VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

PHONEME RECOGNITION BASED ON LONG TEMPORAL CONTEXT

TITLE

DISERTAČNÍ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE AUTHOR	PETR SCHWARZ
VEDOUCÍ PRÁCE SUPERVISOR	JAN ČERNOCKÝ

BRNO 2008

Abstract

Techniques for automatic phoneme recognition from spoken speech are investigated. The goal is to extract as much information about phoneme from as long temporal context as possible. The Hidden Markov Model / Artificial Neural Network (HMM/ANN) hybrid system is used. At first, the Temporal Pattern (TRAP) system is implemented and compared to other systems based on conventional feature extraction techniques. The TRAP system is analyzed and simplified. Then a new Split Temporal Context (STC) system is proposed. The system reaches better results while the complexity was reduced. Then the system was improved using commonly used techniques such as three-state phoneme modelling and phonotactic language model. This system reaches 21.48 % phoneme error rate on the TIMIT database. The STC system was also studied on another databases, in noise and in cross-channel conditions. Finally few applications where the phoneme recognizer was applied are demonstrated.

Keywords

phoneme recognition, TIMIT, neural networks, temporal patterns, long temporal context, split temporal context, language identification

Bibliographic citation

Petr Schwarz: *Phoneme recognition based on long temporal context*, Doctoral thesis, Brno, Brno University of Technology, Faculty of Information Technology, 2008

Abstrakt

Tato práce se zabývá technikami pro automatické rozpoznávání fonémů z mluvené řeči. Cílem je získat co možná nejvíce informace o fonému z co největšího časového kontextu. Je použit hybridní systém založený na kombinaci skrytých Markovových modelů a umělých neuronových sítí. První přístup založený na časových trajektoriích (TRAPS) porovnán se systémy využívajícími konvenční techniky extrakce příznaků. TRAP systém je analyzován a zjednodušen. Následně je navržen nový systém s děleným časovým kontextem (STC), který dosahuje lepších výsledků a snižuje výpočetní náročnost. Tento systém byl ještě vylepšen obvyklými metodami, jako jsou třístavové modelování fonémů a fonotaktický jazykový model. Tento systém dosahuje 21.48 % chyby rozpoznávání fonému na databázi TIMIT. Tento systém byl také testován na dalších databázích, v šumu a na změnu přenosového kanálu. Nakonec je prezentováno několik aplikací, kde vyvinutý fonémový rozpoznávač našel uplatnění.

Klíčová slova

rozpoznávání fonémů, TIMIT, neuronové sítě, časové trajektorie, dlouhý časový kontext, dělený časový kontext, identifikace jazyků

Bibliografická citace

Petr Schwarz: *Phoneme recognition based on long temporal context*, Disertační práce, Brno, Vysoké Učení Technické v Brně, Fakulta informačních technologií, 2008

Prohlášení

Prohlašuji, že jsem tuto disertační prácí vypracoval samostatně pod vedením Doc. Dr. Ing. Jana Černockého. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal. Něteré v zavěru popsané aplikace fonémového rozpoznavače byly řešeny s dalšími členy skupiny Speech@FIT. Toto je vždy explicitně uvedeno.

Acknowledgments

I would like to thank my advisers Jan Černocký and Hynek Heřmanský for the guidance and for many valuable advices they gave me. I would like to thank my colleagues Lukáš Burget, Pavel Matěka, Martin Karafiát and Ondřej Glembek for many discussions and ideas we shared. I would like thank all the other members of the Speech@FIT group at Brno University of Technology, among all Michal Fapšo, František Grézl, Petr Motlíček and Igor Szöke, and also members of the Anthropic Speech Processing Group at OGI, namely Andre Adami, Pratibha Jain and Sunil Sivadas for the unforgettable time and experience. I would like to thank Tomáš Kašpárek, Petr Lampa, Pavel Chytil and others for always working computer infrastructure, and to our secretaries Sylva Otáhalová and Jana Slámová for the service they gave me. I would like to thank also my family for the patience they showed when I was writing this thesis.

Contents

List of Tables

List of Figures

List of Abbreviations

ANN	Artificial Neural Network
CRF	Conditional Random Fields
DCT	Discrete Cosine Transform
EM	Estimation Maximization
EER	Equal Error Rate
FOM	Figure of Merit
GMM	Gaussian Mixture Model
HATS	Hidden Activation Traps
HLDA	Heteroscedastic Linear Discriminant Analysis
HMM	Hidden Markov Model
KWS	Keyword spotting
LDA	Linear Discriminant Analysis
LID	Language Identification
LM	Language model
LVCSR	Large Vocabulary Conversational Speech Recognition
MFCC	Mel Frequency Cepstral Coefficients
ML	Maximum Likelihood
MMI	Maximum Mutual Information
MPE	Minimum Phoneme Error
MLP	Multilayer perceptron
PER	Phoneme Error Rate
PCA	Principal Component Analysis
PLP	Perceptual Linear Prediction Coefficients
PRLM	Phoneme Recognizer followed by Language Model
PPRLM	Parallel Phoneme Recognizer followed by Language Model
RNN	Recurrent Neural Network
SID	Speaker Identification
SSM	Stochastic Segmental Model
TMLP	Tonotopic Multi-Layered Perceptron
TRAPS	TempoRAl PatternS
VAD	Voice Activity Detection
WER	Word Error Rate

Chapter 1

Introduction

Phoneme recognition is very important part of automatic speech processing. Phoneme strings can transcribe words or sentences and the storage space is very small. It can be applied in many areas of speech processing – in large vocabulary continuous speech recognition, keyword spotting, language identification, speaker identification, topic detection, or in much easier tasks like voice activity detection. N-grams of phonemes are easily indexable, therefore phoneme recognition can be a basic part of systems for search in voice archives. In phonotatic language identification, topic detection or speaker identification, the language, topic or speaker can be represented by a phonotactic "language" model modelling dependencies among phonemes in phoneme strings. The accuracy of phoneme recognizer is crucial for the accuracies of all the mentioned technology! Therefore it is worth to investigate phoneme recognition and it is worth to develop as accurate phoneme recognizer as possible.

The thesis is focused on the main part of phoneme recognition, on acoustic modelling techniques. There are many other related issues, like channel normalization, channel and speaker adaptation, multilinguality, robustness in noise, but these issues are not investigated in detail

People recognize words from quite long temporal context. Sometimes we realize what was said even after few seconds, minutes or days. It depends on the quality and complexity of a model of the world we have in our heads. We are still far away form such model. This work investigates a basic model of phoneme and it tries to get as much as possible from the contextual information. Much longer temporal context than usual is used.

The main effort is given to a hybrid Artificial Neural Network / Hidden Markov Model approach.

1.1 Motivation

The main motivation for this work is the wide range of applications/tasks that the phoneme recognition affects. Improving phoneme recognition is not linked to just one particular problem but to wide ranges of problems. Phoneme recognition is not a closed box. It can be seen as an application of investigated acoustic modelling techniques. A better understanding of these techniques can allow us to better react to other needs in speech processing.

Another motivation was my study and then employment in Speech@FIT speech processing group at Brno University of technology and a stay at Oregon Graduate Institute. The groups were already investigating speech modelling techniques and features based on a long temporal context. But that time the techniques were used almost blindly. Deeper understanding helped to speed up the research and motivated research in another areas.

1.2 Original claims

In my opinion, the original contributions – “claims of this thesis” can be summarized as follows:

- Extensive comparison of phoneme recognition systems based on different structures of Artificial Neural Networks (ANN) and Gaussian Mixture Models (GMM).
- Detailed study of Temporal Pattern (TRAP) based system and its simplification.
- Definition of a split temporal contexts (STC) system reaching very good phoneme recognition results.
- Tuning of phoneme recognizers – applying and studying common speech recognition techniques that can decrease the phoneme error rate.
- Studying of phoneme recognizers on different databases, with varying amounts of training data, in noise and in cross-channel condition.
- Application of the long temporal context based phoneme recognizer to language identification, keyword spotting and voice activity detection.
- Discussion about techniques that can help to accurately train neural networks in speech recognition.

1.3 Scope of the thesis

Chapter 2 gives a basic introduction to the structure of phoneme recognizer. It shortly describes feature extraction and presents an imagination of acoustic speech units (phonemes, words) as trajectories in feature space. This imagination is very important as it guides significant portion of the presented work. It also summarizes common techniques for modelling of such trajectories. The TRAP approach and some of its derivations are described.

Chapter 3 is a literature overview of phoneme recognition techniques evaluated on the TIMIT database.

Chapter 4 describes an evaluation task, database partitioning, presents a comparison of a Gaussian Mixture Model based and Artificial Neural Network based system, gives a baseline results on the TRAP system, analyzes the TRAP system and introduces a simplification.

Chapter 5 introduces a system with a split left and right temporal contexts (LC-RC system). The system is studied and a nice reduction in phoneme error rate is reported.

Chapter 6 presents tuning of the LC-RC system by common techniques like phoneme states, language model and a larger training set. Also, some other architectures of neural networks are investigated.

Chapter 7 studies the developed phoneme recognition systems with different amounts of training data, on different databases, in noise and in cross-channel condition.

Chapter 8 demonstrates the usefulness of presented techniques on some application tasks: language identification, keyword spotting, LVCSR and voice activity detection.

Chapter 9 concludes the work.

Chapter 2

Introduction to speech recognition

2.1 Structure of speech recognizer

Classical speech recognizer can be seen in simplification as three main blocks – feature extraction, acoustic matching (classification) and a decoder, see Figure 2.1. The feature extraction block reduces bit rate of the input waveform signal by omitting irrelevant information and compressing relevant one. The acoustic matching block matches parts of the signal with some stored examples of speech units – words or phonemes, or with their models. The decoder finds the best path through the acoustic units (their order), optionally using an additional knowledge about the language.

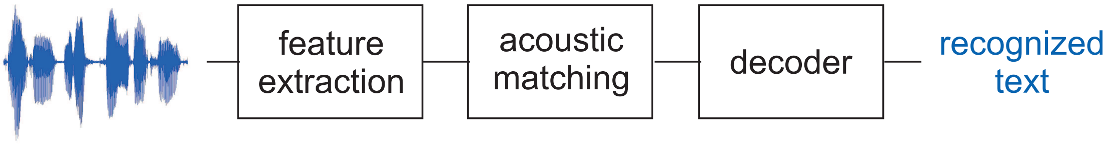

Figure 2.1: *Block diagram of common speech recognizer.*

2.2 Feature extraction – basic imagination

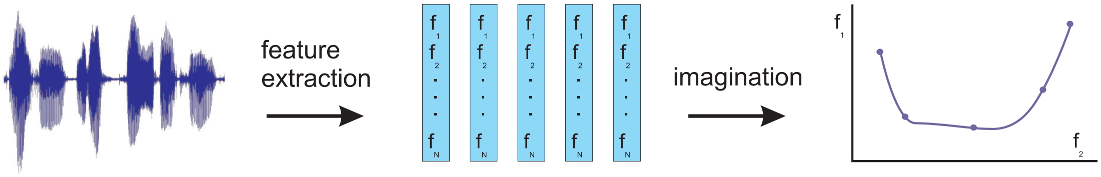

Figure 2.2: *How to see parametrization? Speech, feature vector, moving point in N-dimensional feature space.*

Speech signal is divided into overlapping frames, usually 25 ms length with 10 ms frame shift. Speech is supposed to be stationary in these frames. Then few parameters describing each frame are extracted. The aim is to reduce dimensionality of the speech frame, to adapt the speech frame to the classifier (for example decorrelation) and to suppress the influence of channel, within-class speaker variability etc. Nowadays, the most common feature extraction techniques are Mel Frequency Cepstral Coefficients (MFCC) [1] or Perceptual Linear Prediction (PLP) [2]. The MFCC processing is illustrated in the following section.

The vectors of parameters (feature vectors) can be seen as points in N-dimensional feature space, where N is the dimension of feature vectors (Figure 2.2). These feature vectors represent also (with the influence of the whole transmission chain: air, microphone, communication channel, etc) the state of our articulation organs. As the movements of our articulation organs are slow, points in the N-dimensional feature space representing neighboring feature vectors are also close in feature space. The distance between two neighboring points is not necessarily the same. If there are more similar frames (for example a vowel), the points are very close. On the opposite, points lying on transition between phonemes are farther apart.

A record of such points represents a trajectory. The trajectory is a result of speech generating process. We can imagine this generating process too: it is one point moving in N-dimensional feature space with varying speed. The speed is higher on transition between nonstationary parts of speech and lower in stationary parts of speech.

For us, in recognition, a sentence, a word or a phoneme is a part of the trajectory (a part of record of speech generating process). We need to model the part and we need to model as precisely as possible. The speed of the moving point is also important. The speed carries important information about phoneme duration, an information which is essential to distinguish between word in many languages.

2.2.1 Mel Frequency Cepstral Coefficients

This section shows in detail what is behind one point of trajectory representing an acoustic unit. The Mel Frequency Cepstral Coefficients (MFCC) are presented. MFCC [1] are widely used features nowadays and they are taken as baseline features in this work. The Logarithmic Mel Bank Energies (part of MFCC processing) are extracted and used by novel approaches presented later.

Individual steps are shown on the block diagram in Figure 2.3[1] Output of each step is shown in Figure 2.4 for a segment of voiced speech (vowel 'iy').

First, speech samples are divided into overlapping frames. The usual frame length is 25 ms and the frame rate is 10 ms. Example of one such frame for English vowel 'iy' can be seen in Figure 2.4a. Each frame is usually processed by preemphasis filter to amplify higher frequencies. This is an approximation of psychological findings about sensitivity of human hearing on different frequencies [3]. Hamming window is applied in the next step (Figure 2.4b) and Fourier spectrum is computed for the windowed frame signal (Figure 2.4c). Mel filter bank is then applied to smooth the spectrum: energies in the spectrum are integrated by a set of band limited triangular weighting functions. Their shape can be seen in Figure 2.4c (dotted lines). These weighting functions are equidistantly distributed over the Mel scale according to psychoacoustic findings, where better resolution in spectrum is preserved for lower frequencies than for higher frequencies. A vector of filter bank energies for one frame can be seen as a smoothed and down-sampled version of spectrum (Figure 2.4d). The logarithm of integrated spectral energies is taken with agreement to the human perception of sound loudness (Figure 2.4e). The feature vector is finally decorrelated and its dimensionality is reduced by its projection to several first cosine basis (Discrete Cosine Transform). The coefficients after DCT define the vector in N-dimensional (usually 13 dimensional) space.

[1] Thanks Lukas Burget for these illustrative figures.

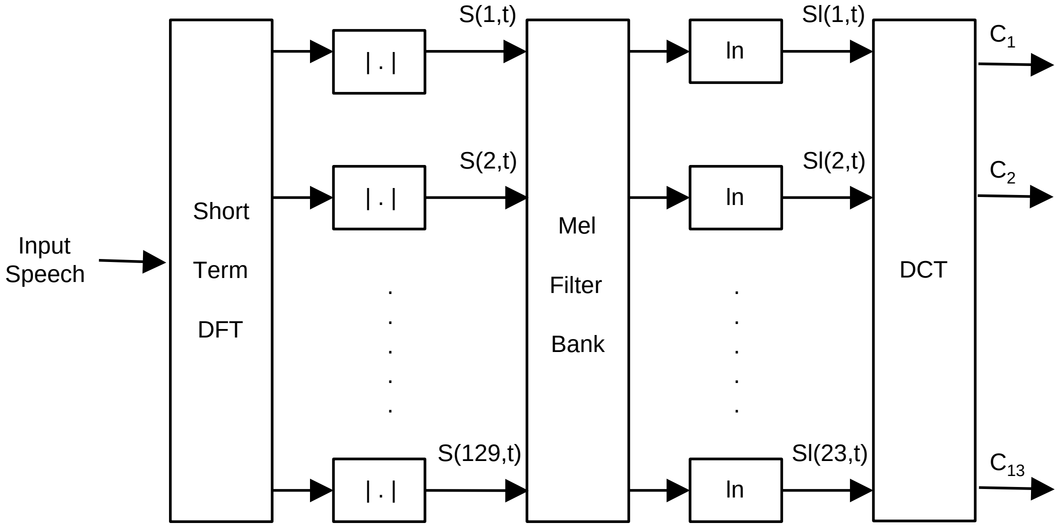

Figure 2.3: *Block diagram showing steps of MFCC computation.*

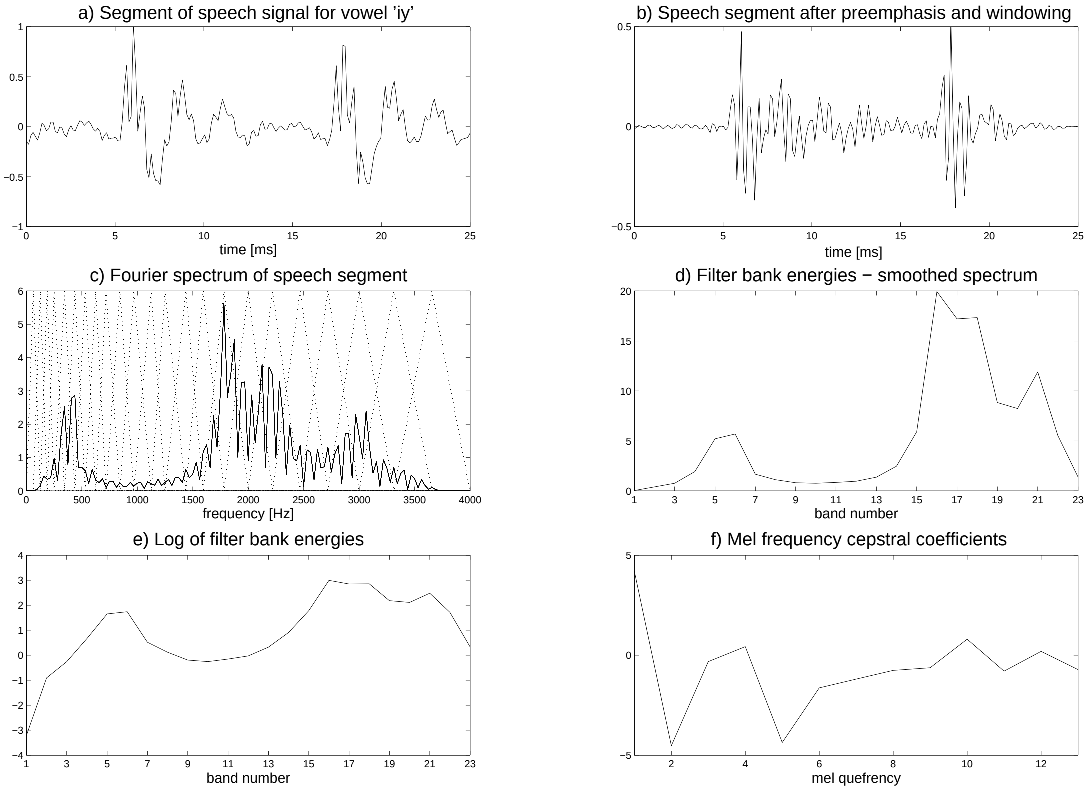

Figure 2.4: *Outputs of individual steps of MFCC computation.*

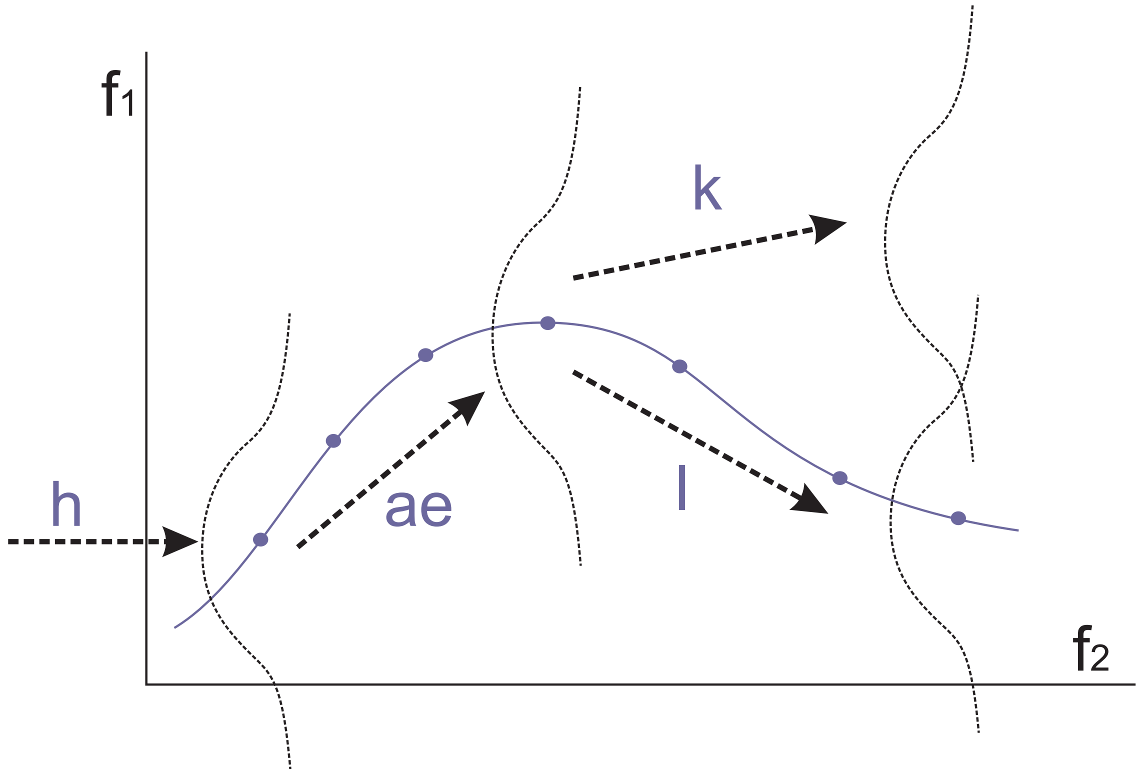

Figure 2.5: *Decoding process in speech recognition.*

2.3 Decoder

2.3.1 How does the decoding work?

Modelling techniques based on long temporal context are investigated and therefore a good knowledge about decoding process used in speech recognition is beneficial. The decoding process is illustrated in Figure 2.5. The aim of decoding is transcription of a chain of feature vectors (representing a part of trajectory) to a string of lexical units (words, phonemes, states). The decoder makes new hypotheses according a recognition network for each frame. For example "the frame belongs to phoneme *ae*, *k* or *l*" (Figure 2.5). The hypotheses are scored. The score is calculated as score of previous hypothesis plus score for the new frame. A tree of hypotheses is built. The source of information for scoring can be acoustic or linguistic. The acoustic knowledge can be the log likelihood of the frame given by the hypothetical unit. The Gaussian probability distribution is shown in the illustrative figure (drawn in one dimension only). The linguistic knowledge can be log of conditional probability of hypothesized linguistic unit given previous ones. After this scoring, new hypotheses can be done. The most likely branch (with the best score) is chosen at the end of utterance and its string is taken as the true one.

2.3.2 Shorter units in acoustic modelling

This section shows how some units shorter than phonemes units can improve acoustic modelling and also some remaining drawbacks of the approach are discussed.

