NTNU

Norwegian University of
Science and Technology

Cochlear Features for Acoustic Segmentation

Jarle Bauck Hamar

Master of Science in Communication Technology
Submission date: June 2009
Supervisor: Torbjørn Svendsen, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

Automatic speech recognition (ASR) is currently mainly formulated as a statistical pattern recognition problem. In this formulation, hidden Markov models (HMM) has been the dominant approach for acoustic modeling. The parameters of the HMM are estimated blindly from large datasets of representative speech. This paradigm is commonly known as ignorance modeling. The paradigm has been developed over several decades, with gradual improvements, but does now appear to have reached a level where significant improvements are unattainable. Compared to human capacity for speech recognition, the machines still have error rates that are at least an order of magnitude higher.

Different speech sounds have varying duration and internal dynamics. In traditional ASR the speech signal is analyzed based on short-time spectral analysis, where the type of analysis and the analysis window is constant, independent of the duration and characteristics of the sounds. In this study, we will base the work on an auditory model for signal representation. Specifically, a model that simulates the manner in which the ear transforms a sound pressure wave to impulses exciting the auditory nerve, i.e. a cochlear model is to be studied.

This signal representation is to be studied in order to assess whether it can serve as the basis for a segmentation algorithm that can identify segments of the speech signal where the speech characteristics exhibit little variation over short time intervals. The segmentation process will be data driven, and result in segments that can be regarded as acoustic subwords that in turn can serve as basis for a signal dependent analysis of the information content of the speech signal.

Relevant research questions in the work can be: what are the properties of the auditive signal representation? Can segmentation based on an auditive model identify the significant acoustic changes in the signal? How must the auditive representation be adapted in order to achieve a segmentation that to a large extent will detect phoneme transitions? Will a segmentation based on an auditive representation exhibit significant differences compared to segmentation based on a traditional signal representation such as mel-frequency cepstral coefficients?

Assignment given: 09. February 2009
Supervisor: Torbjørn Svendsen, IET

Abstract

This work explores an alternative set of features to the frequently used mel-frequency coefficients (MFCCs). The cochlea features simulate the nerve fibre signal sent from the ear to the brain. In this study the usage of the cochlea features for acoustic segmentation is of main interest. Both the cochlea features and a variant of combining them with zero crossing with peak amplitude (ZCPA) have been used as input to an acoustic segmentation algorithm. Also experiments using the cochlea features as input to an artificial neural network (ANN) for classifying each vector as boundary/non-boundary have been performed. The results show that the features contain a great deal of information regarding the speech signal. Especially the combination of cochlea and ZCPA are giving good results.

Preface

This project is performed as the final part of a Master degree in Communication Technology at the University of Science and Technology, NTNU. The work is also a part of the degree of Philosophiae Doctor (PhD) at the Department of Electronics and Telecommunications, IET at NTNU. The PhD degree is part of SIRKUS, a research project funded by The Research Council of Norway.

Acknowledgements

The project is supervised by professor Torbjørn Svendsen, who has given valuable directions on how the experiments should be performed, and interpretations of the results. I would also like to thank Alfonso Martinez del Hoyo Canterla for letting me use his MLP software.

Contents

List of Abbreviations

ANN Artificial Neural Network

ASR Automatic Speech Recognition

HMM Hidden Markov Model

MFCC Mel-Frequency Cepstral Coefficients

MHC Meddis Hair Cell

MLP Multi-Layer Perceptron

ZCPA Zero Crossings with Peak Amplitude

1 Introduction

1.1 Overview

Today computers have a huge impact on daily life. They are basically everywhere and used by nearly everyone. A big problem is that the interaction with computers in some cases is ineffective and not available to everyone. Humans use speech to communicate with each other, a way of communication that is natural and efficient. The ability to communicate with computer systems in the same way would be very beneficial. Talking to a computer would make nearly all humans able to use them and in addition give a highly effective way of interaction.

Despite years of research the current state of the art speech recogniser performs at best one order of magnitude below human performance. Since a typical human is usually uncomfortable when speaking to a machine, too many errors made by the machine will quickly result in the person giving up. Hence, good performance is crucial for the usability of the entire system. Therefore the need for a better large vocabulary recogniser system for spontaneous speech is immediate.

