme classes by LVQ1, phonemic decoding by HMMs, and correction at phoneme level with the DEC. The error rate dropped from 14.0% to 5.8% compared to our original combination (Row 2). An improvement close to that was attained by using the two quasiphoneme mapping methods or relaxation labeling with the rule-based decoding. Even better results could be expected if the HMMs were used in combination with these quasiphoneme mapping methods. Such experiments were not done here due to lack of training data.

It should be noted that the correctness figures we reported earlier in [5] were from the more complex Japanese version of our system. More training data was also used for those figures. The "old" system for Finnish to which rows 1-2 correspond was simpler.

References

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

[2] Teuvo Kohonen, Kai Mäkisara, and Tapio Saramäki. Phonotopic maps – insightful representation of phonological features for speech recognition. In *Proceedings of the 7th International Conference on Pattern Recognition (7th ICPR)*, pages 182–185, Montreal, Canada, July 1984.

[3] Teuvo Kohonen, Kari Torkkola, Makoto Shozakai, Jari Kangas, and Olli Ventä. Implementation of a large vocabulary speech recognizer and phonetic typewriter for Finnish and Japanese. In *Proceedings of the European Conference on Speech Technology*, pages 377–380, Edinburgh, U.K., September 1987.

[4] Teuvo Kohonen, Kari Torkkola, Makoto Shozakai, Jari Kangas, and Olli Ventä. Phonetic Typewriter for Finnish and Japanese. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP88)*, pages 607–610, New York City, USA, April 1988.

[5] Teuvo Kohonen. The 'Neural' Phonetic Typewriter. *IEEE Computer*, 21(3):11–22, March 1988.

[6] Stephen B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, August 1980.

[7] Yoh'ichi Tohkura. A weighted cepstral distance measure for speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(10):1414–1422, 1987.

[8] Anne-Marie Derouault. Context-dependent phonetic Markov models for large vocabulary speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP87)*, volume 1, pages 360–363, Dallas, Tx., April 6-9 1987.

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

[10] Teuvo Kohonen. Dynamically Expanding Context, with application to the correction of symbol strings in the recognition of continuous speech. In *Proceedings 8th International Conference on Pattern Recognition (8th ICPR)*, pages 1148–1151, Paris, France, Oct. 27-31 1986.

[11] Kari Torkkola. A combination of neural network and low level AI-techniques to transcribe speech into phonemes. In *Proceedings of COGNITIVA-90*, pages 637–644, Madrid, Spain, November 20-23 1990.

[12] Kari Torkkola and Mikko Kokkonen. A comparison of two methods to transcribe speech into phonemes: A rule-based method vs. back-propagation. In *Proceedings of 1990 International Conference on Spoken Language Processing (ICSLP90)*, volume 1, pages 673–676, Kobe, Japan, November 18-22 1990.

[13] Robert A. Hummel and Steven W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(3):267–287, May 1983.