What is the drawback of acoustic scoring and decoding process presented in previous section? There is just one probability distribution per phoneme, as can be seen in Figure 2.5. The

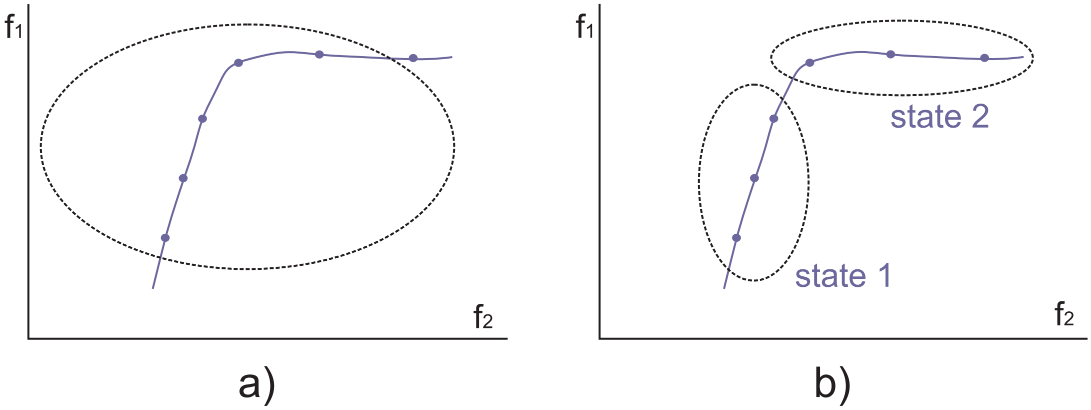

Figure 2.6: *Shorter lexical units in acoustic modelling: a) one probability distribution function per phoneme, b) two probability distribution functions per phoneme (two states).*

probability distribution (Gaussian) is wide because it must model the frames at the beginning and at the end of the phoneme. There are small distances among parts of space modelled by different phoneme models. The score (likelihood) is also worse for frames at the edges of phoneme. One common improvement are shorter acoustic units (states).

The effect of using shorter acoustic units is illustrated in Figure 2.6. Part a) shows one Gaussian probability distribution for modelling phoneme. The probability distribution is indicated by dashed ellipse showing places of one constant value of probability density function. Part b) shows what happens if the unit (phoneme) is split into two units (states) and modelled by two separate Gaussian probability distributions. These two Gaussians are narrower and the likelihood from both is greater now. There is more space for other trajectories representing other acoustic units in the feature space.

But still, even though shorter units allow for more precise modelling, it is not possible to create acoustic units with frame granularity because no unit is pronounced twice in the same way and once the unit can be shorter than second time. The fact that we have not one but many pronunciations of an acoustic unit means that we do not model one trajectory but a bunch of trajectories representing the same thing. Still, the variances of models are greater at the edges of phonemes than variance in the center of phoneme[2]. Trajectories can also cross, so sometimes it is not only important where the trajectory is placed, but its direction matters too. There can be also mistakes in phonetic transcriptions of words (incoherence between annotators or errors in grapheme-to-phoneme conversion) and the only possibility to consistently recognize a phoneme is looking at trajectory parts which belong to neighboring phonemes. Another difficulty can be an improperly chosen phoneme set. The trajectory parts of two phonemes can be indistinguishable. Modelling of a longer part of trajectory is again helpful.

2.4 Features looking at longer temporal context

The theoretical analysis [4][5] or variance analysis [6] of speech showed that significant information about phoneme is spread over few hundreds milliseconds. The phonemes are not completely separated in time but they overlap due to fluent transition of speech production organs from one configuration to another (co-articulation). This suggest that features or models that are able to

[2] This can be easily seen from variances of three state GMM models, see section 4.3.1

catch such long temporal span are needed in speech recognition. Another support for using of such long temporal spans are studies of modulation frequencies of band energies important for speech recognition [7]. The most important frequencies are between 2 and 16 Hz with maximum at 4 Hz. The 4 Hz frequency corresponds to time period of 250 ms, but to capture frequencies of 2 Hz, the an interval of half second is needed.

2.4.1 Deltas, double-deltas and triple deltas

As mentioned in section 2.3.2, some shorter lexical units can help in more precise acoustic modelling but do not solve some particular issues like crossing or trajectories or overlapping of trajectories representing different phonemes.

Delta coefficients

One common technique allowing to distinguish crossing trajectories are delta features. This technique adds an approximation of the first time derivatives of basic features (for example MFCCs) to the feature vector. The derivatives represent rough estimation of direction of trajectory in the feature space[3] and are estimated as:

$$\mathbf{d}_t = \frac{\sum_{i=1}^{N} i(\mathbf{c}_{t+i} - \mathbf{c}_{t-i})}{2\sum_{i=1}^{N} i^2} \tag{2.1}$$

where $\mathbf{d}_t$ is vector of delta coefficients for frame t computed from vectors of static coefficients $\mathbf{c}_{t+i}$ to $\mathbf{c}_{t-i}$. The usual window length is 5 frames, therefore delta features use 65 ms long temporal context (4×10 ms + 1 × 25 ms).

Double-delta coefficients

Equation 2.1 can be applied to delta features and the derivatives of delta features can be attached too. These new derivatives are called delta-delta features or acceleration coefficients. Delta-delta features introduce even longer temporal context. If the window has also 5 frames, the temporal context is 9 frames which is 105 ms (8×10 ms + 1×25 ms). Delta-delta features can say whether there is a peak or a valley on the investigated part of trajectory.

Triple-delta coefficients and reduction of dimensionality

Even a benefit of using triple-delta was seen on some larger databases. These features are attached to a vector of static features and deltas and double-deltas too but the vector is not fed into a classifier directly. Its dimensionality is reduced by linear transform usually estimated by Linear Discriminant Analysis (LDA) or Heteroscedastic Linear Discriminant Analysis (HLDA) on the train data. The purpose of this transform is an adaptation of features to the model, obviously. If there are 13 MFCCs, the full extended vector before dimensionality reduction has 52 values and usually 39 values is kept after reduction. The linear transforms are shortly described in section 2.4.4. A block schema of such feature extraction module is shown in Figure 2.7. The temporal context is 145 ms long. This feature extraction approach is used for example in the AMI[4] LVCSR system [8].

[3] Time derivatives are taken, not spatial.

[4] AMI (Augmented Multi-party Interaction) is an European project with the aim of developing technologies that help people to have more productive meetings – `www.amiproject.org`

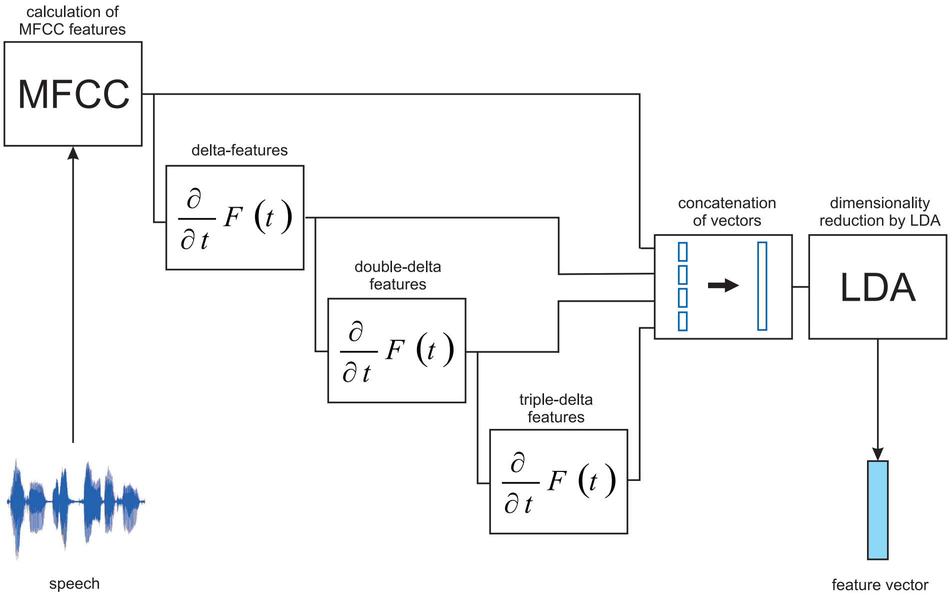
calculation of MFCC features
MFCC
delta-features
$\frac{\partial}{\partial t} F(t)$
double-delta features
$\frac{\partial}{\partial t} F(t)$
triple-delta features
$\frac{\partial}{\partial t} F(t)$
concatenation of vectors
dimensionality reduction by LDA
LDA
speech
feature vector

Figure 2.7: *System with triple-deltas followed by dimensionality reduction.*

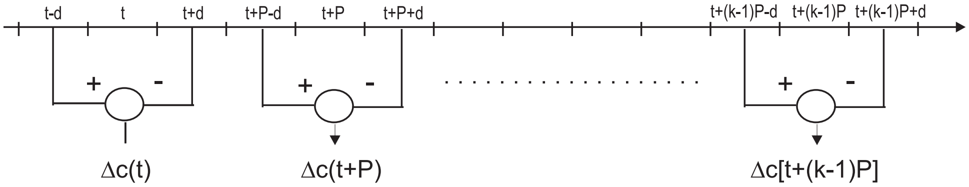

Figure 2.8: *Shifted delta cepstra.*

2.4.2 Shifted delta cepstra

The Shifted delta cepstra (SDC) are features widely used in acoustic language identification [9]. These features do not look at trajectory in one place of feature space but they look at trajectory from more surrounding places by shifting deltas. This allows to catch even word fractions by the features directly.

Computation of SDC is illustrated in Figure 2.8. SDC are characterized by set of four parameters, N, d, P, k, where N is the number of cepstral (MFCC) coefficients computed at each frame, d represents time advance and delay for delta computation, k is the number of blocks whose delta coefficients are concatenated to form the final feature vector, and P is the time shift between consequent blocks. kN parameters are used for each feature vector.

2.4.3 Blocks of features

Very often, a block of consequent MFCC or PLP features, or a block of consequent MFCC or PLP features enhanced with delta and double delta is used as features. This block can be used directly in a classifier, for example in neural networks, or its dimensionality can be reduced and the features decorrelated by a linear transform for GMM. This approach was used for example by IBM in their LVCSR system [10].

The TRAP (Temporal Patterns) feature extration described in 2.6.1 and investigated later in this thesis falls in this category too. Here, the features are critical-band energies and the blocks contain long evolutions in one critical band.

2.4.4 Linear transforms learned on data

Linear transforms are often used for feature decorrelation[5] and dimensionality reduction. Three estimation techniques for linear transforms are discussed.

Principal Component Analysis (PCA)

This technique allows to find dimensions with the highest variability in feature space. The projection itself is then a projection to these dimensions. Eigenvalues obtained from PCA indicate variability in each dimension. This allows to keep as many bases[6] as necessary to keep certain variability in input features. The variability means how precisely the input features can be reconstructed. When the PCA is used, it is necessary to have in mind that an information useful for later classification does not necessarily have the highest variability and the information can be lost during projection and dimensionality reduction. More about this transform can be found in [11][12][13].

Linear Discriminant Analysis (LDA)

In addition to considering the properties of input during estimation of the transform this technique also takes into account distributions of *classes* (states, phonemes ...). LDA allows to derive linear transform whose bases are sorted by their importance for discrimination among classes. It maximizes a ratio of across-class and whitin-class variance. However, assumption that features belonging to each particular class obey Gaussian distribution and that all the classes share the same covariance matrix is quite limiting the optimal functionality of LDA. More about this transform can be found in [11][12][13].

Heteroscedastic linear discriminant analysis (HLDA)

This technique was first proposed by N. Kumar [14][15]. It can be viewed as a generalization of LDA. HLDA again assumes that classes obey multivariate Gaussian distribution, however, the assumption of the same covariance matrix shared by all classes is relaxed. More about this transform can be found in [13].

2.5 Acoustic matching

The acoustic matching block assigns scores to acoustic units hypothesized by the decoder. The Hidden Markov Models (HMMs) [16][17] are commonly used for this purpose. The HMMs introduce an assumption of statistical independence of frames. This implies that the final score (likelihood) of an acoustic unit is given by product (or sum of log-likelihoods) coming from frames. The per frame likelihood is modelled by a probability density function, usually by Gaussian Mixture Model (GMM), or it can be estimated by an Artificial Neural Networks (ANN). In this case we speak about HMM/ANN hybrid [18]. Both approaches have their advantages and disadvantages.

[5] If we say that features are decorrelated, it means that we are not able to estimate value of one feature from another. This property is beneficial for acoustic modelling and allows to simplify modelling technique. For example, a diagonal covariance matrix can be used in Gaussian Mixture Models instead of a full covariance matrix

[6] The base component is a row of the projection matrix. The number of kept bases gives the dimensionality of target feature vector.

2.5.1 Gaussian Mixture Models

The Gaussian Mixture Models model probability distribution of feature vectors. The explicit modelling of data allows for a simple training based on well mathematically based analytics formulas. There is Maximum Likelihood (ML) estimation criterion, but there are also discriminative Maximum Mutual Information (MMI) or Minimum Phoneme Error (MPE) criteria. The formulas use accumulation of statistics that allows to easily parallelize the training. Also, an adaptation is easy. On the opposite, the explicit modelling of probability distributions of data needs more parameters in the model. The recognition phase is therefore slower in comparison to ANNs. The GMMs need to estimate covariance matrices during the estimation. The number of parameters in the covariance matrix (and therefore the amount of training data) grows quadratically with the feature vector dimension. The common approach is using of diagonal covariance matrices. The required amount of training data is smaller then, the model is simpler and faster for evaluation, but then the input features must be decorrelated.

2.5.2 Artificial Neural Networks

The artificial neural network is a discriminatively trained classifier that separates classes by hyperplanes. Therefore, parameters are not wasted for some places in feature space where they can not affect to the classification. This makes the classifier small and simple. It can run very fast and therefore it can be easily ported to low-end devices. The artificial neural networks can process highly dimensional feature vectors more easily than GMM. They also process correlated features.

Multilayer Perceptron (MLP)

One of the simplest neural network structures is multilayer perceptron, which was widely accepted for speech recognition [18]. It is a three layer neural network – the first layer copies inputs, the second (hidden) has the sigmoidal nonlinearities, and the final (third) layer in HMM/ANN uses the SoftMax nonlinearity. This final nonlinearity ensures that all output values sum to one so that they can be considered probabilities. The network is trained to optimize the cross-entropy criteria. Such networks were adopted for this thesis.

Recurrent Neural Network (RNN)

Another kind of ANNs are recurrent neural networks [19]. The recurrent neural network has just two layers – input layer, which copies input feature vector, and an output layer. The output layer does not have only neurons that represent outputs, but it has also some neurons that represent hidden states. These states are sent with a time shift of one frame back to the input. This allows to model theoretically infinitely long time context (to the past) and reach better results than MLPs. Although the RNN can model only one context, some works model both contexts independently and merge the outputs [20]. Many techniques studied in this thesis are used by RNNs implicitly.

2.6 TRAPs and hierarchical structures of neural networks

The multilayer perceptron is one possibility for acoustic matching. But people (for example [21] and [22]) found that more complicated neural network structures can be beneficial for speech recognition. This section presents Temporal Patterns (TRAPs) as a hierarchical structure of MLPs and some approaches derived from TRAPs.

2.6.1 TRAPs

The TRAP system is shown in Figure 2.9. Critical bands energies are obtained in conventional way. Speech signal is divided into 25 ms long frames with 10 ms shift. The Mel filter-bank is emulated by triangular weighting of FFT-derived short-term spectrum to obtain short-term critical-band logarithmic spectral densities. TRAP feature vector describes a segment of temporal evolution of such critical band spectral densities within a single critical band. The usual size of TRAP feature vector is 101 points [21]. The central point is the actual frame and there are 50 frames in past and 50 in future. That results in 1 second long time context. The mean and variance normalization can be applied to such temporal vector. Finally, the vector is weighted by Hamming window. This vector forms an input to a classifier. The outputs of the classifier are posterior probabilities of sub-word (phonemes or states) classes which we want to distinguish. Such classifier is applied in each critical band. The merger is another classifier and its function is to combine band classifier outputs into one. The described techniques yield phoneme probabilities for the center frame. Both band classifiers and merger are neural nets.

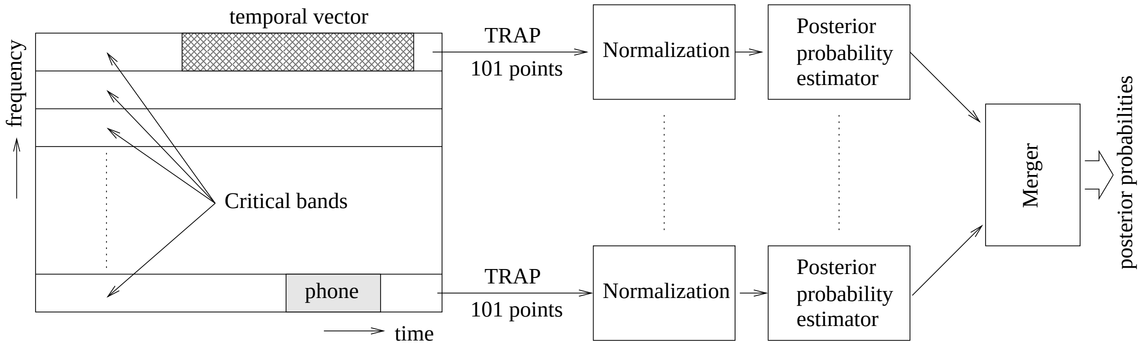

Figure 2.9: *TRAP system.*

2.6.2 3 band TRAPS

Pratibha Jain [23] showed that information coming from one critical band is not enough for band classifier and extended the input temporal vectors for band classifier with vectors from neighboring bands (one from each side). The error rate of TRAP system was significantly reduced.

2.6.3 Sobel filters

The Sobel operators are known in computer graphic. František Grézl [24] used these operators to extract additional information from features in the TRAP system. The time-frequency block of features is preprocessed by a 2D filter before classical TRAP processing. The 2D filter can be designed to emphasize for example differentiation in time or frequency domain, similarly as edge detector in computer graphic. Another filter can perform averaging. Some examples of impulse responses of such filters are shown in Tables 2.1 and 2.2.

Many TRAP systems can be developed using different filters. No improvement was obtained from Sobel operators working with information from one domain only (time or frequency). But

freq. average

0	1	0
0	2	0
0	1	0

freq. difference

0	-1	0
0	0	0
0	1	0

time average

0	0	0
1	2	1
0	0	0

time difference

0	0	0
-1	0	1
0	0	0

Table 2.1: *Sobel operators working in one dimension (time or frequency)*

G1

-1	0	1
-2	0	2
-1	0	1

G2

1	2	1
0	0	0
-1	-2	-1

G3

0	1	2
-1	0	1
-2	-1	0

G4

-2	-1	0
-1	0	1
0	1	2

Table 2.2: *Two dimensional Sobel operators*

operators working with information from both domains gave the best results.

2.6.4 Hidden Activation TRAPs (HATS)

The outputs from band classifiers in the TRAP system are subword units (phoneme/state posteriors). But such representation in the middle of the structure is questionable. B. Chen and Q. Zhu [25][22] supposed that the mapping to posteriors by the band classifiers is useless and all the valuable information for merger was already extracted by the first-layer. Therefore the final layer were removed for all the band classifiers after this networks were trained. The authors showed 8.6 % relative reduction of WER in a tandem[7] based LVCSR system.

2.6.5 Tonotopic Multi-Layered Perceptron (TMLP)

The Tonotopic Multi-layered Perceptron [22] has exactly the same structure as Hidden Activation TRAPs. The difference is in the training. In case of TMLP, a large (composite) neural network is built and this network is trained while optimizing one criterial function. It is a de facto four layer network with some constrains applied for neurons in the second layer (first layer of neurons). The author showed an improvement against conventional TRAP system but worse results than HATs.

[7] The neural network posteriors are used as features in conventional LVCSR system, see section 8.2

Chapter 3

Phoneme recognition on TIMIT

This chapter presents the TIMIT database and works investigating phoneme recognition on it. The list is definitely not exhaustive. The described works present whole phoneme recognition systems and use similar scoring procedure as the one introduced by L. Lee [26]. Many other works study phoneme classification (where the phoneme boundaries are known), deal with recognition of just some phoneme classes or use different scoring procedure.

3.1 Databases

The TIMIT database was chosen for my experiments. The database is small, therefore the experiments are fast. Also, many published results for phoneme recognition on this database already exist. Big advantage of the database is its hand-made phoneme-level transcription. This allows to evaluate the phoneme recognition error rate more precisely than for standard speech databases where the phonetic transcription needs to be generated (by forced alignment) and is itself prone to errors.

The NTIMIT database was used for some cross-channel experiments. This database was created by passing TIMIT through telephone channel. Thus this database represents 8 kHz narrow-band telephone speech.

Another databases are presented in chapter 8, where techniques investigated in this thesis, are used for some application tasks.

3.1.1 TIMIT

Design of the TIMIT [27] corpus was a joint effort among the Massachusetts Institute of Technology (MIT), SRI International (SRI) and Texas Instruments, Inc. (TI). The speech was recorded at TI, transcribed at MIT. The records are sampled at 16000 Hz. Each record was transcribed at word level at first, forced aligned to phonemes by a speech recognizer and hand checked.

Data sets

For my experiments, this database was divided into three sets: training set (512 speakers), cross-validation set (50 speakers) and test set (168 speakers). The cross-validation set is used for tuning of constants and for stopping algorithm that prevents overtraining of neural networks. All *SA* records (identical sentences for all speakers in database) were removed as they could bias the results.

Phoneme set

The phoneme set adopted for this thesis uses 39 phonemes. It is very similar to the CMU/MIT phoneme set [26], but closures were merged with burst instead of with silence (bcl b → b). It is more appropriate for features which use a longer temporal context such as presented here. Mapping between the TIMIT original phoneme set, the CMU/MIT 39 phonemes set used by many researches, and our (BUT) set is in table 3.1.

TIMIT	CMU/MIT	BUT	TIMIT	CMU/MIT	BUT
p	p	p	b	b	b
t	t	t	d	d	d
k	k	k	g	g	g
pcl	sil	p	bcl	sil	b
tcl	sil	t	dcl	sil	d
kcl	sil	k	gcl	sil	g
dx	dx	dx	q	-	-
m	m	m	em	m	m
n	n	n	en	n	n
ng	ng	ng	eng	ng	ng
nx	n	n			
s	s	s	sh	sh	sh
z	z	z	zh	sh	sh
ch	ch	ch	jh	jh	jh
th	th	th	dh	dh	dh
f	f	f	v	v	v
l	l	l	el	l	l
r	r	r	w	w	w
y	y	y	h#	sil	pau
pau	sil	pau	epi	sil	pau
hh	hh	hh	hv	hh	hh
eh	eh	eh	ih	ih	ih
ao	aa	aa	ae	ae	ae
aa	aa	aa	ah	ah	ah
uw	uw	uw	uh	uh	uh
er	er	er	ux	uw	uw
ay	ay	ay	oy	oy	oy
ey	ey	ey	iy	iy	iy
aw	aw	aw	ow	ow	ow
ax	ah	ah	axr	er	er
ix	ih	ih	ax-h	ah	ah
# phonemes			61	39	39

Table 3.1: *Phoneme mapping for TIMIT.*

3.1.2 NTIMIT

The NTIMIT was created by passing TIMIT through a telephone channel. The transmitted speech records were synchronized with the original, therefore the data sets and the phoneme sets are exactly the same.

3.2 Evaluation metric

Phoneme Error Rate (PER) is used for comparison of different phoneme recognition systems. The recognized phoneme string is aligned to the reference phoneme string using dynamic programing. Then the number of substitution errors S – phoneme is recognized as another phoneme, deletion errors D – phoneme is not recognized, and insertion errors I – phoneme is incorrectly included – are counted. PER is calculated using equation:

$$PER = \frac{S+I+D}{N} \times 100\ \% = \left(1 - \frac{H-I}{N}\right) \times 100\ \%, \tag{3.1}$$

where N is number of phonemes in the reference string and H is number of correctly recognized phonemes. PER is calculated exactly in the same way as widely known word error rate (WER), but on the phoneme strings. In my work, the HTK[1] HResults command was used for evaluation.