An automatic speech recognition (ASR) system consists of several parts. This project will concentrate on the first part of the system: the front end. This is the part of the system that employs signal processing to extract all necessary information for the task of discriminating sounds, words and utterances. Any weaknesses in this part will therefore affect all the subsequent parts of the system.

The standard ASR front end extracts a sequence of frequency properties (features) from the signal using a short time spectral analysis. This approach has some weaknesses:

1. The short time spectral analysis is performed with a fixed window length, independent of the underlying signal. To be able to perform a spectral analysis, the signal is assumed to be quasi-stationary within this window. This is not correct if the window contains an abrupt change in the signal, e.g the transition between closure and burst of plosives.

2. Normally the features are used as input to a hidden Markov model (HMM), which calculates the most probable class (phoneme, sub word or similar). When using a HMM it is assumed that succeeding feature

vectors are statistical independent. An assumption that is obviously not correct.

3. The same short time spectral analysis is performed regardless of the properties of the segment under consideration. This is clearly not optimal, as a segment with small changes in the frequency spectrum may be analysed using a larger window, resulting in a better approximation.

By introducing a segmentation procedure which automatically divides the signal into segments, these weaknesses may be improved. Firstly, information about the stationarity of the signal can be exploited to choose the window size. Secondly, by using one feature vector for each segment, the statistical dependency can be reduced. Thirdly, any additional information gained in the segmentation process might be employed to decide the size of the analysis window.

Many approaches to speech segmentation have been proposed, see e.g. [1, 2, 3, 4, 5, 6, 7]. In this project the algorithm described in [4] and [5] will be used.

The use of the mel-frequency cepstral coefficients (MFCCs) in speech technology is extensive. However, since the performance of speech recognisers using MFCCs seems to have reached a limit, an alternative is needed. In this project cochlea features presented in [8] will be explored.

1.2 The SIRKUS Project

This work is performed as a part of the SIRKUS (Spoken Information Retrieval by Knowledge Utilization in Statistical Speech Processing) project[1]. In the SIRKUS project a new automatic speech recogniser is being developed. This system is described in section 2.1. The results of this work are therefore intended to contribute to the SIRKUS project.

1.3 Preceding Work (Master Project)

This work is also strongly connected to the work performed as a part of the master degree during the spring in 2008. In that project an acoustic segmentation algorithm was investigated. In this work, the same algorithm will be used, but the impact of using a different set of features will be explored.

[1] http://www.iet.ntnu.no/projects/sirkus/

1.4 Project Goal

The goal of this work is to explore the cochlea features and their properties. Especially, the use of the features for automatic segmentation is the main focus. Further, it is of interest to investigate the usage of the cochlea features in the SIRKUS project and the SIRKUS recogniser.

1.5 Report Structure

This report is structured using the standard report layout with 6 sections:

Introduction This section.

Theory Describes the most important theory needed for understanding the experiments. Assumes basic knowledge in speech technology.

Experiments A description of the experiments performed.

Results A presentation of the most important results and observations from the experiments.

Discussion A discussion of the results and their impact.

Conclusion The final conclusion of the work.

2 Theory

This section gives a brief overview of the most important theoretical topics of this work. Firstly, the speech recognition system which the resulting segmentation algorithm is intended for, the SIRKUS recogniser, will be presented. In section 2.2 TIMIT, the Speech corpus used, is introduced. Further, the acoustic segmentation algorithm that is the main tool for segmentation is explained in section 2.3. Then, the cochlea features, which are the main focus of this study, are described in section 2.4. In section 2.5 a variant of the cochlea features is presented. Finally, artificial neural networks, which are used as a secondary approach for segmentation, are explored in section 2.6.

2.1 The SIRKUS Speech Recogniser

The SIRKUS project is developing a detection based recogniser. An overview of the recogniser is given in Figure 1. The idea is to detect different phonological features (i.e. manner, place, voicing etc) in the speech signal by using detectors that operate in parallel, and then merge the output from these detectors to find the most likely phoneme sequence.

The recogniser has one branch per (phonological) feature. Each branch is independent and performs four steps: feature extraction, segmentation, segment feature extraction and phonological feature detection. The first feature extraction block extracts the features used in the segmentation block. This should be done with the phonological feature in mind, to provide as much information about the presence of this feature in the output of the segmentation algorithm as possible. The segmentation block then uses the feature vector sequence to divide the signal into segments, and possibly also provides some extra information about the segments. All the information from the segmentation process is then used to perform a segment based feature extraction. The segment features are in turn used by the detector for calculating the probability that the phonological feature of interest is present. These scores are basis for a lattice representation which performs time alignment. Finally, event decoding and linguistic decoding are done to extract the linguistic meaning of the utterance.