3.3 Summary of published works

3.3.1 K. Lee and H. Hon – Diphone Discrete HMMs

The K. Lee's and H. Hon's work [26] is one of the first works using the TIMIT database for phoneme recognition. It introduced mapping to 39 phonemes set for evaluation of phoneme recognition accuracy. The work uses LPC-derived cepstral coefficients, energy, delta and double-delta features. The phonemes are modelled by right-context dependent discrete HMMs. 3 codebooks of size 256 are used – for basic features, for deltas and for double-deltas. The system uses bigram language models. The article also compares different setups with and without language models, and the context-dependent system to a context-independent system. The best phoneme error rate is 33.92 %.

3.3.2 S. J. Young – Triphone Continuous HMMs

S. J. Young [28] took the TIMIT phoneme recognition task as an evaluation task for different approaches of HMM state tying. The input features are MFCC, log energy and deltas. The presented phoneme error rate is 38.3 %. This is higher than the previous result but this system uses just delta features, not double-deltas. The author also noticed that some monophone results from system with more Gaussians are comparable with the triphone results[2].

3.3.3 V. V. Digalakis and M. Ostendorf and J. R. Rohlicek – Stochastic Segmental Models

V. V. Digalakis and colleagues [29] investigate fast search method in monophone Stochastic Segmental Models. The segmental models bypass drawback of HMMs assumption of statistical independence of frames. The phoneme error rate is 36 %.

[1] `http://htk.eng.cam.ac.uk`

[2] A similar observation was done during my experiments using triphone Continuous Density HMMs

3.3.4 D. J. Pepper and M.A. Clements – Ergodic Discrete HMM

D.J. Pepper and M.A. Clements [30] tried to cover the whole acoustic space by one big Ergodic Discrete HMM. Then they trained another discrete HMM or a finite state automation to convert state labels to phoneme strings. The phoneme error rate (PER) obtained using discrete HMM is 58.5 % and the PER obtained using finite state automata is 64.6 %.

3.3.5 L. Lamel and J. Gauvian – Triphone Continuous HMMs

L. Lamel and J. Gauvian [31] built a phoneme recognition system using tied state Continuous Density HMMs. It uses MFCC, log energy, deltas and double-deltas as features. The models are 3-state gender-dependent with tied initial and final states. The duration modelling using Gamma distribution is applied. The system uses trigram language model. A result with the bigram language model and some results on another databases are also presented. The best phoneme error rate is 26.6 %.

3.3.6 S. Kapadia, V. Valtchev and S. J. Young – Monophone HMMs and MMI training

S. Kapadia and colleagues [32] were investigating discriminative training criterion for HMM parameter estimation. They used Maximal Mutual Information (MMI) training instead of Maximum Likelihood (ML) training. They are also compared diagonal and full covariance matrices for Gaussian Mixture Modelling. The diagonal covariance matrix system with 16 Gaussians reached 33.3 % PER using ML criterion and 32.5 % PER using MMI criterion. A full matrix system with 4 Gaussians reached 32.6 % using ML criterion and 30.7 % PER using MMI criterion.

3.3.7 T. Robinson – Recurrent Neural Networks

T. Robinson presented inspirative work on recurrent neural networks (RNN) [19]. The RNN are discriminative and they can model theoretically infinitely long left context. The output nonlinearity is SoftMax and the objective function is cross-entropy. The RNN models its internal states implicitly and produces vectors of phoneme posteriors. A HMM/ANN hybrid system is used. The best phoneme error rate is 25.0 %.

3.3.8 A. K. Halberstadt – Heterogeneous Acoustic Measurements, Segmental Approach

A. K. Halberstadt in his thesis [33] investigated heterogeneous acoustic features for phoneme recognition. He defined a hierarchical approach for phoneme recognition. Different phonemes use different features – MFCCs, PLPs, different window lengths, different time-based features (deltas, averages, DCTs). A segmental decoder is used. All classifiers use Gaussian Mixture Models. The work also investigates merging of classification outputs – voting, weighted linear combination of log-likelihoods and a Gaussian backend. The best presented phoneme error rate is 24.4 %. The thesis gives also a good overview of priors works done on TIMIT on classification of phonemes or recognition of particular phoneme classes.

3.3.9 J. W. Chang – Near-Miss modelling, Segmental Approach

J. W. Chang in his thesis [34] investigated decoding in the segmental speech recognition. The segment-based representation is a temporal graph, where each vector corresponds to a hypothesized phoneme, similarly as phoneme lattice in classical frame based framework. The previous work introduced anti-phoneme modelling of off-best-path segments. The idea was that the off-best-path segment is not a phoneme and can be modelled as an anti-phone. A new approach that generalizes anti-phoneme modelling to more complex modelling of off-best-path segments is introduced. It models a near-miss subset of segments. The phoneme error rate is 25.5 %.

3.3.10 B. Chen, S. Chang and S. Sivadas – MLP, TRAPs, HATs, TMLP

B. Chen and his colleagues [35] were working on a HMM/ANN hybrid. They introduced new structures of neural networks – Hidden Activation TRAPS (HATS) and Tonotopic Multi-Layer Perceptrons (TMLP). These structures (see 2.6.4 and 2.6.5) have similar properties as TRAPs but 84 % less trainable parameters. The authors compare TRAPs (32.7 %), HATS (29.8 %), TMLP (31.1 %) and simple MLP with the PLP features (29.7 %). The simple MLP reached the best results for clean speech. The TRAPs, HATS or TMLP give better results for noisy speech in some cases. The best phoneme error rate (26.5 %) was obtained with frame based combination (multiplication) of phoneme posterior vectors from MLP(PLP) and the HATS system.

3.3.11 J. Moris and E. Fosler-Lussier – TANDEM and Conditional Random Fields

J. Moris and E. Fosler-Lussier [36] used MLPs to extract speech articulation attributes from speech. The input for MLPs are PLPs and delta features. The MLPs are trained on labels obtained from transcription of phoneme labels to articulation attribute labels. Then the attributes are modelled by conventional HMM or by discriminative Conditional Random Fields (CRF). The best presented phoneme error rate 33.31 % is coming from a triphone TANDEM architecture (MLP followed by triphone HMM/GMM). Monophone CRF system gives 34.77 % PER which is a better result than monophone TANDEM system with 38.52 % PER. The work also present results obtained by classical triphone HMM system trained for MFCCs (37.63 %) or PLP (39.92 %).

3.3.12 F. Sha and L. Saul – Large Margin Gaussian Mixture Models

F. Sha and L. Saul [37] found an inspiration in Support Vector Machines and trained their GMMs discriminatively by maximizing margin among classes. Their models are monophone and the input features are conventional MFCC, deltas and double deltas. They got 30.1 % phoneme error rate with 16 Gaussians in comparison to 31.7 % PER when the conventional ML criterion was applied.

3.3.13 L. Deng and D. Yu – Monophone Hidden Trajectory Models

L. Deng and D. Yu [38] built complex model of co-articulated time-varying patterns of speech. The model incorporates two important stages – step from phoneme sequence to vocal tract resonance dynamic (VTR), and then from VTR to cepstrum-based observation vectors. The VTR is modelled by a finite impulse response filter. The cited article extends the Hidden Trajectory Model model with ability to model differential cepstra. This approach gives 24.8 % phoneme error rate.

3.3.14 Comparison and discussion

All the presented works are summarized in Table 3.2 together of publication. The range of works is really wide. A comparison is very difficult because the works use different features, sometimes slightly modified scoring procedure (recognition of 61 or 48 or 39 phonemes followed by mapping to 39) and the year they were published differs. Some earlier systems would definitely reach better results today just because of the hardware. Earlier, the authors used lower number of Gaussians and a special hardware used for neural networks limited the task. Basically, three strong ways to improve phoneme error rate are visible – more different (complementary) features, better (more precise) model and a discriminative training criteria. It is also obvious that a careful engineering work can improve the results.

Among the works, I will emphasize two:

- Recurrent Neural Networks. It is a very simple classifier and without any other tricks, the results are among the best.
- The Ergodic HMM raises a question: "What is phoneme recognition? Why not use the whole LVCSR for phoneme recognition?".

Although the phoneme error rates are already low, still a huge space for improvement is open. All the systems are speaker independent. An adaptation would improve the results. Then there are other techniques commonly used in LVCSR (speaker adaptive training, consensus decoding, posterior features, minimum phoneme error training) that can be applied.

First author	Year	Technique	PER (%)
K. Lee	1989	Diphone Discrete HMMs	33.9
S. J. Young	1992	Triphone Continuous HMMs	38.3
V. V. Digalakis	1992	Stochastic Segmental Model	36.0
D. J. Pepper	1992	Ergodic DHMM	58.5
S. Kapadia	1993	Monophone HMMs, MMI training	30.7
L. F. Lamel	1997	Triphone Continuous HMMs	26.6
T. Robinson	1994	Recurrent Neural Nets	25.0
A. K. Halberstadt	1998	Heterogeneous Measurements	24.4
B. Chen	2003	HATS + MLP(PLP)	26.5
J. Moris	2006	Triphone TANDEM	33.3
J. Moris	2006	Monophone CRF	34.8
F. Sha	2006	Large Margin GMM	30.1
L. Deng	2007	Hidden Trajectory Models	24.8

Table 3.2: *Comparison of different phoneme recognition techniques presented in literature on the TIMIT database*

Chapter 4

Baseline systems

This chapter concentrates on the basic phoneme recognition experiments with HMM/GMM and HMM/ANN systems and novel TRAP based techniques. In order to be comparable to state-of-the-art, all results are reported on TIMIT.

4.1 What system as baseline?

The Temporal Pattern (TRAP) system was taken as baseline. This system was known to give better results than conventional techniques (HMM/GMM with MFCC) in some cases [39] (mainly in cross-channel conditions), and the ANN features (posterior probabilities of phonemes) were known to be complementary features for MFCCs or PLPs [40]. But there was no detailed understanding of the whole approach, therefore the TRAP system is studied. The TRAP system is compared to some conventional systems based on MFCCs. There is a big step between the TRAP system based on HMM/ANN hybrid and a HMM/GMM based on MFCCs, therefore the HMM/ANN and HMM/GMM are compared on MFCC features at first and then the TRAP system is compared to a HMM/ANN hybrid based on MFCCs.

4.1.1 HMM/GMM

All the GMM experiments are done with the HTK toolkit[1]. The features are $MFCC + C_0 + \Delta + \Delta\Delta$ (together 39 coefficients). Detailed parametrization setting can be seen in Table 4.1. This feature set is referred as MFCC39. The HMM models were initialized to global means and variances. Then the models were re-estimated, all the Gaussians split to two and re-estimated again. This was repeated up to 256 Gaussians. The recognition was done using the HVite decoder.

4.1.2 HMM/ANN

The HMM/ANN hybrid is based on the SVite decoder and the QuickNet ANN software[2]. The SVite decoder is a part of BUT STK toolkit[3]. The input features are $MFCC + C_0 + \Delta + \Delta\Delta$ or other features derived from Mel-bank energies in later experiments. The detailed parametrization setting is in Table 4.1 (the same as for HMM/GMM). Neural networks are trained to map

[1] `http://htk.eng.cam.ac.uk`

[2] `http://www.icsi.berkeley.edu/Speech/qn.html`

[3] `http://speech.fit.vutbr.cz/en/software/hmm-toolkit-stk-speech-fit`

sampling frequency	16000 Hz (8000 Hz)
window length	25 ms
shift	10 ms
window	Hamming
pre-emphasis	no
waveform frame mean norm.	yes
# mel banks	23 for 16000 Hz
	15 for 8000 Hz
# cepstral coefs.	13 (including C_0)

Table 4.1: *Setting for Mel-bank energies or MFCC extraction.*

input features to phoneme posteriors according to hard labels (each feature vector is assigned to one phoneme).

4.1.3 HMM/GMM and HMM/ANN based on MFCCs with one state model

This experiment compares HMM/GMM system and HMM/ANN hybrid. The input features are MFCC39. The numbers of parameters in GMM or ANN were found such way that the decrease in phoneme error rate caused by adding new parameters is negligible. This procedure was used also in all following experiments. The final number of Gaussian mixtures is 256 and final number of neurons in the hidden layer is 500. The results are in Table 4.2. There is almost no difference in PER (0.3 %), so if the features are well adapted to the model and the training procedure is optimal, it should be possible to reach similar results with both HMM/GMM and HMM/ANN systems. Table 4.3 shows the number of parameters in both systems. The HMM/ANN system 5 % parameters compared to the HMM/GMM system.

system	ins	sub	del	PER
GMM	4.1	18.7	15.2	38.0
AMM	4.7	20.6	12.4	37.7

Table 4.2: *Comparison of HMM/GMM and HMM/ANN based on MFCCs with one-state model.*

system	# parameters (floating point numbers)
GMM	788736
NN	39539

Table 4.3: *Comparison of numbers of parameters in HMM/GMM and HMM/ANN systems based on MFCCs.*

4.2 Basic TRAP system

The TRAP system is shown in detail in Figure 4.1. Speech is segmented into frames 25 ms long and for each frame, mel-bank energies are calculated. Temporal evolution of energy for each band is taken (101 values = 1 second), normalized to zero mean and unit variance across the temporal vector, windowed by Hamming window and then normalized to zero means and unit variances across all training vectors. This is beneficial for the ANN as it is ensured that all inputs have the same dynamics. For testing, the later normalization coefficients are not calculated but taken from the training set. Such prepared temporal vectors are presented to band neural networks. These neural networks are trained to map temporal vectors to phonemes. A vector of phoneme posterior probabilities is obtained at the output of each band neural network. The posterior probabilities from all bands are concatenated together, the logarithm is taken and this vector is presented to another neural network (merger). The merger is trained to map the vectors to phonemes again. The output is a vector of phoneme posterior probabilities. Such vectors are then sent to the Viterbi decoder to generate phoneme strings.

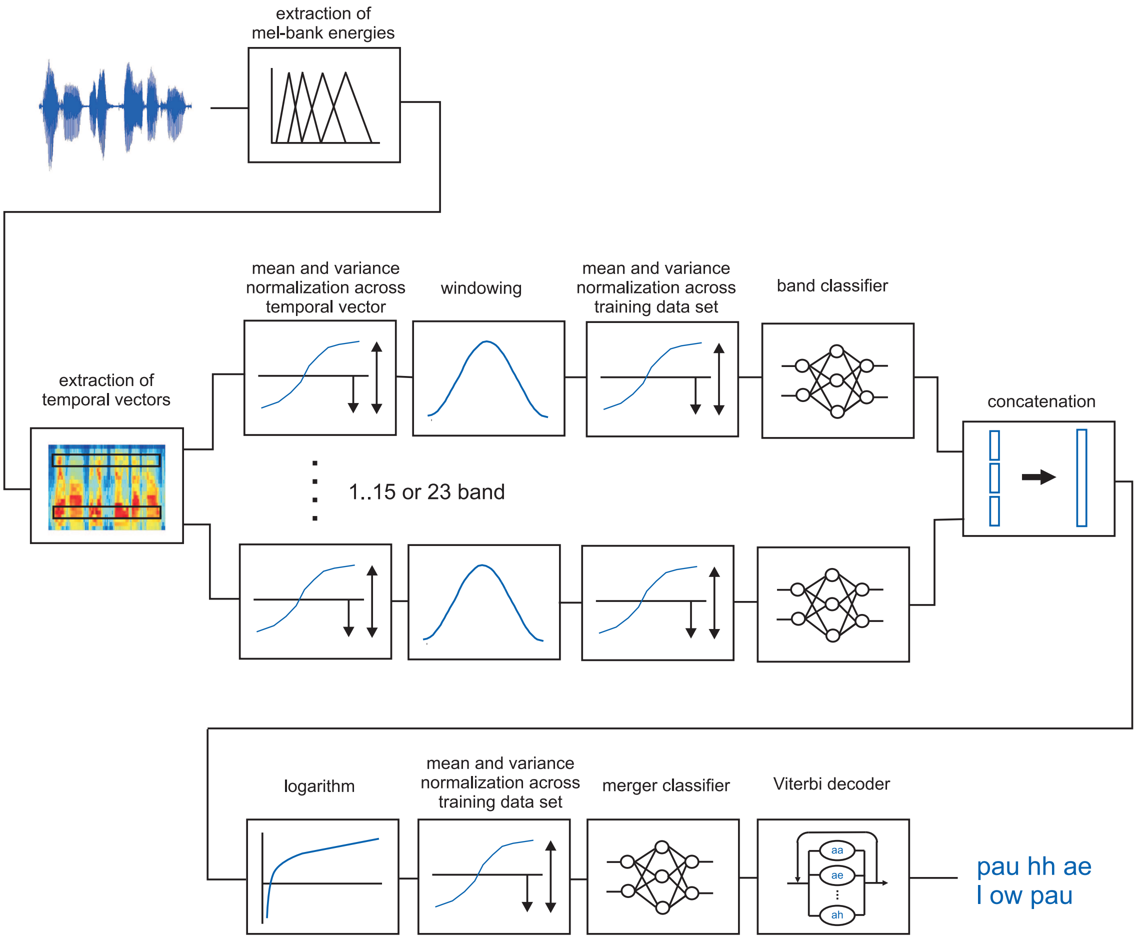
extraction of mel-bank energies
extraction of temporal vectors
mean and variance normalization across temporal vector
windowing
mean and variance normalization across training data set
band classifier
concatenation
1..15 or 23 band
logarithm
mean and variance normalization across training data set
merger classifier
Viterbi decoder
aa
ae
ah
pau hh ae
l ow pau

Figure 4.1: *Block diagram of the TRAP system.*

4.2.1 Effect of mean and variance normalization of temporal vector

The mean and variance normalization of temporal vector makes the TRAP system more robust against channel change. The normalizations works similarly as cepstral mean and variance normalization[4], commonly applied in MFCC.

The mean normalization can be seen also as a temporal filtering, similar to RASTA [41]. A change in the length of temporal vector affects characteristics of the filter. There is no visible

[4] Cepstral coefficients are extracted from Mel-bank energies by the DCT transform. DCT is a linear transform.

benefit from tying the temporal vector length and the window length for mean and variance normalization. Both can be tuned separately. This normalization was disabled in performed experiments. The main focus of this thesis is on the acoustic modelling and if this normalization is applied, it can influence other parameters, mainly the optimal length of temporal context.

4.2.2 Windowing and normalization across the data set

The window used to select the trajectory out of the evolution of critical band energy has no effect in the TRAP system. The window is canceled out by the mean and variance normalization across the training data set:

$$\tilde{\mathbf{x}} = \frac{\mathbf{x} - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \tag{4.1}$$

where $\tilde{\mathbf{x}}$ is normalized vector, $\mathbf{x}$ is input vector. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are mean vector and vector of standard deviations, both estimated from all vectors in the training set:

$$\boldsymbol{\mu} = \frac{1}{F}\sum_{i=1}^{F} \mathbf{x}_i \tag{4.2}$$

$$\boldsymbol{\sigma}^2 = \frac{1}{F}\sum_{i=1}^{F} (\mathbf{x}_i - \boldsymbol{\mu})^2 = \frac{1}{F}\sum_{i=1}^{F} \mathbf{x}_i^2 - \left(\frac{1}{F}\sum_{i=1}^{F} \mathbf{x}_i\right)^2 \tag{4.3}$$

F is the number of frames in the training set. When equations 4.2 and 4.3 are substituted into equation 4.1, it can be easily seen that both vectors with weighting window applied and without weighting window are equal after normalization.

$$\frac{\mathbf{w}\mathbf{x} - \frac{1}{F}\sum_{i=1}^{F} \mathbf{w}\mathbf{x}_i}{\sqrt{\frac{1}{F}\sum_{i=1}^{F} \mathbf{w}^2\mathbf{x}_i^2 - \left(\frac{1}{F}\sum_{i=1}^{F} \mathbf{w}\mathbf{x}_i\right)^2}} = \frac{\mathbf{x} - \frac{1}{F}\sum_{i=1}^{F} \mathbf{x}_i}{\sqrt{\frac{1}{F}\sum_{i=1}^{F} \mathbf{x}_i^2 - \left(\frac{1}{F}\sum_{i=1}^{F} \mathbf{x}_i\right)^2}} \tag{4.4}$$

$\mathbf{w}$ is weighting window vector.

4.2.3 Mean and variance normalization across the data set and ANN training

This normalization is related to the training of neural networks. The main benefit from it is a faster training and reduced chance to get stuck in a local minima during criterial function optimization. The effect of the normalization is illustrated in Figure 4.2. Let's suppose just one neuron with two inputs:

$$y = \mathcal{F}(w_1 x_1 + w_2 x_2 + t) \tag{4.5}$$

where y is the output of the neuron, $\mathcal{F}$ is a nonlinear output function (for example sigmoid), w_1 and w_2 are weights (we want to train them), x_1 and x_2 are inputs and t is a threshold (also trained). The weights and the threshold are set randomly in certain dynamic range at the beginning of the training. The dashed line defining division of space[5] by the neuron is:

$$\begin{aligned} 0 &= w_1 x_1 + w_2 x_2 + t \\ x_1 &= -\tfrac{w_2}{w_1} x_2 - \tfrac{t}{w_1} \end{aligned} \tag{4.6}$$

[5] In this case a 2D plane

The threshold t moves the discrimination line up and down. The points represent two classes in feature space (for example phonemes). They can be far away from the center of axes if no normalization is applied. It is necessary to run many training iterations to move the discrimination line closer to the data clusters (Figure 4.2a). If only mean normalization is applied, the dynamic range of weights and threshold is not necessarily appropriate to the dynamic range of data. The discrimination line can be even out of the data points and again, the training will need more iterations (Figure 4.2b). Figure 4.2c shows the data points after mean and variance normalization.

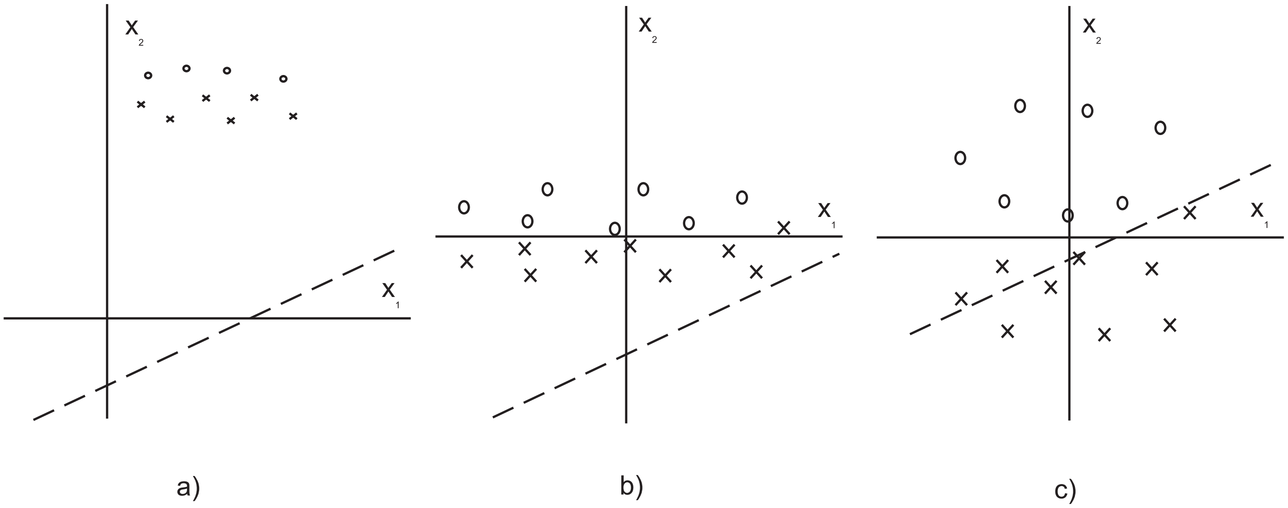

Figure 4.2: *Effect of mean and variance normalization across data set – two class example: a) without normalization, b) with mean normalization, c) with mean and variance normalization.*

4.2.4 Comparison to systems based on classical features

In this section the TRAP system is compared to two classical systems: a hybrid system based on MFCC39 and a hybrid system based on multiple frames of MFCC39. The optimal number of consequent frames for the multiframe system was experimentally found to be 4. The TRAP system reached slightly better PER than the MFCC39 multiframe system, but the improvement to the pure MFCC39 system is significant. The results are in Table 4.4.

	ins	sub	del	PER
MFCC39	4.7	20.6	12.4	37.7
MFCC39 – 4 frames	5.5	19.0	9.6	34.1
TRAPS – 1 second	4.3	18.6	10.9	**33.8**

Table 4.4: *Comparison of systems based on MFCC to the TRAP system.*

4.2.5 Optimal length of temporal context

The one second long temporal context, usually used in literature [21], is not necessarily optimal.

Some weights of neural networks could be uselessly spent on parts of temporal context with a little relevant information. We may also not have enough training data to extract this information. Therefore the optimal length was found experimentally. The length of TRAP is being increased from 100 ms to 1 second and the PER is evaluated.

It is very important not to use mean and variance normalization of temporal vector for this experiment. These normalizations dramatically increase PER for short contexts and bias the experiment.

Figure 4.3 and Table 4.5 show the results. The optimal length is about 300 ms $\div$ 400 ms. It means using 150 ms $\div$ 200 ms to the future and 150 ms $\div$ 200 ms to the past. The optimal temporal context length is shorter than the 1 s used by other authors. The fact that shorter input is effective may have positive implications in applications where minimal algorithmic delay is required. During other experiments not described here, the optimal length was found to depends on task (it is longer for digit recognition), on the size of neural network and on the amount of the training data. The PER is already much better than for the MFCC39 multiframe system which is a proof that longer temporal context is usefull.

length (ms)	110	210	310	410	510	610	710	810	1010
PER (%)	33.6	**31.3**	**31.3**	**31.3**	31.6	32.0	32.2	32.6	33.8

Table 4.5: *Effect of temporal context length in the TRAP system.*

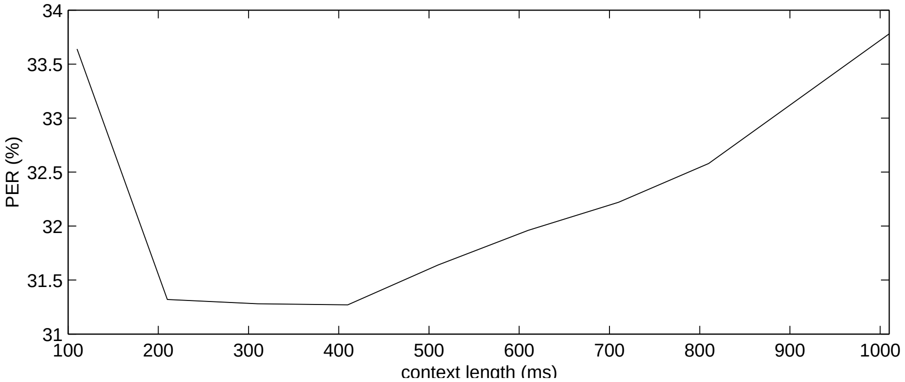

Figure 4.3: *Effect of temporal context length in the TRAP system.*

4.2.6 How to see band classifiers?

The TRAP system is quite complex due to the band neural networks. The huge complexity makes the system slow and hardly usable in practical application. Are the band neural networks necessary? What do they do? Let us evaluate whether the band neural networks work more like nonlinear mapping functions or like classifiers at first.

The band neural networks represent nonlinear mapping functions $\mathcal{F}_i(\mathbf{o}_i)$ where $i \in (1, N)$ is band index and $\mathbf{o}_i$ is input vector for the network. Let us say we will see the band neural network as a nonlinear mapping function if the mapping is invertible.

$$\mathcal{F}_i^{-1}(\mathcal{F}_i(\mathbf{o}_i)) = \mathbf{o}_i \tag{4.7}$$

This means that the data belonging the different classes are not overlapped after transformation. We will see band neural network as a classifier if the transformation function is not invertible.

$$\mathcal{F}_i^{-1}(\mathcal{F}_i(\mathbf{o}_i)) \neq \mathbf{o}_i \tag{4.8}$$

The following experiment brings more insight to this question. Two TRAP systems are compared. Both use 310 ms long temporal context. The first is a classical TRAP system. Band phoneme posteriors in the second system are quantized to a high value for the maximal posterior probability (winning phoneme), and to a low value for the other posteriors. The results are in Table 4.6.

	ins	sub	del	PER
classical TRAP	4.1	17.4	9.8	31.3
hard classification TRAP	4.7	25.2	14.1	44.0

Table 4.6: *Comparison of classical TRAP system and hard classification TRAP system.*

The classification TRAP system reaches 12.7 % worse result. The imagination of band neural networks to be mapping functions is closer. The merger does not only use the information about classes but it uses also the distribution of data points inside the classes. The data points for different classes overlap.

4.2.7 Band neural networks and different lengths of temporal context

The TRAP system was studied as a whole until now. Another important knowledge for deeper understanding can be obtained if the system is analyzed part by part. The optimal length of temporal context was studied for band neural networks. The results can be seen in Table 4.7 and in Figure 4.4. The position of minima are very interesting. They are behind 500 ms. This means that despite the optimal context length for the whole TRAP system is between 300 ms and 400 ms, the band classifiers are able to extract useful information about phonemes from even longer temporal context (more than 500 ms).

length (ms)	110	210	310	410	510	610	710	810	1010
band 1	73.8	72.0	71.1	70.7	**70.1**	70.8	70.3	70.4	70.7
band 5	68.2	65.6	64.5	64.1	**63.3**	64.0	64.2	63.7	64.4
band 23	75.8	72.6	71.1	70.4	70.3	**69.8**	69.8	70.1	69.7

Table 4.7: *Frame error rates for different lengths of temporal contexts and three different band classifiers from the TRAP system.*

4.2.8 Discussion

The main motivation for the TRAP system presented in literature is greater robustness against channel change and noise due to independent processing of frequency band and ability to extract information from a longer temporal context.

The later motivation was verified to be correct. The longer temporal context brings new information and moves data points representing different phonemes further apart in feature space. Therefore the system is more robust.

The former motivation was not verified yet. The greater robustness can come from mean and variance normalization of temporal vectors (not applied here). But the normalization can be done separately on in the structure of a classifier. The hierarchical structure of neural networks

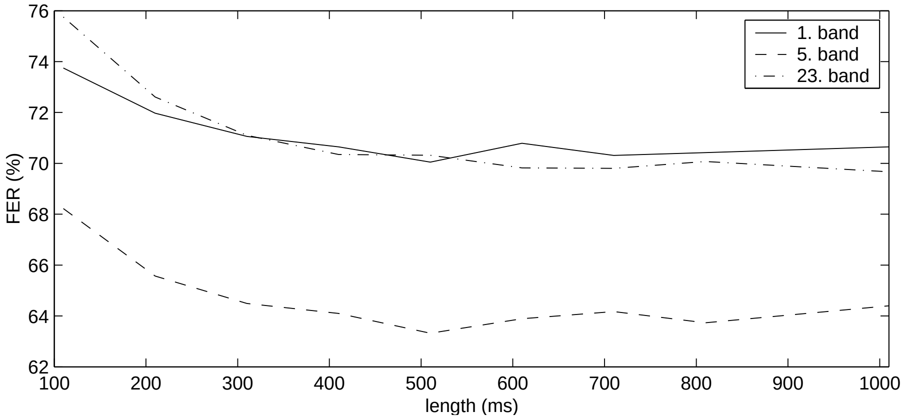

Figure 4.4: *Frame error rates for different lengths of temporal contexts and three different band classifiers from the TRAP system.*

can still perform just a nonlinear mapping function. It is not able to find which information is incorrect and selectively discard this information.

The purpose of band neural networks needs a deeper investigation. The experiments indicate that the purpose of these nets is not classification to phonemes for a simple decision in merger, but rather a data preprocessing for merger. Otherwise the optimal temporal context length would be similar for both the bands and the whole system.

4.3 Simplified system (one net system)

The TRAP system is complex and runs slowly. Even the experiments are slow, therefore the TRAP system is simplified. The simplification is necessary also for a better understanding of the whole system.

The band neural network represents a nonlinear mapping function. Let us replace this nonlinear function with a linear one: a linear transform is estimated instead of neural network weights and biases. And let's go further and omit the mapping to phonemes. The assumption is that the useful information is characterized by a variance in data. The Principal Component Analysis (PCA), see section 2.4.4, is used to estimate the linear transform. One transform is estimated for each band. The obtained base components are shown in Figure 4.5. These bases are very similar to Hamming window weighted Discrete Cosine Transform (DCT) bases, therefore a simplification to DCT was also tested. An experiment confirmed that the DCT degraded the results negligibly, therefore the DCT transform is used in the following experiments. A dimensionality reduction follows the linear transform. Network training can be helped by optimal choice of the dimensionality of input feature vector.

	ins	sub	del	PER
simplified system	3.7	16.6	9.6	**29.9**
TRAP system	4.1	17.4	9.8	31.3
TRAP + DCT	4.0	17.3	9.8	31.1

Table 4.8: *Comparison of simplified system and the TRAP system.*

Comparison of the simplified system to the TRAP system in terms of PER can be seen in

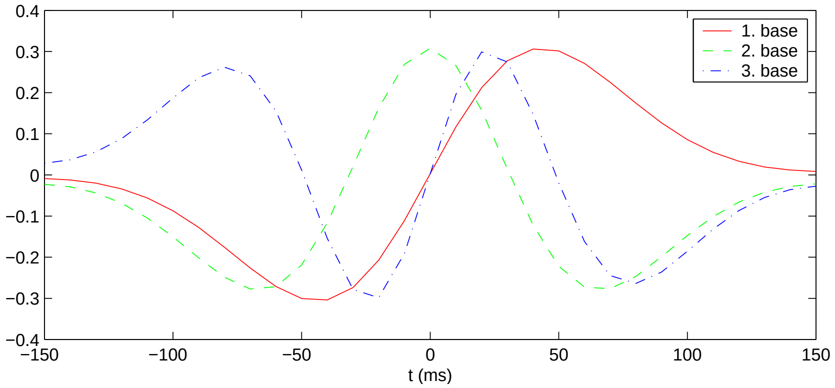

Figure 4.5: *First three bases of the PCA transform applied to temporal vectors in the 5th band.*

Table 4.8. The length of temporal vectors is 310 ms and 16 DCT coefficients were kept. The experiment showed that the linear transformation is enough. The simplified system gives even better results than the complex TRAP system.

For investigation of the effect of nonlinear transforms in bands, the DCT and dimensionality reduction were applied also before band neural networks in the TRAP system. This was done previously by František Grézl but without any explanation [24][42]. This approach reached better result than the TRAP system but worse than the simplified system. This could mean that band neural networks do something similar as chain of windowing, DCT and dimensionality reduction. This chain is discussed thoroughly in the following subsections.

4.3.1 Weighting of temporal vectors and DCT

The weighting of temporal vectors has no effect in the TRAP system. It was canceled out by the subsequent normalization. The situation changed in the simplified system, the weighting start to be beneficial. Let us see an experiment. The simplified system was trained with and without DCT and with or without Hamming window. The results are in Table 4.9.

The first two rows indicate that it does not matter whether the window is applied or not if the DCT is not applied. Precisely, the result with window is even worse, but this can be just a bad luck as the training algorithm got stuck in a local optimum. If the DCT is applied (third row), the result is significantly better. The improvement comes from smaller patterns (less parameters at the input of the network). The dimensionality reduction implies the fact that the temporal trajectory can be down-sampled twice without any degradation in accuracy. This had been already found in [43] and [44]. The DCT with dimensionality reduction can be also seen as a kind of temporal filtering, similar to RASTA [41]. Here, smaller and smoother patterns imply less trainable parameters in the neural network and less chance to get stuck in local optimum during the training. If the window is applied together with DCT (last row), the result is even better. The DCT saved the window and it was not canceled out by the normalization! The window attenuates values at the edges of temporal context, so the training algorithm can focus to the center of the context during the initial phase of training.
Why is the attenuation important? At first, we can look at histograms of values at different places of the temporal vector (Figure 4.6). The histogram is narrow for the center (the variance is low). Then the width grows and it is the highest at the edges. The trajectory in feature space representing a phoneme is affected by neighboring phonemes. The DCT tries to describe the input pattern by first few bases in such a way that the variance in the pattern is preserved.

	ins	sub	del	PER
no window, no DCT	4.2	18.0	10.4	32.6
Hamming, no DCT	4.0	18.5	10.5	33.0
no window, DCT	4.2	17.3	9.2	30.7
Hamming and DCT	3.7	16.6	9.6	**29.9**

Table 4.9: *Effect of windowing of temporal vectors (PER).*

The DCT features must be definitely focused to the edges if no window is applied. The window allows to describe the central part of context with a better resolution.

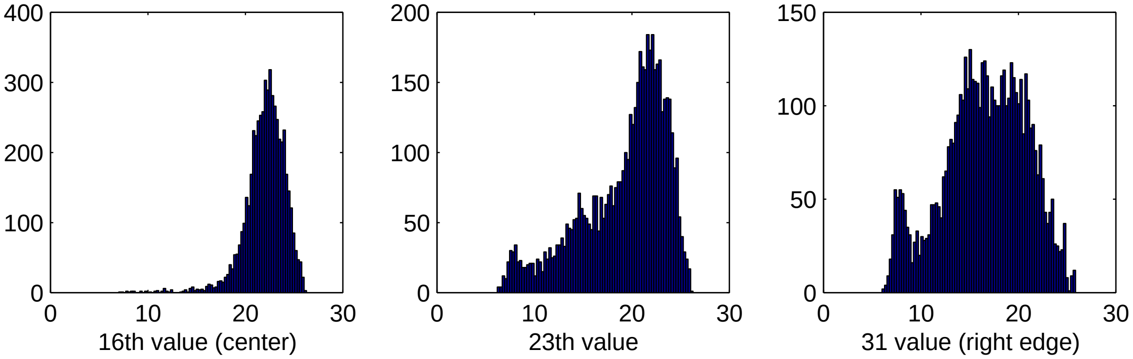

Figure 4.6: *Histograms of values at different places of temporal vector, 5 th band, phoneme aa.*

If we know that the windowing is important, another experiment can be done. The simplified system without DCT is taken and the window is applied after mean and variance normalization across the data set (sections 4.2.2 and 4.2.3). The results can be seen in Table 4.10. The PER is 2.2 % better than for the simplified system where the window is applied before normalization and canceled out.

	ins	sub	del	PER
no window, no DCT	4.2	18.0	10.4	32.6
weighted norms	3.6	17.1	10.1	30.8

Table 4.10: *Effect of feature vector scaling before variance normalization.*

These experiments brought also a possible explanation for the purpose of the band neural networks in the TRAP system. The patterns being presented to these networks are very simple and therefore the training algorithm is successful and the patterns can be longer. The phoneme posterior probabilities are good features (simple and smooth) for the merger. The merger has less chance to get stuck in a local optimum.

4.3.2 What weighting window shape is optimal?

When we know that the window can have very beneficial effect to the performance of simplified system, we can ask: Is the Hamming window optimal? Probably we should use a window derived from the within-class variance at different places of temporal vectors, but let's try few different shapes at first to get a feeling. The different windows are shown in Figure 4.7 and the results obtained are in Table 4.11

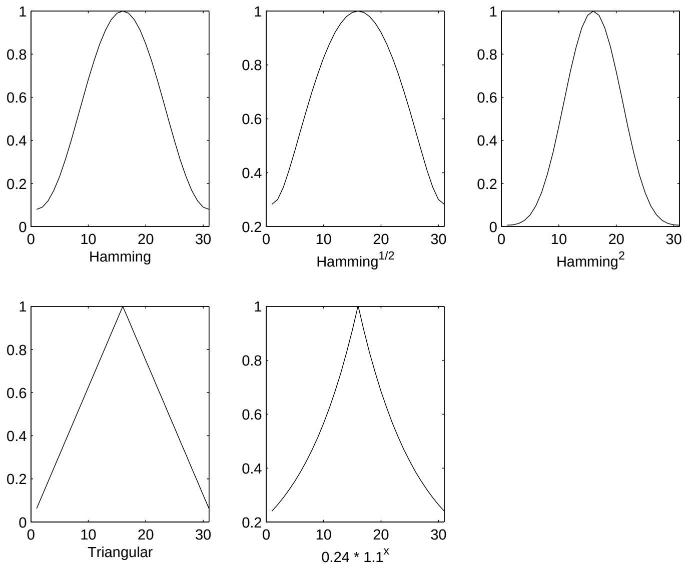

Figure 4.7: *Shapes of weighting window applied to temporal vectors.*

	ins	sub	del	PER
Rectangular (none)	4.2	17.3	9.2	30.7
Hamming	3.7	16.6	9.6	29.9
Hamming$^{1/2}$	4.9	17.2	8.5	30.6
Hamming^2	3.9	17.1	9.5	30.5
Triangular	3.5	16.9	8.7	29.1
0.24×1.1^x	4.5	16.7	7.6	**28.8**

Table 4.11: *Effects of different weighting windows applied to temporal vectors (PER).*

The Hamming window is not the optimal one, as was expected. As can be seen, the narrower window the better window. The best shape from the investigated ones is an exponential window. The decrement in PER between the best investigated window and the Hamming window is 1.1 %.

Because this experiment was done later in time, the Hamming window is used in the following experiments.

4.3.3 Comparison of different linear transforms applied in bands

For completeness the comparison between the DCT and PCA transforms is given. This experiment was done on TIMIT database down-sampled to 8000 Hz, therefore the results are not directly comparable with other experiments.

For each transformation, the number of kept base components was varied. The reason is simple. The higher bases describes too small changes in patterns and these bases can be noisy. The aim is to find how many bases are useful for classification. Transforms were applied to 310 ms long temporal vectors weighted by the Hamming window. The PCA was estimated for each band separately. The results can be seen in Table 4.12 or better in Figure 4.8.

The PCA gave slightly better PER than DCT but this difference is not significant. The PCA needs to keep less bases. This is more interesting because it can have a positive influence in an application: the following classifier can be smaller.

# coef	5	7	8	10	12	15	20	25	30
DCT	37.6	34.5	-	33.4	**33.2**	33.4	33.6	33.4	34.0
PCA	35.7	33.8	33.4	**33.0**	33.3	33.3	34.0	34.1	34.3

Table 4.12: *Effects of different linear transformations applied to temporal vectors in simplified system (8 kHz, PER).*

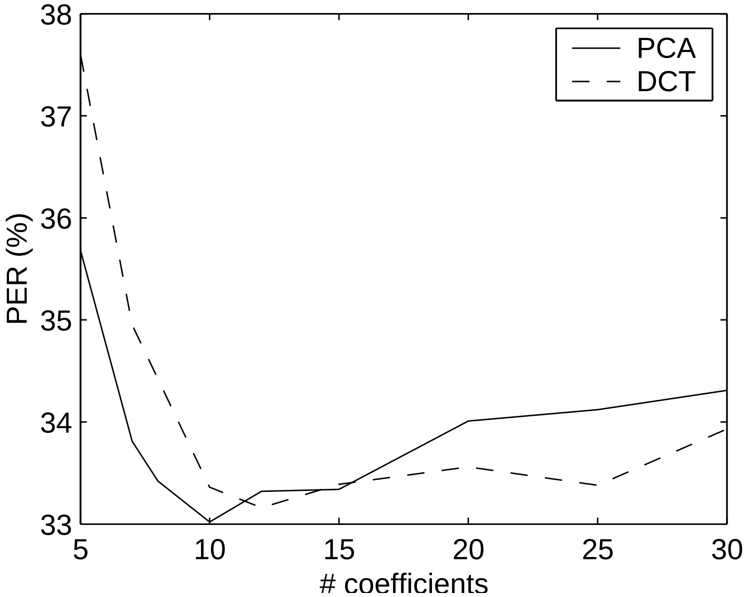

Figure 4.8: *Effect of different linear transformations applied to temporal vectors in simplified system (8 kHz).*

4.3.4 The Discrete Cosine Transform as a frequency filter

The DCT applied to temporal vector can be seen as a modulation frequency band-pass filter. What are the important frequencies that needed to be modelled? The lower frequency limit is given by the length of temporal vector. If the length is higher, lower frequencies can be modelled. The upper frequency limit is given by the number of used DCT coefficients. But the number required DCT coefficients to keep a constant upper frequency limit grows also with the length of the temporal vector. Is it better to keep the input for neural network constant and model narrower frequency range for longer context, or is it better to increase the input and keep the frequency range constant? The following experiment gives answers to these questions. The optimal length of temporal context is evaluated for fixed number of DCT coefficients ($15 + C_0$) and then the number of DCT coefficients is varied according to equation:

$$n_{DCT} = \frac{context\_length}{2} + 1 \tag{4.9}$$

This equation ensures fixed upper frequency limit. The context length is in frames (10 ms units). The results are in Table 4.13 and in Figure 4.9. Both lower frequency and upper frequency limits are reported. The optimal length of temporal contexts is about 300 ms for both cases. This is similar as for the TRAP system. It is definitely better to keep the upper frequency limit constant (to increase the number of DCT coefficients), as can be seen from the figure. It is possible to get an additional information using a longer temporal context, but it is necessary to model the whole trajectory with equal variance (detail) as before.

	length (ms)	110	210	310	410	510	610	710	810	1010
	lower f_m (Hz)	4.6	2.4	1.6	1.2	1.0	0.8	0.7	0.6	0.5
fixed	upper f_m (Hz)	68.2	35.7	24.2	18.3	14.7	12.3	10.6	9.3	7.4
# DCT	# DCT	16	16	16	16	16	16	16	16	16
	PER (%)	-	-	**29.9**	30.2	30.8	32.4	33.9	35.7	39.6
varied	upper f_m (Hz)	22.7	23.8	24.2	24.4	24.5	24.6	24.7	24.7	24.8
# DCT	# DCT	6	11	16	21	16	31	36	41	51
	PER (%)	34.5	30.8	**29.9**	30.4	30.6	30.8	31.5	31.3	32.4

Table 4.13: *Effect of temporal context length for fixed and varying number of DCT coefficients (the f_m is modulation frequency).*

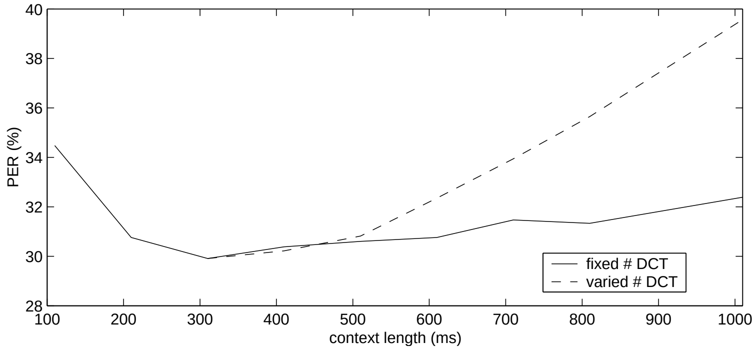

Figure 4.9: *Dependency of PER on temporal context length for fixed and varied number of DCT coefficients.*

4.4 3 band TRAP system

Pratibha Jain showed [23] a benefit from using temporal vectors form neighboring frequency bands as an input to the band neural network in the TRAP system. The input patterns created by the concatenation are still not too complicated and the band network has more relevant information. It would not be fair to compare other techniques to the basic TRAP system if a better variant exists, therefore the 3 band TRAP system was also evaluated. The comparison is summarized in Table 4.14. The 3 band TRAP system gives significantly lower PER. The PER is comparable with results obtained by the simplified system.

system	PER
1 band TRAP	31.3
3 band TRAP	29.2

Table 4.14: *Comparison of one band and 3 band TRAP systems.*

4.5 Study of amount of training data

The phonemes are represented by trajectories in the feature space. There is not one trajectory for one phoneme, but many. The number grows with the length of temporal context. Let us consider one phoneme: this phoneme can be affected by 39 phonemes on the left and by 39 phonemes on the right. Each of these phonemes can be affected by 39 others. The number of trajectories will grow exponentially.