Only the first two blocks of the system, the first feature extraction and the segmentation process, are subjects for investigation in this work.

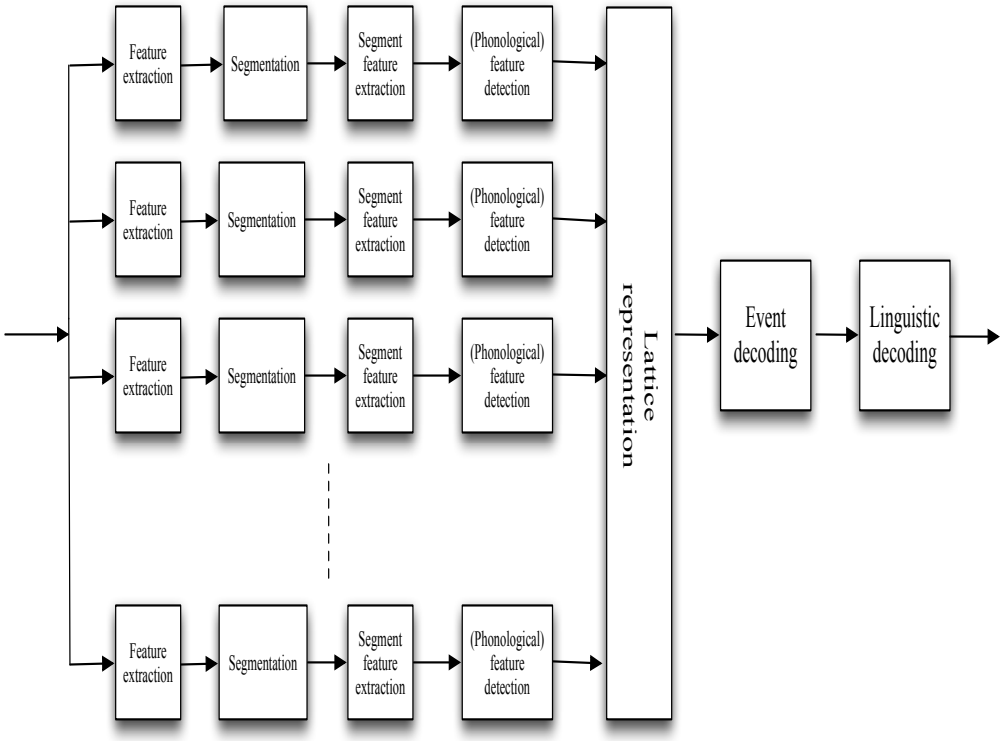

Figure 1: The SIRKUS recogniser

2.2 The TIMIT Speech Corpus

The corpus chosen for the experiments was DARPA TIMIT acoustic-phonetic continous speech corpus ([9]) because it includes a phonological transcription. TIMIT is a corpus of read speech spoken by American English speakers. There are speakers of both sexes and several different dialects. In total there are 630 different speakers, each reading 10 sentences. The corpus is divided into a training set and a test set consisting of 4620 sentences and 1680 sentences respectively.

The phonetic labelling has been valuable for evaluating the different segmentations generated in the experiments.

2.3 The Acoustic Segmentation Algorithm

Automatic segmentation is the process of dividing a speech signal into several segments. Depending on the intended usage of the generated segments, the process uses different criteria. As described in section 2.1, in the SIRKUS recogniser it is of interest to isolate segments containing a specific phonological feature. The algorithm is the same as the one used in [4] and [5].

The goal of the algorithm is to find the set of segments that minimises a distortion measure. The distortion measure used will be explained first, before a description of the algorithm itself is given.

2.3.1 Distortion Calculation

The input to the segmentation algorithm is a sequence of feature vectors, which are to be divided into segments. A segmentation of the vectors can be evaluated by calculating a distortion which describes how good the segmentation given is. If the segmentation is close to a segmentation we want, the distortion should be low, and vice versa. Hence, the choice of how to calculate the distortion will in a large degree decide the properties of the final segmentation.

The distortion measure chosen in this work is found by first finding an approximation of the segment, and then calculate the error made by representing the segment with that approximation.