Let's study the amount of data we have in the database for certain lengths of the temporal context. The average phoneme length is a good unit for measurement. The task can be simplified and the n-grams statistics can be used[6].

Table 4.15 shows the coverage of n-grams in the test part of TIMIT database. The most important columns are the third (numbers in brackets) – percentage of n-grams occurring in the test part but not in the training part, and fourth – error which would be caused by a decoder if the unseen n-grams are not allowed. The error is calculated by sum of occurrences of unseen n-grams divided by sum of occurrences of all n-grams:

$$error = \frac{\sum_{i \in N} C\left(n_i\right)}{\sum_{i \in A} C\left(n_i\right)} \tag{4.10}$$

N is a set of unseen n-grams, A is a set of all n-grams and $C\left(n_i\right)$ gives number of occurrences of n-gram n_i in the test part of database.

For bi-grams, there are 2.26 % of unseen cases but this amount causes almost no error (0.13 %). The situation is much worse for trigrams with 18.83 of unseen cases causing 7.60 % of error. It is almost impossible to model four-grams due to 44.10 % of error. These errors can be expected to be smaller in case of larger databases but still the maximum possible length of context seems practically to be three times or four times phoneme length due to exponential growth of error.

To conclude, the most limiting issue for a system based on long temporal context is the amount of training data because the demand for data grows exponentially with the temporal context length. This situation force us to look for a way around. One solution is to collect huge databases. Current systems use more than 1000 hours of training data [45]. This system just 2.5 hours. The collecting and annotation of new databases is very costly. But it is the mostly used way today. Another solution is the development of clever algorithms. This way is chosen for this thesis.

n-gram order	# different n-grams	# not seen in the train part	error (%)
1	39	0 (0.00%)	0.00
2	1104	25 (2.26%)	0.13
3	8952	1686 (18.83%)	7.60
4	20681	11282 (54.55%)	44.10

Table 4.15: *Numbers of occurrences of different n-grams in the test part of the TIMIT database, number of different N-grams which were not seen in the training part and error that would be caused by omitting unseen N-grams in the decoder.*

[6] Note that we never use those n-grams in phoneme recognition, it is just a tool to show amounts of sequences of different lengths!

Chapter 5

System with split temporal context (LC-RC system)

5.1 Motivation

The study of amount of data needed to train a long temporal context based system (section 4.5) showed that very large databases are necessary. A development of techniques that need less data and limit the cost spent on data collection and annotation would be beneficial. This chapter investigates one such technique. This technique is inspired by the function of band neural networks in the TRAP system and Table 4.15.

If we are not able to classify long trajectories in the feature space because there are simply many of them and very big portion was not seen during training, let us to split the trajectores into more parts.

These parts can be modelled separately and then the results can be merged together. An assumption of independence is done. Obviously by the split, a part of information is lost.

Let us see what will happen if the trajectory is split into two parts on n-gram statistics. All trigrams were split into two bigrams. The error caused by unseen trigrams 7.60 % was replaced by two times the error of bigrams which is only 2 × 0.13 % = 0.26 %. For four-grams, the error was reduced from 44.10 % to just 15.2 %. The reduced errors are summarized in Table 5.1.

n-gram order	# different n-grams	# not seen in the train part	error (%)	reducted error (%)
2	1104	25 (2.26%)	0.13	0.00
3	8952	1686 (18.83%)	7.60	0.26
4	20681	11282 (54.55%)	44.10	15.2

Table 5.1: *Effect of splitting trajectories into two parts – reduced errors. All other columns are unchanged.*

5.2 The system

The experimental system is derived from the simplified system described in section 4.3. The Mel-bank energies were extracted and the 310 ms long temporal vectors (31 values) of evolution of critical bank energies were taken. Each temporal vector was split into two parts – left part (values 0 - 16) and right part (values 16 - 31). Both parts were windowed by corresponding half of Hamming window and projected to the DCT bases. 11 DCT coefficients were kept for each part. Such preprocessed vectors were concatenated together for each part of context separately and sent to two neural networks – these are trained to produce phoneme posteriors, similary as in the TRAP system. Output posterior vectors are concatenated, transformed by logarithm and sent to another (merging) neural network trained again to deliver phoneme posteriors. Finally, the phoneme posteriors are decoded by a Viterbi decoder and strings of phonemes are produced. The whole process is illustrated in Figure 5.1. This system is called the Left context – Right context system, or shortly LC-RC system.

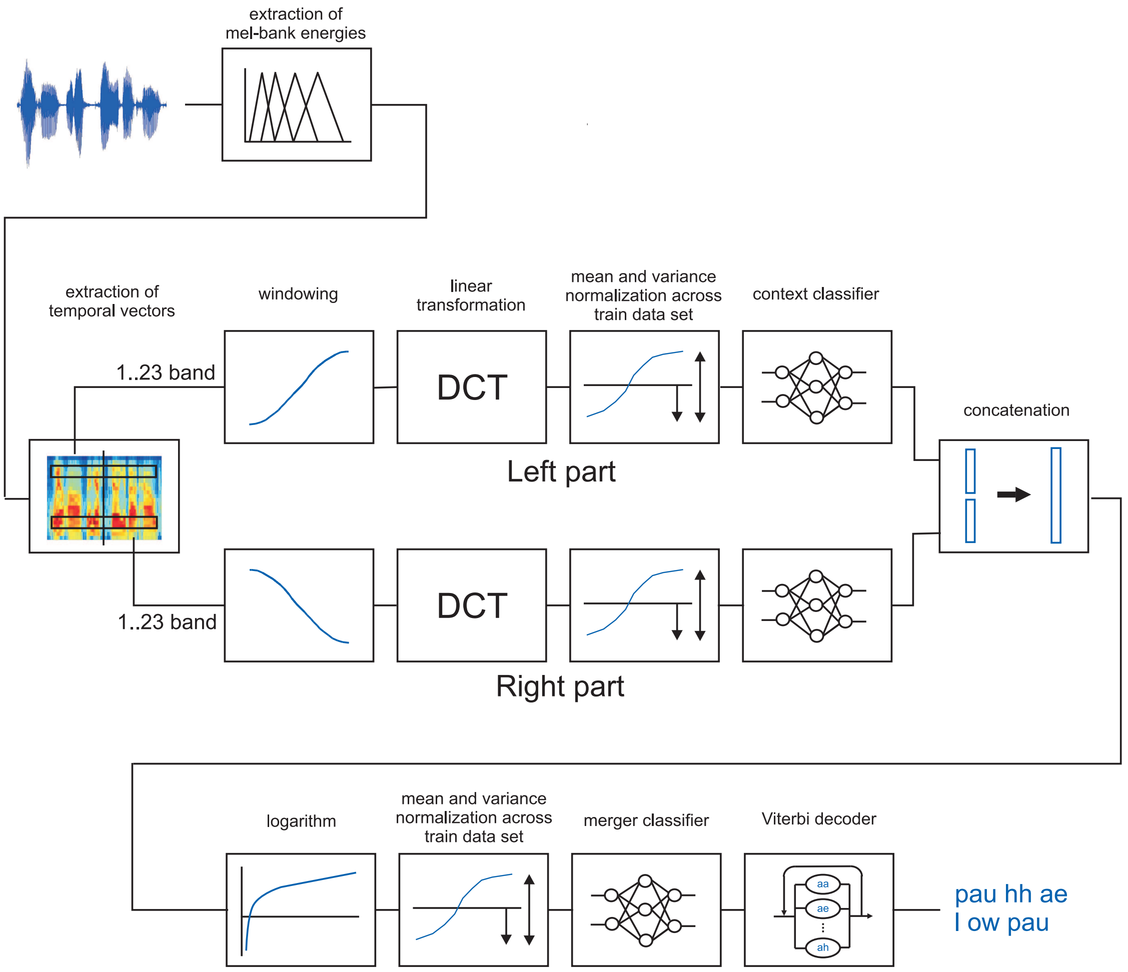

Figure 5.1: *Block diagram of the Split Temporal Context system.*

5.3 First result and comparison to the simplified system

The LC-RC system was compared to the simplified system. The results are in Table 5.2. The RC-LC system reached significanlty better result. The motivation was proven to be correct despite the independence assumption.

system	ins	sub	del	PER
simplified	3.7	16.6	9.6	29.9
LC-RC	4.0	15.4	9.0	**28.4**

Table 5.2: *Comparison of the LC-RC and simplified systems.*

5.4 Modelled modulation frequencies

Where the improvement in the LC-RC systems comes from? The following experiment tries to answer the question. The optimal number of DCT coefficients was found for the left context. Table 5.3 and Figure 5.2 show the results. It is the best to include 14 DCT coefficients (almost all).

Now let us compare modulation frequencies modelled by both the LC-RC and the simplified systems. The comparison is in Table 5.4. The upper limit for modelled frequencies is much higher for the LC-RC system. The LC-RC system models the trajectory with lower variance (higher details). The two blocks also increase the temporal resolution. The remaining question is the drawback of the LC-RC system coming from not seeing frequencies bellow 1.67 Hz. Removing C_0 in the simplified system causes increment in PER as the information about vertical shifts in different bands is lost and is not seen by the network. However the lost in PER does not exeed 0.5 %.

# coef	6	8	9	10	11	12	13	14	16
upper f_m (Hz)	16.7	23.3	26.7	30.0	33.3	36.7	40.0	43.3	50.0
PER (%)	36.9	36.0	35.7	35.7	35.4	35.5	35.4	**35.2**	35.4

Table 5.3: *Optimal number of DCT coefficients (including C_0) for the left context in the LC-RC system and corresponding modulation frequencies.*

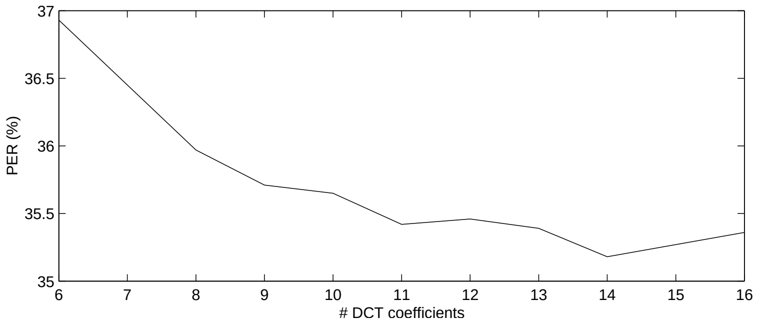

Figure 5.2: *Optimal number of DCT coefficients for the left context of the LC-RC system.*

5.5 Optimal lengths of left and right contexts

The previous experiment showed that the upper limit of modulation frequency used by the LC-RC is significantly higher than for the simplified system. If we have a more capable classifier, is not it worth to extend also the temporal context? At first, let us evaluate the optimal temporal context length for context networks. The results are in Table 5.5 and in Figure 5.3. The number

system	context length	optimum # coefs	lower f_m	upper f_m
	(ms)	(-)	(Hz)	(Hz)
simplified system	310	16	1.67	25.00
LC part	160	14	3.33	43.33

Table 5.4: *Comparison of minimal and maximal modulation frequencies for the left part in the LC-RC system and the simplified system.*

of DCT coefficients was set according to equation:

$$n = \text{int}\left(\frac{2}{3}\frac{len}{10}\right) \tag{5.1}$$

This equation ensures the upper limit of modulation frequencies constant (about 33 Hz). The operator "int" is rounding to the first lower integer.

len (ms)	100	120	140	160	180	200	220	240	260	310	360	410
LC PER (%)	36.5	36.0	36.3	35.4	35.8	**35.1**	35.2	35.4	35.8	35.9	35.8	36.5
RC PER (%)	37.8	38.0	37.4	37.7	37.2	**37.1**	37.2	37.5	37.3	37.4	37.7	38.1

Table 5.5: *Optimal length of left and right temporal contexts in the LC-RC system.*

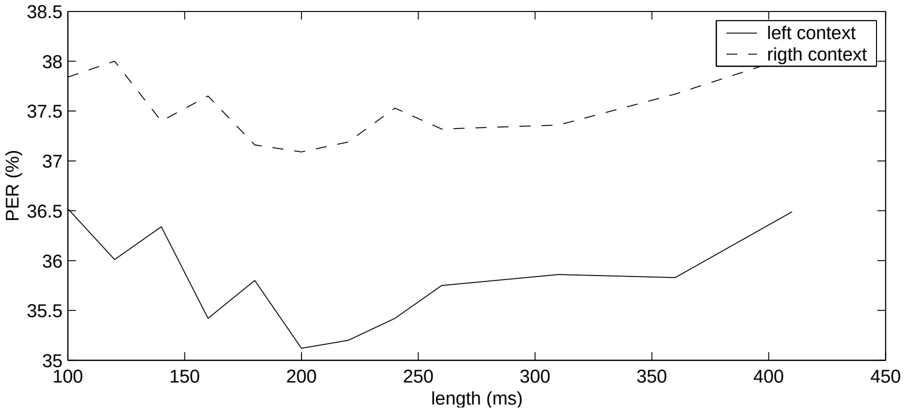

Figure 5.3: *Optimal length of left and right temporal contexts in the LC-RC system.*

The minima for both contexts are at 200 ms. This is interesting, because the full context is about 400 ms which is closer to the optimum for band neural networks in the TRAP system seen in section 4.2.7, where we know that useful information for classification is contained. The beginning of both graphs in Figure 5.3 seems to be quite noisy. The peaks partially disappear if more DCT coefficients are used. This suggests that the DCT transform is not the best choice to model higher modulation frequencies. The PER is better for the left contexts. This indicates that the signal at the beginning of phoneme is more important.

len (ms)	270	310	350	390	430	470	510	550
13 DCTs	28.5	27.9	27.8	27.8	27.8	**27.6**	27.8	27.9
16 DCTs	-	28.3	28.0	27.8	27.9	**27.6**	27.8	27.8

Table 5.6: *Optimal length of temporal context for the whole LC-RC system.*

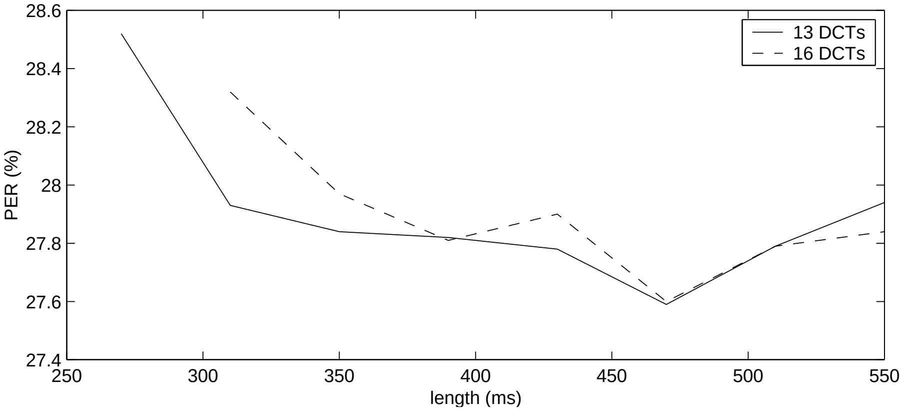

Figure 5.4: *Optimal length of temporal context for the whole LC-RC system.*

5.6 Optimal length of temporal context for the whole LC-RC system

An optimistic result from the previous section does not ensure that the whole system will use all the 400 ms given by sum of both optimal context lengths. Therefore the same experiment was repeated for the whole system. Both contexts have the same length. This time, the number of DCT coefficients was fixed to ensure stability in the initial part of graph. The results are in Table 5.6 and Figure 5.4. The optimal length is even higher than the sum of optimal lengths for both contexts! The optimal lengths of contexts for merging differ from the context lengths with minimal PER. The final part of the graph (crossing lines) shows again that it is important not to cut off the upper modulation frequencies.

5.7 Discussion

This chapter proved that the information usable for recognition of a phoneme is spanned across almost 500 ms. And we are actually able to extract the information! This chapter also brought more insight to the training of neural networks. It is beneficial to introduce some reasonable constrains coming from the task.

Although we see the optimal parameters, the later experiments are done with a shorter temporal context (310 ms) and less number of DCT coefficients (11 per context). The reason is comparison with the baseline systems, and also a faster turnover of experiments.

Chapter 6

Towards the best phoneme recognizer

The previous two chapters described the development of a good phoneme recognizer. The next goal described in this chapter was is to improve it as much as possible by adding techniques commonly used in speech recognition.

6.1 More states

One of the most common techniques in speech recognition are state models. The main purpose of states is the selection of particular views at features. The decoder proposes new direction of trajectory and a model in a state verifies whether the direction is correct or not. It is similar as if someone advices us a route. We can verify that we are still on the route according to some important objects at different places of the route. The more important objects we see the more sure we are.

Features based on long temporal context and ANN can see many important objects from one point. But the main benefit of states comes during the training. The training algorithm is focused to certain parts of phonemes. We guide the training. The focused patterns are easier and sharp. The weights are associated with certain parts of phoneme and in case of increasing the number of parameters of the network, we have a chance to decrease the error.

The hierarchical structure of neural networks also benefits from states during recognition. The lower network (band or context network) roughly estimates a place (state) where the recognition is in the feature space. The position is more precise with more states. The upper network (merger) uses the knowledge about this place and it can focus on details.

Another benefit is minimum phoneme duration. If three state models are used, the minimum duration of phoneme is 30 ms. This is good to prevent the decoder from switching of one phone to another, although this can be also enforced by repetition of existing HMM states or setting appropriate phoneme insertion penalty.

6.1.1 Implementation of states

The parametrization and neural network structure is unchanged for this approach. The neural networks were trained on force-aligned state transcriptions. The decoder was modified to force a pass through the state sequences in phoneme models. The phoneme models are left-to-right with no skip states.

6.1.2 Forced alignment

The state-level transcription is needed for this approach. Two approaches for generating state-level transcription were compared: forced alignment based on a classical HMM/GMM system, and force alignment based on the target HMM/ANN system. Both approaches provided a good state-level transcription and some differences in results were negligible. The later approach was adopted for the following experiments.

The forced-alignment starts with a uniform segmentation of phoneme labels into state labels. Then few iteration of training and realignment are done.

6.1.3 Results

The realignment does not bring any improvement for one-state models. Three iterations were sufficient for three-state models. Different one-state systems and three-state systems are compared in Table 6.1.

system	1 state	3 states	difference
MFCC, 9 frames	39.9	35.6	4.3
MFCC39	37.7	32.8	**4.9**
MFCC39, 4 frames	34.1	29.9	4.2
simplified system (310 ms, 11 $DCTs$)	29.9	28.7	1.2
3 band TRAPs	29.2	25.8	3.4
LC-RC system	28.5	**24.4**	4.1

Table 6.1: *Comparison of 1-state and 3-state systems*

The three state systems are able to significantly reduce the phoneme error rate. The LC-RC system profits about 4.1 % from the 3-state system. The simplified system has the smallest reduction.

6.1.4 Where does the improvement comes from?

Not the whole improvement is caused by finer representations of neural network outputs. A part of this improvement comes from the decoding process. To evaluate this, an experiment was done: posteriors from the three state system (with four vectors of MFCC39 features) were converted to one state posteriors by summing posteriors for each phoneme. This representation was sent to the decoder. Then a minimum duration of phonemes (3 frames) was fixed and the decoder was run again. The results are in Table 6.2.

posteriors	PER (%)
3 state	**29.9**
converted to 1 state	31.1
converted to 1 state, fixed minimum duration	31.1
1 state	34.1

Table 6.2: *Three state posteriors converted to one state posteriors in the MFCC39 system with four frames*

The improvement between one and three states is 4.2 %. We see that finer representation of neural network output removes 3.0 % from PER. The limitation of minimum phoneme duration

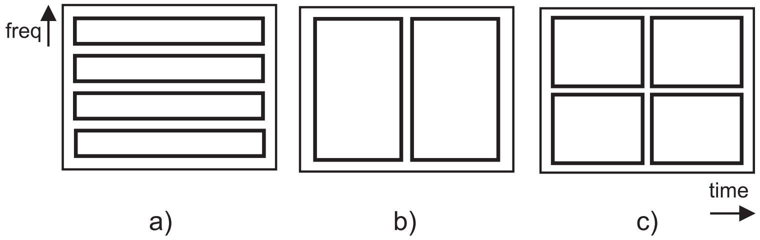

Figure 6.1: *Different time and/or frequency split architectures: a) TRAP system, b) LC-RC system, c) 2 x 2 system*

# bands per net	1	3	5	7	13
PER (%)	28.2	25.8	**24.8**	24.9	25.6

Table 6.3: *Optimal number of joint bands for band neural network in the 3 state multiband TRAPs system*

has no effect and 1.2 % comes from the three state structure in the decoder. The improvement in the decoder is not surprising: if three-state posteriors are summed within one phoneme, de facto a three state model with arbitrary order of states is created. We know however, that the order of parts of phonemes matters for the recognition.

This experiment showed that adding new information during training of neural networks helps a lot. An improvements was also seen by other authors when the neural network was trained for multiple tasks, for example for classification of speech frames and gender detection [46].

6.2 Other architectures

All the previous experiments indicated that the clue to build a good recognizer based on HMM/ANN is the ability to focus the training algorithm on well defined coherent segments with as descriptive features as possible. Let us experiment with some more variants of the TRAP and the LC-RC systems.

6.2.1 How many bands in the TRAP multiband system are optimal?

If the number of joint bands is small, the band neural network does not have enough information for classification, the error rate is higher and the input pattern for merger is very difficult. If the number of joint bands is higher, the band neural network input patterns start to be difficult. A tradeoff must be found. The optimal number of joint bands is evaluated in Table 6.3. For wideband speech, the optimal number is 5. Another experiment showed that 3 is optimal for narrow band speech.

6.2.2 Split temporal context system (STC) with more blocks

The trajectory in feature space representing phoneme can be split into more than two parts and a generalization of the LC-RC system can be done (see Figure 6.1b). In this experiment the optimal number of parts is found. The input temporal vectors are split to 2, 3 and 5 parts. The Hamming windows are applied to all parts followed by dimensionality reduction to 11, 8, and 5 bases by DCT.