The mean vector is used as an approximation of the segments, because it is wanted to represent each segment by a constant vector. The idea is that

the features will change their statistical properties at a transition between phonological features. Hence, by dividing the utterance into parts that have a constant representation, the segmentation might describe the phonological boundaries.

Let the input feature vectors belonging to segment i be denoted by $\underline{x}_i[n]$. Further, let the total number of vectors in segment i be N_i. Then the mean vector of segment i is found by

$$\bar{\underline{x}}_i = \frac{1}{N_i} \sum_{n=0}^{N_i-1} \underline{x}_i[n] \tag{1}$$

The Euclidean distance between each of the feature vectors and the mean vector of the corresponding segment is used as a local distortion:

$$d(\underline{x}_i[n], \bar{\underline{x}}_i) = (\underline{x}_i[n] - \bar{\underline{x}}_i)^T (\underline{x}_i[n] - \bar{\underline{x}}_i) \tag{2}$$

The total distortion is finally found by a summation of all the local distortions:

$$D = \sum_{i=0}^{N_s-1} \sum_{n=0}^{N_i-1} d(\underline{x}_i[n], \bar{\underline{x}}_i) \tag{3}$$

where N_s is the number of segments in the utterance.

This is a quite simple and efficient distortion measure which results in segments where the features have a roughly constant mean.

2.3.2 The Segmentation Algorithm

Now, with the distortion measure at hand, the algorithm can be explained. A basic outline of the algorithm is shown in Figure 2. The goal of the segmentation process is to find the segmentation that minimises the total distortion. Since the number of possible segmentations is huge, a simple brute force technique trying them all would demand a good deal of time and resources. Hence, a dynamic programming approach is used: the algorithm begins with only two segments and then adds one segment at a time (level building), until a segmentation that fulfils a certain stopping criterion is found. There are several options for choosing a stopping criterion. A natural choice would be to stop when the distortion reaches a given threshold. However, since the properties of the features used in this work are varying, the threshold would have to be adjusted all the time. Therefore the iteration was stopped when

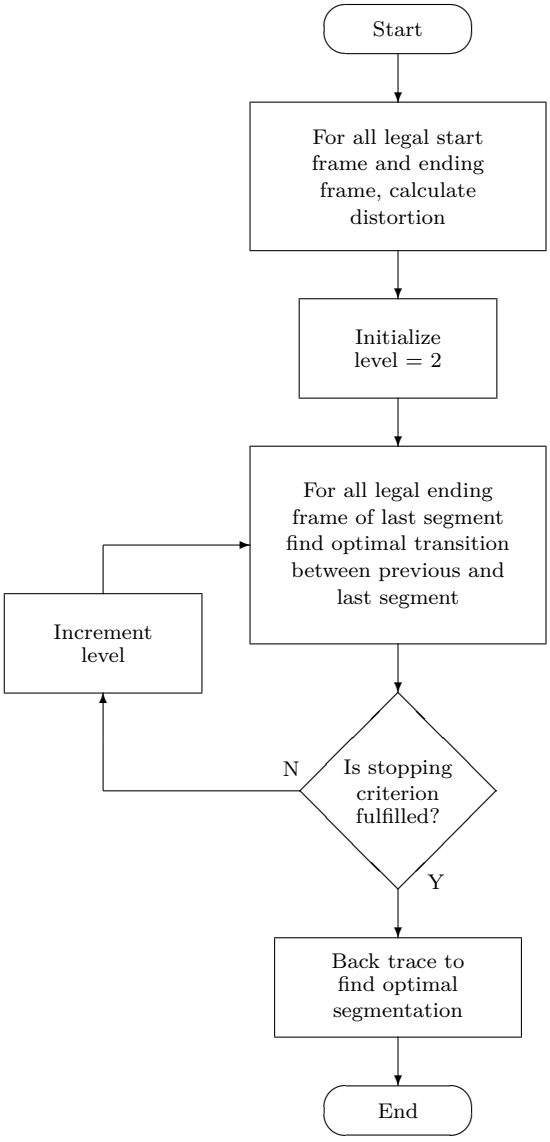

Figure 2: Flow chart of segmentation algorithm

the number of segments was a fixed number of times the number of manual labels from TIMIT. This is of course impossible to do later in a real situation where the manual labelling is unavailable. However, at that time a specific feature set has been chosen, and the threshold can be set accordingly. A simple way of doing this, is to use the mean distortion when the number of segments compared to the TIMIT labelling is fixed.