# blocks	1	2	3	5
PER (%)	26.8	24.4	24.2	**23.4**

Table 6.4: *Optimal number of blocks in 3-state split temporal context system*

system	1 state	3 states
3 band TRAPs	29.2	25.9
5 band TRAPs	-	24.8
STC - 2 blocks (LC-RC)	28.5	24.4
STC - 5 blocks	-	**23.4**
2 x 2	-	24.1

Table 6.5: *Comparison of different time and/or frequency split neural network architectures.*

The tradeoff must be found even here. If the number of parts increases, the input pattern for merger also increases and starts to be difficult. The results can be seen in Table 6.4. The best number of blocks is 5. It may be even more, but this was not evaluated – the system starts to be slow and impractical.

6.2.3 Combination of both – split in temporal and split in frequency domain

The system is called ”2 x 2 system” – two temporal parts and two frequency parts (see Figure 6.1c). The system contains 5 neural networks (4 blocks and 1 merger). The preprocessing is similar to the preprocessing for the LC-RC system.

6.2.4 Comparison of the TRAP, STC and ”2x2” architecture

The architectures are compared in Table 6.5. The lowest PER is obtained by the 5 block STC system. But the PERs for the 5 band TRAPs and the ”2 x 2” systems are very close. This proved that both assumption – split in time and split in frequency – are helpful. It is not very important how the split is done. It is more important that the obtained patterns are easily learnable by the neural networks. The STC (LC-RC) system is used in later experiments because it needs less computer resources.

6.2.5 Tandem of neural networks

A tandem of two neural networks (Figure 6.2) is another possibility to reduce the phoneme error rate. The lower network is trained in classical way, for example to classify multiple frames of MFCCs to phoneme or state posteriors. The posteriors from lower network are sent to upper network together with the original input features (those seen also by the lower network). In my interpretation, the lower network prepares the phoneme or state space for the upper network: It could be said that the lower network it is able to roughly localize phonemes or states and the upper one performs the precise classification.

In Table 6.6, properties of concatenation of two and three neural networks are presented. The net replaced with the tandem is the left block network of the LC-RC system.
The second network (left panel of Figure 6.2) is able to add one percent. The third network (right panel) adds another 0.3 %.

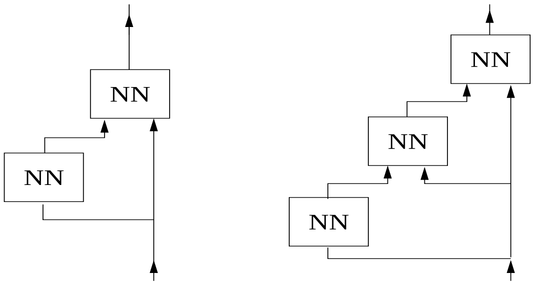

Figure 6.2: *Tandem architectures.*

# nets	1	2	3
PER (%)	31.6	30.6	30.3

Table 6.6: *Tandems of neural networks*

Relations to recurrent neural networks

Recurrent neural networks (RNNs) are reported to reach low phoneme error rates [19]. At least two links between RNNs and the approaches described above can be found:

1. RNN could be decomposed into two networks – lower network which generates the state vector and upper network using this state vector for finer classification. RNN actually works similarly as described above; one frame delay used in RNN does not really matter in comparison to lengths of contexts (around 30 frames).

2. RNNs creates the state vector implicitly, the size is usually 3 to 4 times the number of phonemes [19]. This information is actually used during the training similarly as it is described above.

The advantage of the tandem architectures over RNNs is that they are based purely on standard forward neural networks, common training algorithms and existing tools.

6.3 Tuning to the best performance

The STC with 5 blocks was taken and tuned to the best performance mainly by improved NN training: The scheduler for neural network learning rate was changed to use the *training set*. The scheduler halves the learning rate learning if the decrease in the frame error rate (FER) is less than 0.5% (the *cross-validation set* vas used before). The number of training epochs was fixed at 20.

Then, the numbers of hidden layer neurons in networks were increased from 500 to 800. I have seen that it was almost impossible to overtrain neural networks with 800 neurons in 20 epochs, therefore the CV set was added to the training one. At the end, bigram language model[1] estimated (without any smoothing) on phonetic transcriptions of the training part was included. All described steps are summarized in Table 6.7.

[1] Known as phonotactic model in language recognition

system	PER (%)
STC - 5 blocks	23.4
20 epochs in training	22.7
20 epochs in training + 800 neurons	22.1
+ CV part (18 minutes)	21.8
+ bigram LM	**21.5**[2]

Table 6.7: *Improvements to the 5-block STC system*

6.4 Discussion

This chapter showed that the results can be significantly improved by a few easy and cheap tricks – finer representation of neural network outputs, introduction of more independence assumption to the neural network structure, more epochs in neural network training and a language model.

Also, few other structures of neural networks were studied. Although for example the tandem structure seems to be very perspective, it is not used later due to its higher complexity and more difficult training. It is rather a motivation for an investigation of different neural network structures.

[2] This correspond to the classification error rate 17.2%

Chapter 7

Properties of investigated systems in different condition

7.1 Amount of training data

This section investigates a behavior of different systems when the amount of training data varies. The TIMIT database is definitely not the best database for this kind of study because of its size. It does not allow to study the systems in a area of saturation where the PER stops decreasing. On the opposite, all the systems were well tuned for TIMIT. I wanted also to know whether it is beneficial to use more states even if the amount of training data is extremely low. Therefore both one-state and three-state systems were investigated with varying amount of training data. The systems were repeatedly trained with 0.5 h up to 2.8 h of training data. The results can be seen in Figures 7.1 (1-state systems) and 7.2 (3-state systems), or in Tables 7.1 and 7.2[1].

Almost all systems are still far from saturation. The best PER reported in the previous chapter is already close to 20 %. It can be seen that this boundary could be easily crossed using more training data. The improvement coming from different systems is constant across different lengths of the training set. This indicates ability to extract some additional information by better systems, not just an ability to learn faster. But this assumption must be verified on a bigger database in the area of saturation.

Both 1-state and 3-state LC-RC system results were plotted in the same figure (Figure 7.2) to see the differences. The distance between both lines is constant, although much lower distance was expected for a smaller training set. This proves that using more states is a safe method, and it can be used even with very small training sets.

7.2 Robustness against noise

Some articles, for example about TRAPs, make an impression that the hierarchical structures of neural networks are more robust against noise than other approaches. This experiment should bring more insight into this issue. Three type of noises were artificially added to TIMIT database and the systems were trained and evaluated for each.

[1] The results can slightly differ compared to the results reported in previous chapters due to different versions of training software.

length (hours)	0.5	1.0	1.5	2.0	2.5	2.8
MFCC39	42.8	39.9	39.4	37.9	37.7	37.7
MFCC39, 4 frames	39.3	36.6	35.5	34.4	34.0	34.1
simplified system	36.2	33.2	31.8	31.0	30.3	29.9
TRAPs	38.8	35.5	33.7	32.4	31.6	31.2
3 band TRAPS	37.3	33.7	31.8	30.7	30.1	29.2
LC-RC system	35.6	32.1	30.8	29.4	28.4	28.5

Table 7.1: *Comparison of different 1-state systems trained with varying amounts of training data.*

length (hours)	0.5	1.0	1.5	2.0	2.5	2.8
MFCC39	38.7	35.6	34.9	34.2	33.4	32.8
MFCC39, 4 frames	35.4	33.5	31.5	31.2	30.0	29.9
simplified system	34.0	31.0	29.5	28.7	28.1	28.7
TRAPs	34.9	31.5	30.0	29.0	28.6	28.2
3 band TRAPS	32.8	29.3	27.9	27.3	26.5	25.8
LC-RC system	31.2	27.8	26.5	25.9	25.0	24.4

Table 7.2: *Comparison of different 3-state systems trained with varying amounts of training data.*

7.2.1 8 kHz versus 16 kHz speech

The noises were taken from the AURORA database. They are sampled at 8 kHz, therefore the TIMIT database was also down-sampled to 8 kHz. For consistency, the difference in PER between wide-band speech and narrow-band speech was also evaluated. It can be seen in Table 7.3. The highest degradation is seen for the MFCC39 system, LC-RC system and 5 block STC system. These systems are able to dig the most from higher frequencies. The TRAP system has a lower ability to use these frequencies, probably because of very big patterns for the merger. The worst ability to use higher frequencies can be seen for the simplified system. The input patterns are still too difficult and the training algorithm focus rather on major shapes at lower frequencies. A special case is the hard classification TRAP system (see section 4.2.6). The classification error rates of band classifiers working on high frequencies are so bad that these neural networks are not useful at all.

system	16 kHz	8 kHz	difference
MFCC39	32.8	35.7	-2.9
MFCC39, 4 frames	29.9	32.0	-2.1
simplified system	28.7	30.1	-1.4
TRAPs	28.2	30.0	-1.8
TRAPs - hard classification	44.0	43.9	0.1
3 band TRAPS	25.8	27.8	-2.0
LC-RC system	24.4	27.2	-2.8
5 block STC	23.4	26.8	-3.4

Table 7.3: *Difference between 16 kHz and 8 kHz 3-state systems (PER).*

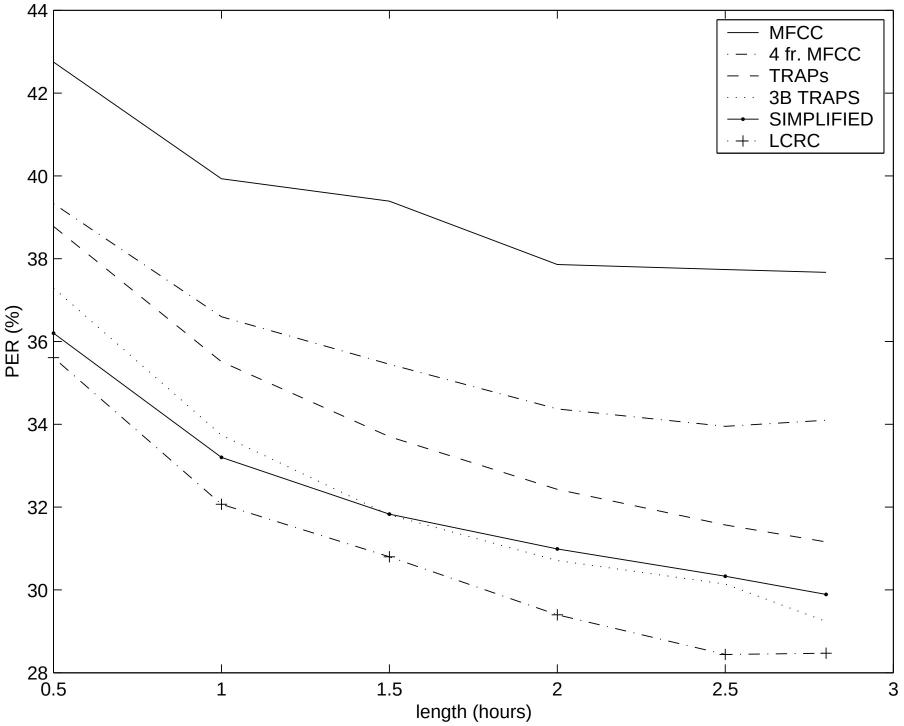

Figure 7.1: *Comparison of different 1-state systems trained with varying amounts of training data.*

7.2.2 Training and testing in the same noise condition

The behavior was investigated for four different SNRs levels and three different noises. The results can be seen in Table 7.4 or on Figures 7.4, 7.5 and 7.6. The dependencies of PER on SNR (excluding hard classification TRAP system) show parallel lines for different systems. This means no system is able to learn noisy patterns better than the others. Some system are just more accurate than others. The degradation in PER for different systems is constant. An exception is hard classification TRAP system. This system has less steep dependency and starts to be useful for very high SNRs (less than 0). The quantization of information is a useful technique for improving robustness against noise. All three noise types have similar tendencies.

7.2.3 Cross-noise condition experiments

The previous experiment investigated systems in the same training and testing condition. The ability of systems to learn noisy patterns was studied. Now let us see what happens if some new data never seen during training come to the input. The car noise was used. The systems are trained for five different noise levels and evaluated in each level. Table 7.6 gives absolute PERs. Table 7.6 than gives numbers relative to PERs where the noise levels match. The PER for the actual test noise level is subtracted from the PER of the training noise level. A value greater than 0 means that the system (given by row) can be used with success for the actual test noise level. To ensure certain PER, it is always better to train the recognizer on records with a higher noise level. The system is then able to recognize less corrupted patterns with good PERs.

Unfortunately this does not work if the target condition is clean speech. The patterns which the classifier sees start to be totally different. The difference comes mainly from the log used in

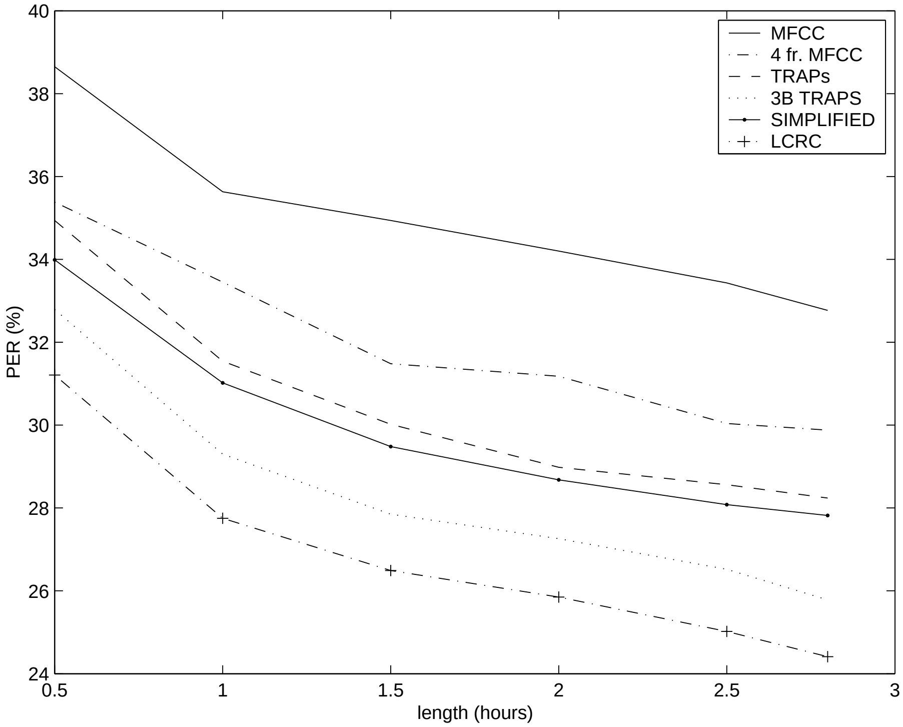

Figure 7.2: *Comparison of different 3-state systems trained with varying amounts of training data.*

the parametrization. In clean speech and unvoiced parts, the logarithm of energy can be close to $-\infty$.

Figure 7.7 compares systems trained for SNR15 visually. The TRAP system and the LC-RC system have more difficulties for lower SNRs than the MFCC systems. At SNR0, the absolute PER for the TRAP system is even worse than for the MFCC system. The bigger patterns (longer temporal context) have simply a higher chance to be corrupted.

Figure 7.8 repeats the same dependency for systems trained for SNR0. The figure demonstrates again that it is better to train a noise robust system on lower SNRs to guarantee the PER even for better SNRs. The curves show similar tendencies for all systems, except the TRAP system. It is working significantly worse for cleaner speech than the other systems. One explanation is: the band neural networks are forced to use the information from one critical band only, but the band patterns can be dangerously affected by the ”log problem” (the temporal vector always see an unvoiced region). Other systems can benefit from the frequency information.

7.3 Robustness against channel change

Not only noise can corrupt speech. The speech can be also corrupted by the transmission channel. This property was very briefly investigated in cross TIMIT-NTIMIT experiments. The NTIMIT database [47] has exactly the same structure and files as TIMIT database. It is the TIMIT transmitted across a telephone channel. Therefore also the training and test sets are the same. At first, the systems were evaluated on the same databases where the systems were trained on. Then the NTIMIT test set was recognized by the TIMIT systems (labeled T→N) and the TIMIT test set was recognized by the NTIMIT systems (labeled T→N). The results can

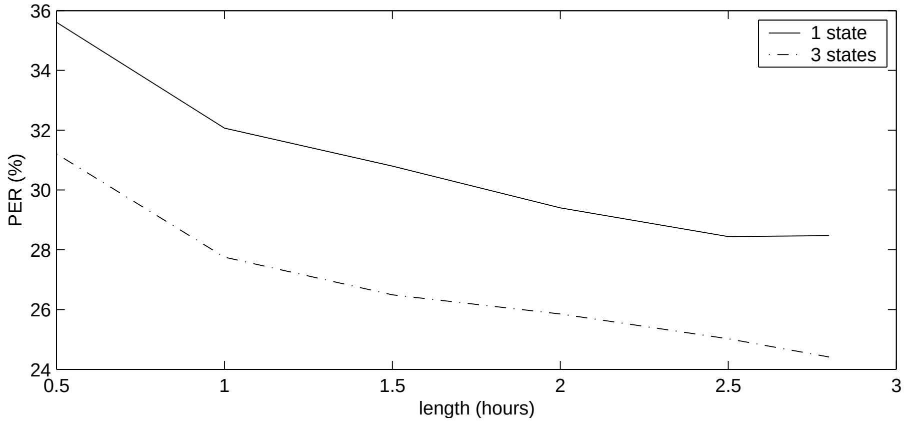

Figure 7.3: *Comparison of one and three state LC-RC systems trained with varying amounts of training data.*

be seen in Table 7.7. Also differences of cross-condition PER and the one obtained for matched condition are reported for better readability. Negative numbers mean degradation.

Using a system trained on clean speech to recognize more corrupted speech (TIMIT→NTIMIT) brings huge degradation. The degradation is similar to the one seen in previous cross-noise-level experiments. This raises an idea that the degradation could be partially removed by adding an artificial noise to the training records.

The TRAP systems have the smallest degradation. This proved the initial intention of authors to improve robustness of speech recognition systems using independent processing in bands to be right. The absolute PERs obtained by the TRAP systems are even better than those obtained by the STC systems. The least degradation comes from the hard classification TRAPs, but still, the absolute values are worse than for other systems.

Degradations for the opposite cross-condition (NTIMIT→TIMIT) are not so terrible as in the previous case. It is better to train the systems for a worse condition to ensure the PER. Again, the least degradation comes from the TRAP systems. The best absolute PER is obtained with the 3-band TRAP system, but just a slightly worse result (1 % worse) can be obtained also with the STC system.

7.4 Different databases

7.4.1 OGI Multilanguage Telephone Speech Corpus

Our lab works on language identification systems. Many researches train their phoneme recognizers on the OGI Multilanguage Telephone Speech Corpus [48][49] for this purpose, therefore our choice was to use this corpus too.

The corpus contains 11 languages, but just 6 languages have a phonetically labeled part. These languages are: English, German, Hindi, Japanese, Mandarin and Spanish. The phonetically transcribed records were split to three sets – the training set, the cross-validation set and the test set. The set lengths and number of phonemes for each language are summarized in Table 7.8.

The 3-state LC-RC system was trained for each language. The results can be seen in Table 7.9. The phoneme error rates are quite high for all languages. The difference to TIMIT is almost 17.5 %. The 3 % degradation comes from narrow band speech, another 2 % from less

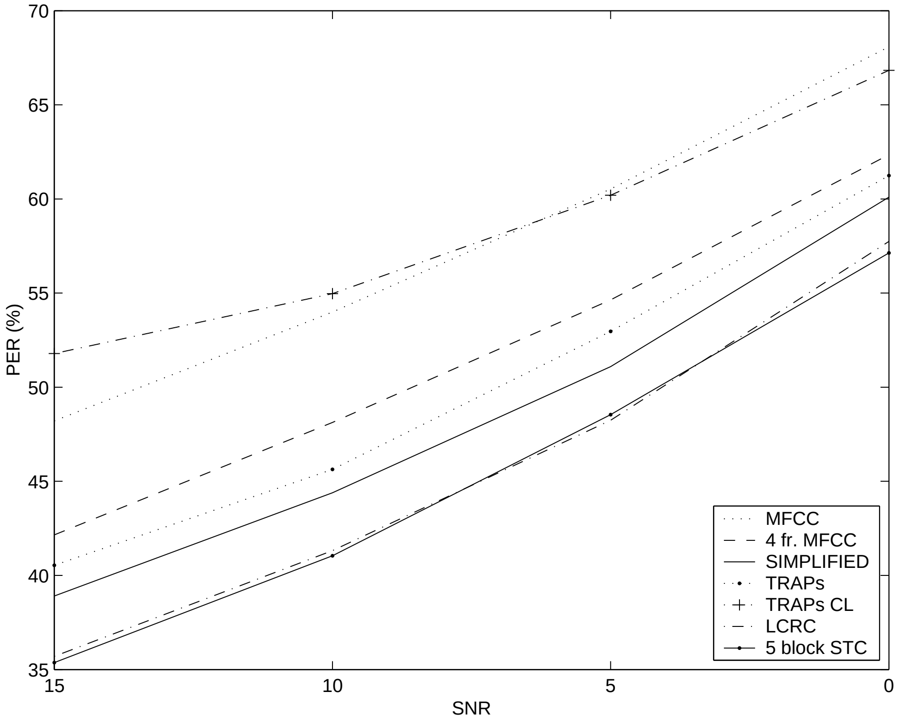

Figure 7.4: *Robustness of 3-state systems against car noise for the same training and test condition.*

data[2], but the degradation of 12.5 % is not expected and needs more investigation. It could come from conversational speech style, from a worse transcription or from not fully optimal training parameters. Although these results are worse than expected, the PERs are better than published results [48].

7.4.2 SpeechDats-E databases

The phoneme error rates obtained on the OGI Multilanguage Telephone Speech Corpus are high. We wanted to develop a good tokenizer with significantly lower error rate that would significantly improve the language identification. It was obvious that building of system with less than 2 hours of training data could not bring us much improvement, therefore the SpeechDat-E [50] databases were used instead.

The Czech, Hungarian, Polish and Russian databases are used. Table 7.10 shows lengths of the data sets and numbers of phonemes for each language. The SpeechDat databases are not transcribed on phonetic level, therefore a GMM/HMM system was used to produce phoneme transcription at first.

Then the 3-state LC-RC systems were trained. A slightly different approaches than before is used. The phoneme labels were split uniformly into states and 3 iterations of training – realignment are done. These three iterations use smaller nets with 500 neurons. Then the number of neuron was increased to 1500 and the system was retrained. Table 7.11 shows the results. The PER for the Czech recognizer is very competitive to the TIMIT one, even though the SpeechDat data is narrow-band speech.

[2] See relevant TIMIT experiments in section 7.2.1 and 7.1

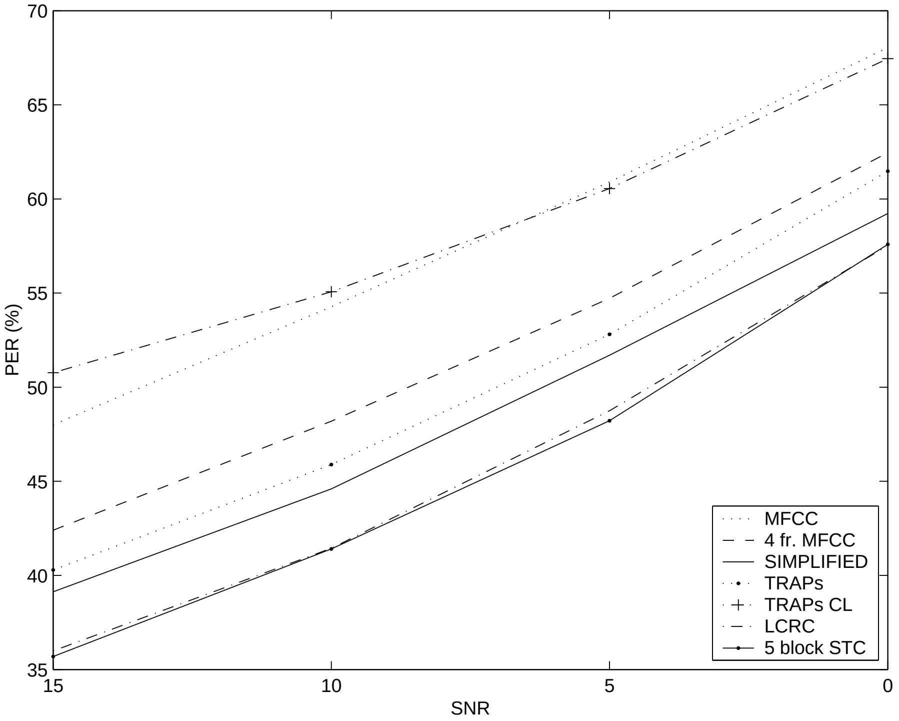

Figure 7.5: *Robustness of 3-state systems against street noise for the same training and test condition.*

The PERs for other language are higher but the phoneme sets are also bigger. A dependency of PER on the geographical size of the country where the database was recorded is also visible. Worse results were obtained for large countries like Poland and Russia.

A larger database gives also an opportunity to repeat the experiment with varying amount of training data. How much can we get from additional 7.5 hours of training data? The Czech LC-RC recognizer was trained repeatedly for different amounts. The neural networks uses just 500 neurons. The results are in Table 7.12 and in Figure 7.9. The PER decreases exponentially with the amount of training data. The most important are the first few hours. Additional 7.5 hours in comparison to TIMIT reduces the PER by 3.5 %. The additional data also allows to train more parameters in the neural networks. For Czech SpeechDat, the optimal number of neurons is 1500 here – it decreases the PER by another 3.2 %.

7.5 Discussion

This chapter demonstrated that the hierarchical structures of neural networks are usable also for another databases and conditions without much tuning. The amount of training data is crucial. The TIMIT results could be easily improved by adding more data.

The robustness against noise and channel change were investigated as well. It is better to train phoneme recognizers on more noisy speech to ensure certain PER. The split temporal context systems work better in noise condition and the TRAP systems work better in cross-channel condition.

Finally, it is questionable why the systems gives a worse PERs on the OGI corpora and this issue should be investigated.

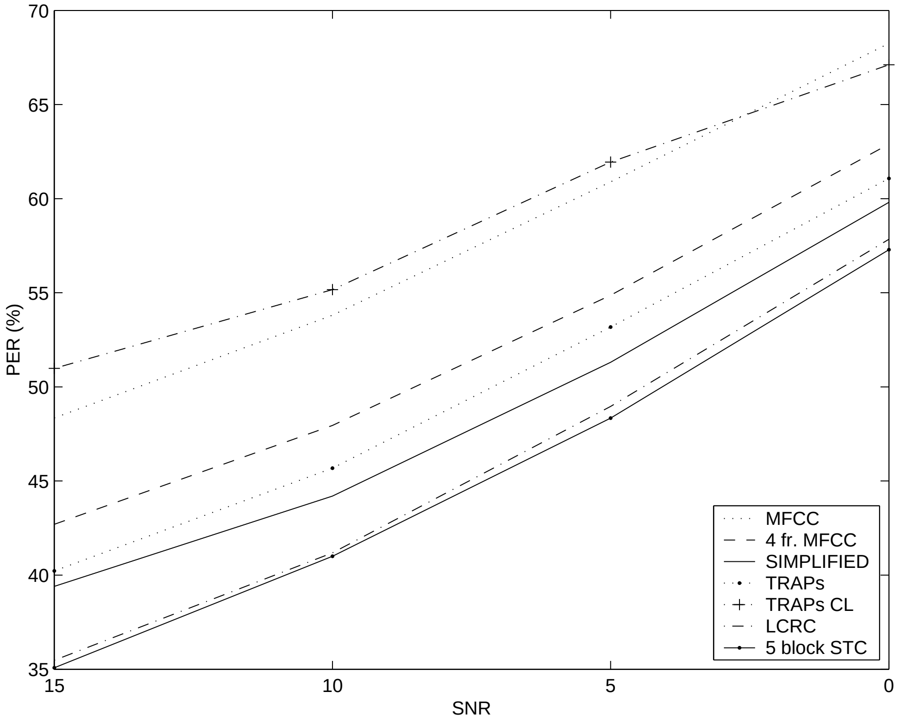

Figure 7.6: *Robustness of 3-state systems against babble noise for the same training and test condition.*

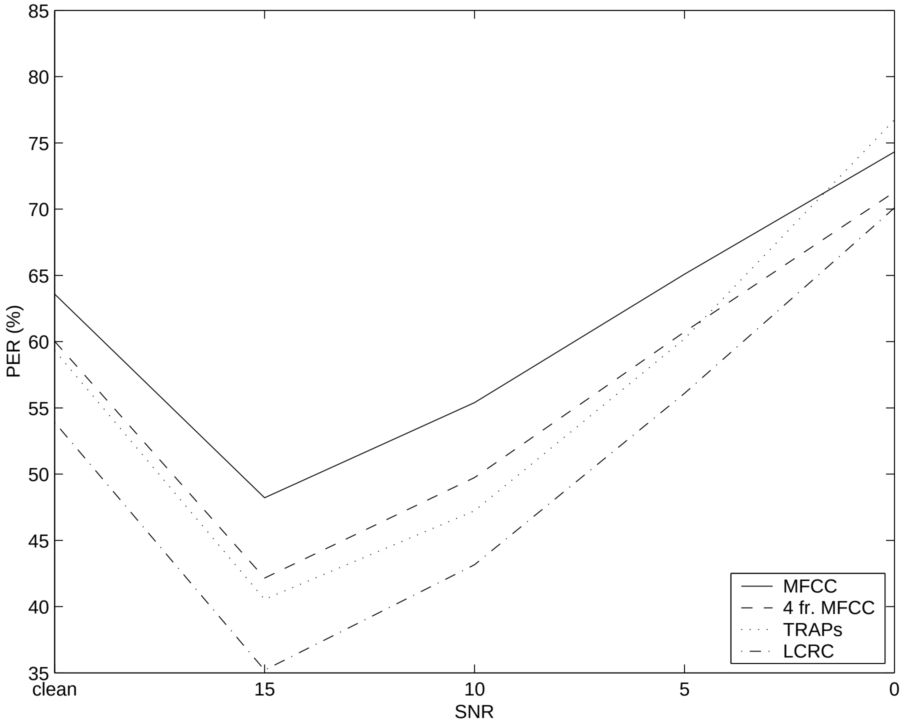

Figure 7.7: *Robustness of 3-state systems against car noise. The system is trained on SNR15 and tested on different SNR levels.*

noise type	clean	car noise				street noise				babble noise			
SNR	-	15	10	5	0	15	10	5	0	15	10	5	0
MFCC39	35.7	48.2	54.0	60.5	68.1	48.0	54.3	60.9	68.0	48.4	53.8	60.9	68.3
MFCC39, 4 frames	32.0	42.2	48.1	54.6	62.4	42.4	48.2	54.7	62.5	42.7	48.0	54.9	62.9
simplified system	30.1	38.9	44.4	51.1	60.1	39.1	44.6	51.7	59.2	39.4	44.2	51.3	59.8
TRAPs	30.0	40.5	45.6	53.0	61.2	40.3	45.9	52.8	61.5	40.2	45.7	53.2	61.1
TRAPs - hard classification	43.9	51.8	55.0	60.2	66.8	50.8	55.1	60.6	67.5	51.0	55.2	61.9	67.1
LC-RC system	27.2	35.7	41.3	48.2	57.7	36.0	41.4	48.8	57.5	35.5	41.2	49.0	57.8
5 block STC	26.8	35.4	41.0	48.5	57.1	35.7	41.4	48.2	57.6	35.1	41.0	48.3	57.3

Table 7.4: *Robustness of 3-state systems against noise for the same training and test condition – car, street and babble noise.*

MFCC39	clean	SNR15	SNR10	SNR5	SNR0
clean	**35.7**	73.2	79.2	85.1	90.0
SNR15	63.6	**48.2**	55.4	65.1	74.3
SNR10	75.5	50.1	**54.0**	62.0	71.2
SNR5	80.3	55.1	55.5	**60.5**	68.8
SNR0	84.8	62.8	61.0	62.3	**68.1**
MFCC39, 4 frames	clean	SNR15	SNR10	SNR5	SNR0
clean	**32.0**	72.0	78.0	83.7	88.5
SNR15	60.0	**42.2**	49.7	60.7	71.3
SNR10	70.5	44.1	**48.1**	56.7	67.5
SNR5	77.6	49.2	49.6	**54.6**	63.7
SNR0	84.1	57.7	55.6	56.7	**62.4**
TRAPs	clean	SNR15	SNR10	SNR5	SNR0
clean	**30.0**	69.5	76.2	83.8	89.2
SNR15	59.3	**40.5**	47.2	60.2	76.8
SNR10	76.2	42.9	**45.6**	54.0	66.7
SNR5	84.8	52.4	49.9	**53.0**	61.9
SNR0	83.5	67.1	59.8	57.7	**61.2**
LC-RC	clean	SNR15	SNR10	SNR5	SNR0
clean	**27.2**	69.6	76.3	82.7	88.3
SNR15	53.9	**35.2**	43.2	56.1	70.1
SNR10	65.1	37.9	**41.3**	50.6	63.8
SNR5	74.7	43.8	43.5	**48.2**	58.7
SNR0	77.2	54.7	51.5	52.1	**57.7**

Table 7.5: *Robustness of 3-state systems against car noise for different training and test SNRs. The training condition is in rows and the test condition is in columns. Equal training and test conditions are emphasized by bold font.*

MFCC39	clean	SNR15	SNR10	SNR5	SNR0
clean	0.0	-37.4	-43.5	-49.4	-54.3
SNR15	-15.4	0.0	-7.2	-16.9	-26.1
SNR10	-21.6	3.9	0.0	-8.0	-17.2
SNR5	-19.7	5.5	5.0	0.0	-8.3
SNR0	-16.7	5.3	7.0	5.8	0.0

MFCC39, 4 frames	clean	SNR15	SNR10	SNR5	SNR0
clean	0.0	-40.0	-45.9	-51.7	-56.5
SNR15	-17.9	0.0	-7.6	-18.6	-29.1
SNR10	-22.4	4.1	0.0	-8.6	-19.4
SNR5	-23.0	5.4	5.1	0.0	-9.1
SNR0	-21.8	4.6	6.8	5.6	0.0

TRAPs	clean	SNR15	SNR10	SNR5	SNR0
clean	0.0	-39.4	-46.2	-53.8	-59.2
SNR15	-18.8	0.0	-6.7	-19.7	-36.2
SNR10	-30.5	2.7	0.0	-8.4	-21.1
SNR5	-31.9	0.6	3.1	0.0	-8.9
SNR0	-22.2	-5.9	1.4	3.5	0.0

LC-RC	clean	SNR15	SNR10	SNR5	SNR0
clean	0.0	-42.4	-49.1	-55.5	-61.1
SNR15	-18.7	0.0	-8.0	-20.9	-34.9
SNR10	-23.8	3.5	0.0	-9.3	-22.5
SNR5	-26.4	4.5	4.7	0.0	-10.4
SNR0	-19.4	3.0	6.3	5.6	0.0

Table 7.6: *Robustness of 3-state systems against car noise for different training and test SNRs. The training condition is in rows and the test condition is in columns. The PER for actual* SNR *is subtracted from PER for* SNR *where the system is trained on.*

condition	TIMIT	NTIMIT	T→N	diff(T→N)	N→T	diff(N→T)
MFCC39	35.7	47.1	70.3	-34.6	49.4	-2.3
MFCC39, 4 frames	32.0	42.1	67.3	-35.2	48.6	-6.5
simplified system	30.1	39.5	65.5	-35.4	45.1	-5.6
TRAPS	30.0	40.8	63.0	-32.9	45.2	-4.4
TRAPS - hard classification	43.9	54.6	65.6	**-21.7**	55.5	bf -1.0
3 band TRAPS	27.8	37.2	**60.8**	-33.0	**41.3**	-4.2
LC-RC system	27.2	36.2	63.7	-36.5	42.3	-6.2
STC, 5 blocks	26.8	36.2	65.1	-38.4	42.2	-6.0

Table 7.7: *Robustness of 3-state systems against channel change.* diff$(T \to N) = PER(T \to N) - PER(T)$ *and* diff$(N \to T) = PER(N \to T) - PER(N)$.

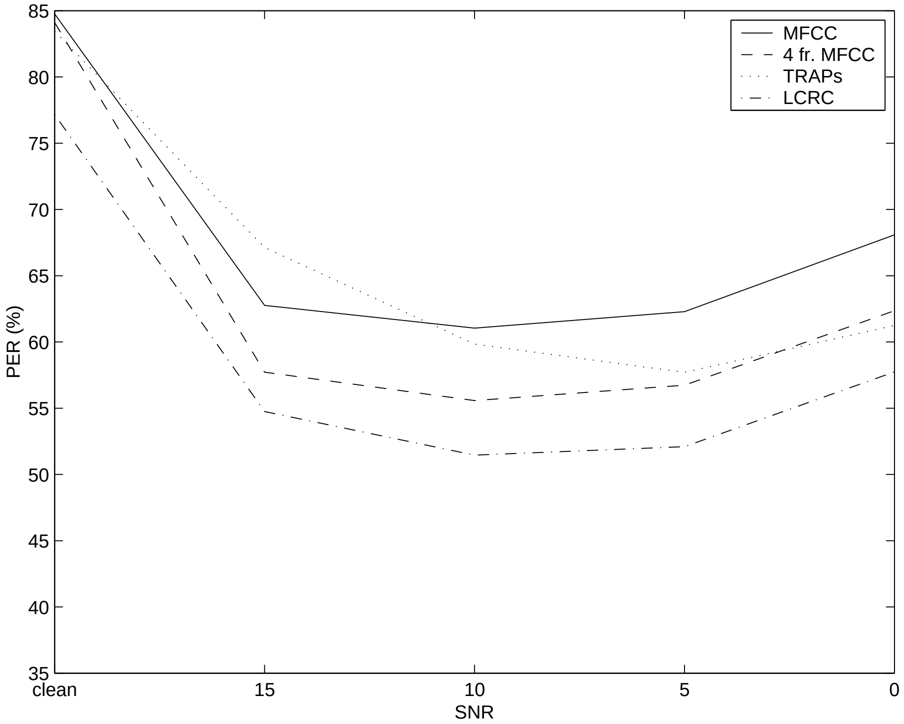

Figure 7.8: *Robustness of 3-state systems against car noise. The system is trained on SNR0 and tested on different SNR levels.*

language	ENG	GER	HIN	JAP	MAN	SPA
train (hours)	1.71	0.97	0.71	0.65	0.43	1.10
cv (hours)	0.16	0.10	0.07	0.06	0.03	0.10
test (hours)	0.42	0.24	0.17	0.15	0.11	0.26
# phonemes	40	44	47	30	45	39

Table 7.8: *Amounts of speech data used to train phoneme recognizers on the OGI Multilanguage corpus.*

language	ENG	GER	HIN	JAP	MAN	SPA
PER (%)	45.3	46.1	45.7	41.2	49.9	39.6

Table 7.9: *Phoneme error rates of 3-state LC-RC system on the OGI Multilanguage corpus.*

language	CZE	HUN	POL	RUS
train (hours)	9.72	7.86	9.49	14.02
cv (hours)	0.91	0.77	0.88	1.57
test (hours)	2.26	1.97	2.34	3.89
# phonemes (-)	46	62	41	53

Table 7.10: *Amounts of speech data used to train phoneme recognizers on SpeechDat-E corpora.*

language	CZE	HUN	POL	RUS
PER (%)	24.2	33.4	36.3	39.3

Table 7.11: *Phoneme error rate of 3-state LC-RC system on SpeechDat-E corpora.*

length (hours)	1	3	5	7	10
PER (%)	34.9	30.8	29.6	28.3	**27.4**

Table 7.12: *Dependency of PER on the length of training set for the 3-state Czech LC-RC system.*

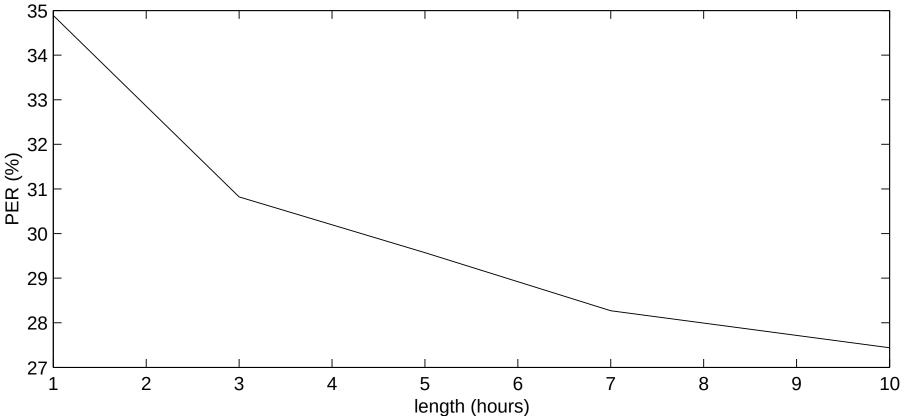

Figure 7.9: *Dependency of PER on the length of training set for the 3-state Czech LC-RC system.*

Chapter 8

Applications

The phoneme recognition has a huge potential in wide range of applications. A package of the phoneme recognizer with networks trained for TIMIT English, SpeechDat Czech, SpeechDat Hungarian and SpeechDat Russian is available at the web-pages of our group[1]. This chapter gives a short introduction to some of the applications, where the phoneme recognition is currently used. Only basic techniques and baselines are presented and full information can be found in our publications available from the web[2].

8.1 Language identification

The language identification system is usually composed from two parts – phonotactic language identification and acoustic language identification [51]. The phonotactic identification models the order of different sounds in language. The acoustic identification models how the language sounds like. The phoneme recognizer is the main part of the phonotactic language identification system: it produces phoneme strings or lattices, and the following language model[3] models the order of phonemes. This architecture is called Phoneme Recognizer followed by Language Model (PRLM). The phoneme recognizer does not necessarily have to be trained for the language that should be identified, but it should cover as many phonemes from the language as possible. The accuracy of phoneme recognizer is crucial for good identification [52]. The accuracy here does not necessarily mean the lowest PER, but the most consistent string under all conditions. If a phoneme is always confused with another one, it is modelled in the language model that way and it is fine for language identification.

8.1.1 Language modelling

The most common language model used for language identification is classical 3-gram LM, where conditional probabilities of a word given two preceding words $P(w_i|w_{i-2}, w_{i-1})$ are estimated.

During identification, a probability that the sentence was generated by the model is evaluated:

$$P_L = \prod_{i=1}^{N} P(w_i|w_{i-2}, w_{i-1}), \tag{8.1}$$

where N is number of phonemes in the phoneme string.

[1] http://speech.fit.vutbr.cz/en/software/phoneme-recognizer-based-long-temporal-context

[2] http://speech.fit.vutbr.cz/node/22

[3] Also called phonotactic model

One model is trained for each language. For classification, simply the model with maximal P_L is found and the corresponding language is winning.

$$L = \operatorname*{argmax}_{i=1}^{M} P_{Li}, \tag{8.2}$$

where M is the number of models (languages). There is still a question of n-grams unseen during the training where we do not have the probability $P(w_i|w_{i-2}, w_{i-1})$. Usually, no smoothing of the model is done[4]. The n-grams are simply not used for evaluation in all language models.

8.1.2 Score normalization

In case the software is used as a detector of language, another post-processing (normalization) of scores must be done. One possibility is to use posterior probabilities of languages. Equation 8.2 is usually implemented in logarithmic domain. It can be rewritten to:

$$log(P_L) = \sum_{i=1}^{N} log(P(w_i|w_{i-2}, w_{i-1})) \tag{8.3}$$

For normalization, the log-probabilities are divided by numbers of phonemes. This ensures constant sharpness[5] of the posterior probabilities of languages with varying length of the record. Then the value is multiplied by a scaling factor. This factor allows the user to change the sharpness and adapt the scores to the task. Finally, the values are re-normalized to probabilities. These three steps are summarized by:

$$log(P_L)' = \frac{c}{N} log(P_{Li}) - \operatorname*{logadd}_{j=1}^{M} \frac{c}{N} log(P_{Lj}), \tag{8.4}$$

where N is number of phonemes in phoneme string, M is number of competing languages and c is the scaling factor.

$$logadd(a, b) = log(e^a + e^b) \tag{8.5}$$

A fixed threshold is used for detection. If $log(P_L)'$ is higher than a threshold, a detection is made.

The detector can be evaluated by Equal Error Rate (EER) – the threshold is set in such way, that the number of miss-detections and false alarms equals. The EER is the number of miss-detections or the number of false alarms.

8.1.3 Speech@FIT phonotactic language identification systems

All the phoneme recognizers have the LC-RC structure. The results are reported on the 30 s condition from NIST 2003 language identification evaluation[6]. The language models are trained on the CallFriend databases [53]. The LID system is shown in Figure 8.1. The indicated background language model is another possibility to show the score normalization.

[4] Classical smoothing techniques for word recognition can not be used directly because the set of phonemes is small and closed.

[5] The distribution is sharp in case the winning language has probability 1 and the others have probability 0. In the opposite case, all probabilities are equal. There is direct link to the entropy.

[6] See `http://www.nist.gov/speech/tests/lre/` for evaluation plan.

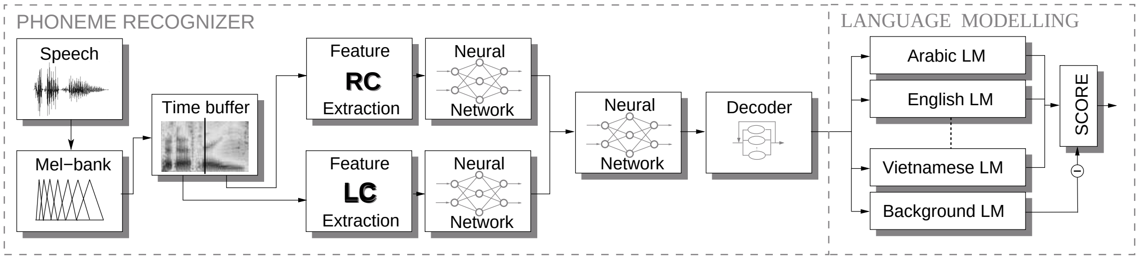

Figure 8.1: *Phonotactic language identification system.*

8.1.4 LID systems based on phoneme recognizers trained on the OGI data

At first, the LID systems were built using phoneme recognizers trained on the OGI Multilingual corpora [54]. The recognizers are described in section 7.4.1. Just one phoneme recognizer was used each time as a tokenizer.

The results are in Table 8.1, line *baseline*. The results are poor, therefore the decision to retrain all the phoneme recognizer using all available phonetically transcribed data from the OGI Multilingual corpora (+ test set, + the cross-validation set) was done. We lost the possibility to evaluate the recognizers using PER. The results can be found in Table 8.1, line *retrained*. The results show clearly the improvement thanks to more data. The table also presents a linear fusion of all the LID systems. Such system is known as Parallel Phoneme Recognition followed by Language Model (PPRLM). The posterior probability vectors (Equation 8.4) are weighted and summed together. One weight is found for each system using using the simplex method on the NIST 1996 evaluation data.