To decrease the total amount of possible segmentations, a minimum and maximum segment duration were defined. This constraint decreases the amount of different segmentations that need to have their distortion calculated. Since the duration of acoustic events in speech is in the range of a few milliseconds to about 100 ms, the min/max segment duration can be set accordingly. Hence, the segment length criterion can be inserted without doing any harm. Unfortunately, depending on the choice for minimum and maximum segment lengths and the length of the entire utterance, there is still a great amount of different possible segmentations.

Before the level building, the algorithm initialises a distortion matrix containing the distortion of all possible segments (not segmentations!). This is to avoid calculating the distortions several times in the later process. Due to the large amount of different possible segments much of the total CPU time of the algorithm is spent in this step.

The level building step starts by using only two segments (levels), and for every legal ending point (the interval from 2 times the minimum segment duration to 2 times the maximum segment duration) it finds the optimal point of transition between the two segments. The optimal distortion of the two segments is stored along with the placement of the transition. Most likely, the two segments are too short to cover the entire utterance, and the distortion of the segmentation is set to a large value. If the stopping criterion is not fulfilled, the algorithm increases the number of segments and continues in the same manner. For each legal ending point of the last segment, it finds the transition between the two last segments that minimises the total distortion. The total distortion for a given transition point is calculated by simply adding the distortion of the last segment and the distortion of all the previous segments calculated at the previous step. Notice that the latter value was stored together with the the previous optimal transition point. Hence, the last step of the algorithm, after the stopping criterion is reached, is simply to backtrace to find all the optimal transition points.

2.4 Cochlea Features

With the segmentation algorithm explained, an overview of the input to the algorithm and the main purpose of this work can be given. The cochlea features are a feature set designed to simulate the nerve fiber signals which are sent from the ear to the brain. The feature extractor is composed of two main elements: an auditory filter bank composed with gammatone filters proposed by Roy Patterson and John Holdsworth in [10], and a Hair Cell Model proposed by Ray Meddis in [8]. Since these features are a simulation of the signal transmitted from the ears to the brain, they should contain all relevant information in the speech signal.

There are various toolkits available for extracting cochlea features. In this work a toolkit called Auditory Toolbox[2], developed by Malcom Slaney, was used. An overview of the implementation of the filter bank and the Hair Cell model is given next.

2.4.1 Auditory Filter Bank

The auditory filter bank converts the audio waveform into a simulation of the response at different parts of the basilar membrane. The filter bank consists of a set of bandpass filters defined in the time domain by their impulse response on the form:

$$gt(t) = at^{n-1}e^{-2\pi b}\cos(2\pi b f_c t + \Phi), \tag{4}$$

where n is the order of the filter, b is the bandwidth, f_c is the center frequency and Φ is the phase. These filters are called gammatone filter because the envelope of the impulse response is the gammafunction from statistics. Patterson used filters of order 4, and a bandwidth of 1.019 of the Equivalent Rectangular Bandwidth (ERB)

$$ERB(f_c) = 24.7\left(\frac{4.37 f_c}{1000} + 1\right), \tag{5}$$

where f_c is the center frequency of the filter. This bandwidth gives an 3 dB bandwidth of $0.887 ERB(f_c)$. The filters are placed in equal distance in frequency according to the ERB scale. Figure 3 shows a gammatone filter bank with 10 filters on the frequency range 100-16000 Hz.