Language	ENG	GER	HIN	JAP	MAN	SPA	fusion
baseline	11.83	11.67	9.75	11.42	15.08	14.08	6.92
retrained	10.58	10.33	8.92	9.08	12.83	11.33	5.58

Table 8.1: *Language identification results (EER) of single PRLMs trained on the OGI Multilanguage database and tested on the 30 second task from the NIST 2003 LRE evaluation.*

8.1.5 LID systems based on phoneme recognizers trained on SpeechDat-E data

The SpeechDat-E phoneme recognizers were used exactly in the same way as described in previous section. The recognizers are described in section 7.4.2 and LID results are in Table 8.2. As can be seen, the EER is significantly better with the SpeechDat-E phoneme recognizers.

The influence of amount of training data on EER was studied and the results are reported in Table 8.3. The EER decreases exponentially, similarly to PER. Using more data would be still beneficial.

Language	CZE	HUN	POL	RUS	fusion
EER (%)	5.42	4.42	6.75	4.75	2.42

Table 8.2: *Language identification results (EER) of single PRLMs trained on the SpeechDat-E database and tested on the 30 second task from the NIST 2003 LRE evaluation.*

training data (hours)	PER (%)	EER 30 sec (%)	EER 10 sec (%)	EER 1 sec (%)
1	34.9	9.17	18.08	28.92
3	30.8	6.50	15.75	27.00
5	29.6	5.67	15.17	26.42
7	28.3	5.42	14.25	26.83
10	27.4	5.42	14.17	26.00

Table 8.3: *Effect of amount of training data for Czech SpeechDat-E phoneme recognizer on PER and language identification EER (NIST 2003 LRE task).*

8.1.6 Discussion

The developed phoneme recognizers were used with success for the language identification task. The consistency of phoneme strings generated by the recognizers was verified to be very good and the recognizers can be safely used even for transcription of unseen languages. The purpose of this chapter is not to show the state-of-the-art systems, but rather to demonstrate suitability of developed techniques for language identification. There is a lot of follow-up work done in our lab by Pavel Matějka [55] that extends phoneme strings to phoneme lattices and introduces phonotactic anti-models [56]. There is also a work on Decision Trees and factor analysis done by Ondřej Glembek [57]. The phoneme strings can be also modelled using Support Vector Machines. All these techniques together with a discriminatively trained acoustic part give the state-of-the-art performance that brough our group among top-scoring sides in NIST 2005 and 2007 LRE evaluation.

8.2 Large Vocabulary Conversational Speech Recognition

Although the structure of current state-of-the-art LVCSR systems is very complex (speaker adaptive training, different kinds of adaptations, discriminative training), the techniques described in this thesis can help even here. We did not reached the state-of-the-art performance with the hybrid HMM/ANN system, but rather the TANDEM architecture introduced in [58] became very popular. The TANDEM architecture uses phoneme (or state) posterior vectors as features in a classical HMM/GMM system. Gaussianization, decorrelation and dimensionality reduction of posterior features is necessary:

- The Gaussianization of features makes binomial distribution with two sharp peaks in 1 and 0 more gaussian. The logarithm can be used for this purpose. Another choice is to remove the final nonlinearity in the last (merger) neural network.
- The decorrelation and dimensionality reduction can be done using Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) or Heteroscedastic Linear Discriminant Analysis (HLDA).

Although these features give better results in comparison to classical MFCC or PLP features, even better result can be obtained using concatenation of the classical and these novel features. Such system is shown in Figure 8.2[7]. These features were used for example in the AMI system

[7] Thanks František Grézl for this figure.

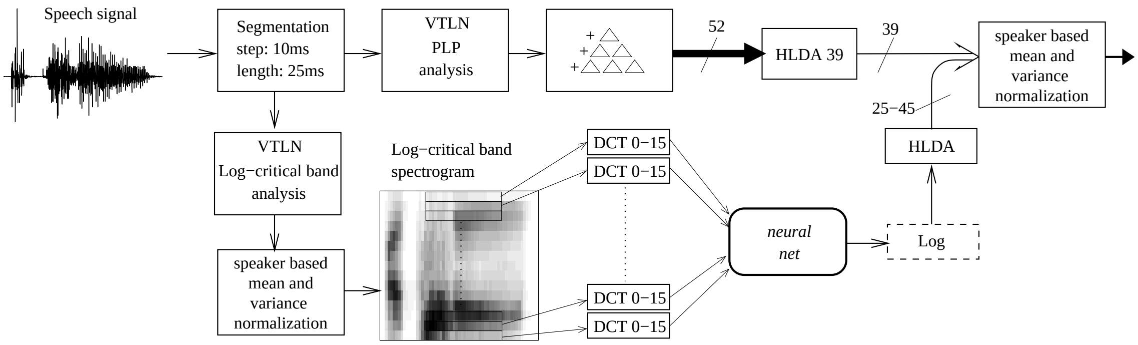

Figure 8.2: *Incorporation of long temporal context based features to a state-of-the-art LVCSR system.*

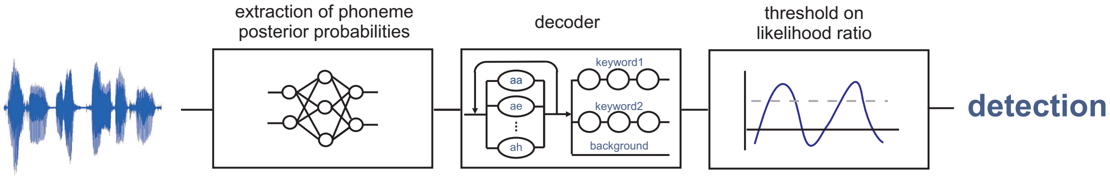

Figure 8.3: *Keyword spotting system.*

for transcription of meetings [8] and improved the system about 11.5 % relative in word error rate (WER). There is a follow up work done by František Grézl [42] investigating bottle-neck features.

8.3 Keyword spotting

Neural network based techniques can be very fast. This property makes them good candidates for a fast on-line keyword spotting. This work was done together with Igor Szöke [59]. A scheme of such keyword spotting system can be seen in Figure 8.3.

The same front-end as for phoneme recognition is used (parametrization and the neural network structure), but the decoder is different. It is necessary to model the keyword by an HMM. There are parts of speech before and after the keyword that can carry useful information for spotting and need to be modelled too. For example, the keyword will follow some words more probably than other words. A likelihood of generation of a speech segment by such composite model can be evaluated during spotting. But the likelihood can differ for each speaker, microphone and pronunciation of the keyword. Therefore we need an universal model of speech (background model) to compare the likelihood with. The likelihood ratio between the composite model and the background model is used a score (the schema of keyword spotting decoder is outlined in Figure 8.4):

$$S = \frac{L_{front\_filler} L_{keyword} L_{back\_filler}}{L_{background}} \tag{8.6}$$

The score is then compared with a threshold and if it is higher than a threshold, a detection of keyword is made. In our case, the part front of the keyword is modelled by a phoneme loop, the part behind the keyword is not modelled at all, and the background model is again a phoneme loop.

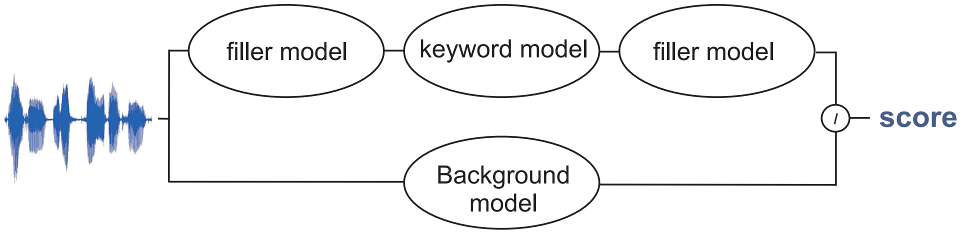

Figure 8.4: *Keyword spotting decoder.*

Still, many detections for one keyword, just slightly shifted can be generated. Therefore, the time trace of score[8] was further post-processed. The maximas were found and:

1. If the maximum is higher than the previous one and it is occurrence of the same keyword occurrence, the previous maximum is removed.

2. If the maximum is lower than the previous one and it is occurrence of the same keyword occurrence, the actual maximum is removed.

Whether it is the same keyword occurrence or not can be decided from the keyword length. Some partially overlapped keyword occurrences are marked as one occurrence.

8.3.1 Evaluation

The keyword spotting systems was tested on a large database of informal continuous speech of ICSI meetings [60]. Attention was paid to the definition of fair division of data into training/development/test parts with non-overlapping speakers. It was actually necessary to work on speaker turns rather than whole meetings, as they contain many overlapping speakers. We have balanced the ratio of native/nonnative speakers, ratio of European/Asiatic speakers and moved speakers with small portion of speech or keywords to the training set. The amounts of data in the training, development and test parts are 41.3 h, 18.7 h and 17.2 h respectively.

The development part was used for system tuning (phoneme insertion penalty, etc.). In the definition of keyword set, we have selected the most frequently occurring words (each of them has more than occurrences in each of the sets) but checked, that the phonetic form of a keyword is not a subset of another word nor of word transition. The percentage of such cases was evaluated for all candidates and words with high number of such cases were removed. The final list consists of 17 keywords: **`actually`**, **`different`**, **`doing`**, **`first`**, **`interesting`**, **`little`**, **`meeting`**, **`people`**, **`probably`**, **`problem`**, **`question`**, **`something`**, **`stuff`**, **`system`**, **`talking`**, **`those`**, **`using`**.

Our experiments are evaluated using Figure-of-Merit (FOM), which is the average of correct detections per 1, 2, $\cdots$ 10 false alarms per hour. Obviously, in real scenarios, more specific words than doing, probably, etc. will be used. For statistical evaluation using FOM, we however need a set of keywords with many occurrences in the data.

Three systems were used for comparison:

1. ICSI10 – a cross-word context dependent HMM/GMM system trained on 10 hours of ICSI data

[8] score for individual frames

2. CTS277 – a more advanced HMM/GMM system trained on 277 hours of conversational telephone data and MAP-adapted to the meeting domain on the training part of the ICSI database.
3. LCRC41 – a neural network based (LC-RC) keyword spotting system trained on the training part of the ICSI database.

The results can be seen in Table 8.4. Even through the neural network based system is very simple and the network uses monophones only, the system can reach results similar (or slightly better) as context dependent cross-word HMM/GMM system.

System	FOM
ICSI10	61.88
CTS277	63.66
LCRC41	**64.46**

Table 8.4: *Comparison of different keyword spotting systems.*

8.3.2 Discussion

The reason of using the neural network based approaches for keyword spotting is not to reach the best spotting accuracy. In this case more precise discriminatively trained models and adaptation techniques should be be evaluated. The main reason is the speed. The spotting accuracy is comparable with classical techniques and the experimental system runs 0.15×RT on one-core machine (Intel P4, 2GHz). Recent experiments showed that the system can be speed-up at least twice without degradation in spotting accuracy. Since it is common to have multi-core processors, 0.02 RT can be easily reached with 4 cores. These techniques very are promissive for processing of large speech archives and for search engines working with spoken speech.

8.4 Voice Activity Detection

The Voice Activity Detection (VAD) is very important application of phoneme recognition techniques. A phoneme recognizer is used to transcribe speech to phoneme strings. Then the phonemes (and noise marks) are mapped to two classes – speech and silence. The neighboring segments with the same labels are mapped together. This long temporal context based segmentation is very reliable. It can be used directly or post-processed using an energy function, or information form other channels in case of multichannel speech records. This VAD was successfully used for preprocessing of speech records for almost all our submissions to NIST evaluations (Language Identification, Speaker Recognition, LVCSR, Spoken Term Detection), and in many applications.

8.5 Discussion

The range of applications and research branches described here is wide. It proved the usefulness of the research. Sometimes looking at the technology trough the applications can bring more insight into it. The requests and optimization criteria differ. It is good to have always in mind that the best PER is not only the one criterion and that a good understanding of the whole technology is necessary.

Chapter 9

Conclusions

This work showed that it is possible to develop highly accurate phoneme recognizers on very low amount of training data. The accuracy comes from modelling of long temporal contexts for phonemes (few hundreds of milliseconds). The difficulty is the design of models for such large phoneme patterns. This thesis describes many techniques that allow to train neural networks for this purpose. The most important one is incorporation of some constraints coming from the task to the neural network structure. The possibility that the training algorithm will get stuck in local extreme is reduced. A hierarchical structure of neural networks was proposed for this purpose. The other techniques are dimensionality reduction of input patterns, windowing of the patterns or a finer representation of neural network outputs. Such designed phoneme recognizer with the split temporal context was integrated to a software package and it is now publically available on our web page[1].

A reviewer of one of my articles argued that ”the TIMIT was beaten to dead by this work”. It is impossible to study new promising techniques without coming to their limits and without having well trained classifiers. Although the phoneme error rate is already low (21.48 %), it is definitely not the final number, and even not for this unadapted system. Different normalization techniques, better language model, duration modelling or other complementary features can be applied. Then the system can by improved by speaker adaptation, speaker adaptive training, channel compensation and other techniques.

All the reported results here are phoneme recognition error rates. But a lower phoneme recognition error rate does not automatically mean lower word recognition error rate. It is always necessary to verify the advantage of new techniques on the final task. The relation between phoneme error rate, word error rate and language models will be studied in my future work.

[1] `http://speech.fit.vutbr.cz/en/software/phoneme-recognizer-based-long-temporal-context`

Bibliography

[1] S. B. Davis and P. Mermelstein, "Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.

[2] H. Hermansky, "Perceptual linear predictive (plp) analysis of speech," *Jurnal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[3] Brian C.J. Moore, *An introduction to the psychology of hearing*, Academic press, Boston, USA, 1997.

[4] J. A. Bilmes, "Maximum mutual information based reduction strategies for cross-correlation based joint distribution modelling," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Seattle, USA, May 1998.

[5] H. H. Yang, S. Sharma, S. van Vuuren, and H. Hermansky, "Relevance of time-frequency features for phonetic and speaker-channel classification," *Speech Communication*, vol. 31, no. 1, pp. 35–50, Aug 2000.

[6] S. Kajarekar, *Analysis of variability in Speech with Applications to Speech and Speaker Recognition*, Ph.D. thesis, OGI, Portland, USA, Jul 2002.

[7] N. Kanebara, T. Arai, H. Hermansky, and M. Pavel, "On the importance of various modulation frequencies for speech recognition," in *Proc. Eurospeech*, Rhodes, Greece, Sep 1997.

[8] T. Hain, V. Wan, L. Burget, M. Karafiat, J. Dines, J. Vepa, G. Garau, and M. Lincoln, "The ami system for the transcription of speech in meetings," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, Hononulu, US, 2007, pp. 357–360, IEEE Signal Processing Society.

[9] P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, and J.R. Deller Jr., "Approaches to language identification using gaussian mixture models and shifted delta cepstral features," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sept. 2002, pp. 89–92.

[10] G. Saon, G. Zweig, B. Kingsbury, L. Mangu, and U. Chaudhari, "An architecture for rapid decoding of large vocabulary conversational speech," in *Proc. Eurospeech*, Geneva, Switzerland, Sept. 2003.

[11] R. Duda and P. Hart, *Pattern Classication and Scene Analysis*, John Wiley & Sons, New York, 1973.

[12] K. Fukunaga, *Introduction to statistical pattern recognition*, Academic press, Boston, USA, 1990.

[13] L. Burget, *Complementarity of Speech Recognition Systems and System Combination*, Ph.D. thesis, Brno University of Technology, Brno, Czech Republic, 2004.

[14] N. Kumar and A. G. Andreou, "Heteroscedastic discriminant analysis and reduced rank hmms for improved speech recognition," *Speech Communication*, , no. 26, pp. 283–297, 1998.

[15] N. Kumar, *Investigation of Silicon-Auditory Models and Generalization of Linear Discriminant Analysis for Improved Speech Recognition*, Ph.D. thesis, John Hopkins University, Baltimore, USA, 1997.

[16] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, Feb. 1989.

[17] S. Young et al., *The HTK Book*, Cambridge University, Cambridge University Engineering Department, 2005.

[18] H. Bourlard and N. Morgan, *Connectionist speech recognition: A hybrid approach.*, Kluwer Academic Publishers, Boston, USA, 1994.

[19] T. Robinson, "An application of recurrent nets to phone probability estimation," *IEEE Transactions on Neural Networks*, vol. 5, no. 3, 1994.

[20] T. Robinson, M. Hochberg, and S. Renals, "The use of recurrent neural networks in continuous speech recognition," *Inet*, 1994.

[21] H. Hermansky and S. Sharma, "Temporal patterns (traps) in asr of noisy speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Phoenix, Arizona, USA, Mar 1999.

[22] B. Chen, *Learning Discriminant Narrow-Band Temporal Patterns in Automatic Recognition of Conversational Telephone Speech*, Ph.D. thesis, University of California, Berkeley, Berkeley, CA, USA, 2005.

[23] P. Jain and H. Hermansky, "Beyond a single critical-band in trap based asr," in *Proc. Eurospeech*, Geneva, Switzerland, Sep 2003.

[24] F. Grezl, *TRAP-based Probabilistic Features for Automatic Speech Recognition*, Ph.D. thesis, Brno University of Technology, 2007.

[25] B. Chen, Q. Zhu, , and N. Morgan, "Learning long-term temporal features in LVCSR using neural networks," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Jeju Island, South Korea, Oct 2004.

[26] K. Lee and H. Hon, "Speaker-independent phone recognition using hidden markov models," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 37, no. 11, pp. 1641–1648, Nov. 1989.

[27] "TIMIT Acoustic-Phonetic Continuous Speech Corpus," http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S1.

[28] S. J. Young, "The general use of tying in phoneme-based hmm speech recognizers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, San Francisco, USA, Mar 1992.

[29] V. V. Digalakis, M. Ostendorf, and J. R. Rohlicek, “Fast algorithms for phone classification andrecognition using segment-based models,” in *IEEE Transactions on Signal Processing*, Dec 1992, vol. 40, pp. 2885–2896.

[30] D. J. Pepper and M.A. Clements, “Phonemic recognition using a large hidden markov model,” *Signal Processing*, vol. 40, no. 6, pp. 1590 – 1595, Jun 1992.

[31] L. Lamel and J. Gauvian, “High performance speaker-independent phone recognition using cdhmm,” in *Proc. European Conf. Speech Communication and Technology*, 1993, pp. 121–124.

[32] S. Kapadia, V. Valtchev, and S. J. Young, “Mmi training for continuous phoneme recognition on the timit database,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Minneapolis, USA, Apr 1993.

[33] A. K. Halberstadt, *Heterogeneous Acoustic Measurements and Multiple Classifiers for Speech Recognition*, Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1998.

[34] J. W. Chang, *Near-Miss modelling: A Segment Based Approach to speech Recognition*, Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1998.

[35] B. Chen, S. Chang, and S. Sivadas, “Lerning discriminative temporal patterns in speech: Development of novel traps-like classifiers,” in *Proc. Eurospeech*, Geneva, Switzerland, Sep 2003.

[36] J. Moris and E. Fosler-Lussier, “Combining phonetic attributes using conditional random fields,” in *Proc. Interspeech*, Pittsburg, USA, Sep 2006, pp. 597 – 600.

[37] F. Sha and L. Saul, “Large margin gaussian mixture modelling for phonetic classification and recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, pp. 265 – 268.

[38] L. Deng and D. Yu, “Use of differential cepstra as acoustic features in hidden trajectory modelling for phonetic recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawai’i, USA, Apr 2007.

[39] J. Černocky P. Schwarz, P. Matějka, “Recognition of phoneme strings using trap technique,” in *Proc. Interspeech*, Geneve, Switzerland, 2003, pp. 1–4.

[40] Q. Zhu, B. Chen, N. Morgan, and A. Stolcke, “On using mlp features in lvcsr,” in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Jeju Island, South Korea, Oct 2004.

[41] H. Hermansky and N. Morgan, “Rasta processing of speech,” *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 4, pp. 578–589, Oct 1984.

[42] F. Grezl, “Combinations of trap based systems,” in *Proc. Seventh International conference on Text, Speech and Dialogue*, Brno, Czech Republic, 2004, pp. 323–330, Faculty of Informatics MU.

[43] S. van Vuuren and H. Hermansky, "On the importance of components of modulation spectrum for speaker verification," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Australia, Sydney, 1998, vol. 2.

[44] H. Hermansky and P. Jain, "Down-sampling speech representation in asr," in *Proc. Eurospeech*, Hungary, Budapest, Sep 1999, vol. 2.

[45] G. Evermann, H.Y. Chan, M.J.F. Gales, B. Jia, X. Liu, K.C. Sim D. Mrva, P.C. Woodland L. Wang, and K. Yu, "Development of the 2004 cu-htk eenglish cts systems using more than two thousand hours of data," in *Proceedings of Fall 2004 Rich Transcription Workshop*, Newyork, Nov. 2004.

[46] J. Stadermann, W. Koska, and G. Rigoll, "Multi-task learning strategies for a recurrent neural net in a hybrid tied-posteriors acoustic model," in *Proc. Eurospeech*, Lisbon, Portugal, Sept. 2005.

[47] "NTIMIT Acoustic-Phonetic Continuous Speech Corpus,"
http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S2.

[48] Y. Yan and E. Barnard, "An approach to automatic language identification based on language-dependent phoneme recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Detroit, USA, May 1995.

[49] J. Kohler, "Multi-lingual phoneme recognition exploiting acoustic-phonetic similarities of sounds," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Philadelphia, USA, Oct 1996.

[50] "SpeechDat-East project," http://www.fee.vutbr.cz/SPEECHDAT-E.

[51] P. Matějka, L. Burget, P. Schwarz, and J. Černocký, "Brno University of Technology System for NIST 2005 Language Recognition Evaluation," in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006, pp. 57–64.

[52] P. Matějka, P. Schwarz, J. Černocký, and Pavel Chytil, "Phonotactic language identification using high quality phoneme recognition," in *Proc. Eurospeech*, 2005, pp. 2237–2240.

[53] "CallFriend Corpus, telephone speech of 15 different languages or dialects,"
www.ldc.upenn.edu/Catalog/byType.jsp#speech.telephone.

[54] "OGI Multilanguage Corpus,"
http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC94S17.

[55] P. Matějka, *Language identification*, Ph.D. thesis, Brno University of Technology, Brno, Czech Republic, 2008.

[56] P. Matějka, P. Schwarz, B. Burget, and J. Černocký, "Use of Anti-Models to Further Improve State-of-the-art PRLM Language Recognition System," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, pp. 197–200.

[57] P. Matějka, L. Burget, O. Glembek, P. Schwarz, V. Hubeika, M. Fapšo, T. Mikolov, O. Plchot, and J. Černocký, "BUT language recognition system for NIST 2007 evaluations," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Brisbane, Australia, Sept. 2008.

[58] H. Hermansky, D. Ellis, and S. Sharma, “Tandem connectionist feature extraction for conventional hmm systems,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Istanbul, Turkey, June 2000.

[59] I. Sz oke, P. Schwarz, L. Burget, M. Fapšo, M. Karafiát, J. Černocký, and P. Matějka, “Comparison of keyword spotting approaches for informal continuous speech,” in *Proc. Eurospeech*, Lisabon, Portugal, 2005, pp. 633–636.

[60] “ICSI Meeting Speech,”
http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004S02.