[2] http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/

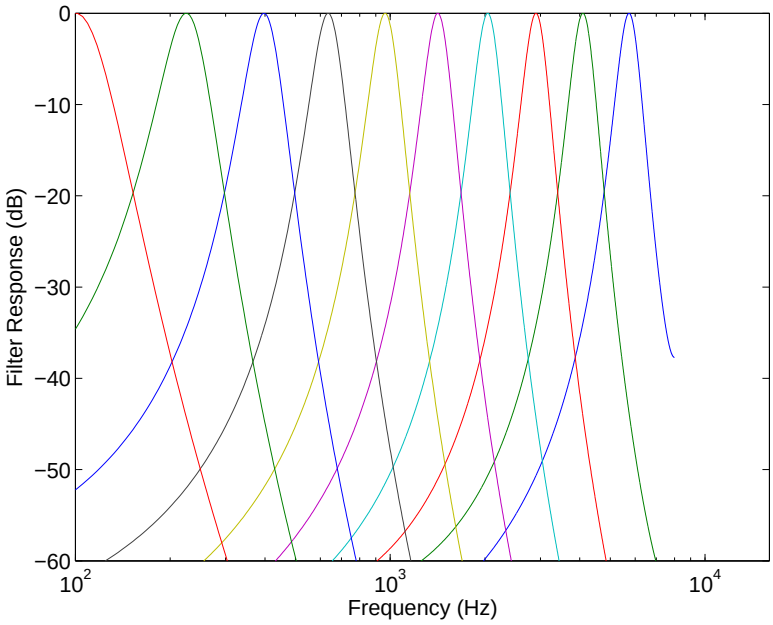

Figure 3: A gammatone filter bank with 10 filters on the range from 100 Hz to 16000 Hz

2.4.2 Meddis Hair Cell Model (MHC)

The Meddis Hair Cell modell takes the basilar membrane vibrations from the gammatone filter bank as input and outputs an estimate of the firing rates of the hair cells. I.e. one filter bank channel is input to one hair cell model. The model is described in [8], where the implementation details are given.

The model describes the movement of something called transmitter substance packages, which give rise to nerve fiber firings when they are located in the pre-synaptic cleft. An overview of the model is depicted in Figure 4. The transmitter substance is located in three reservoirs within the hair cell: a transmitter pool, the cleft and a reprocessing store. In addition new transmitter substance is generated by the factory.

The movement of the transmitter substance can be read out of Figure 4. In the transmitter pool that lies close to the cell membrane, it is an amount of $q(t)$ packets of transmitter substance. These packets are released across the membrane at a rate $k(t)$, which depends on the stimulus of the basilar membrane (the input to the model). The amount of transmitter substance in the cleft determines the probability of nerve fiber firings. The transmitter substance located in the cleft can either be lost by a rate $l \cdot c(t)\, Hz$, where

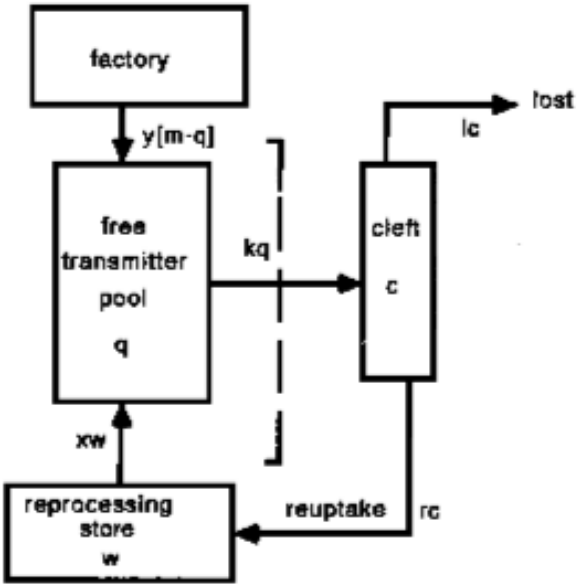

Figure 4: Summary of the operation of the hair cell model with differential equations and parameters for a high spontaneous rate fiber. The figure is copied from [8]

l is a constant, or returned to the cell and into the reprocessing store by a rate $r \cdot c(t)\, Hz$, where r is a constant. The amount of transmitter substance packets $w(t)$ in the reprocessing store are returned to the free transmitter pool at a rate $x \cdot w(t)\, Hz$, where x is a constant. New transmitter packets are generated in the factory and compensate for the loss of packets by refilling the transmitter pool by a rate of $y(M - q(t))\, Hz$, where M is the maximum number of packets in the free pool.

The model is implemented by updating all the reservoir counts for each input value. This is done according to the four equations:

$$\frac{dq}{dt} = y(M - q(t)) + x \cdot w(t) - k(t)q(t) \tag{6}$$

$$\frac{dc}{dt} = k(t)q(t) - l \cdot c(t) - r \cdot c(t) \tag{7}$$

$$\frac{dw}{dt} = r \cdot c(t) - x \cdot w(t) \tag{8}$$

$$k(t) = \begin{cases} g \cdot dt \frac{s(t)+A}{s(t)+A+B} & s(t) + A > 0 \\ 0 & s(t) + A < 0 \end{cases} \tag{9}$$

where dt in practice is the sampling rate of the input $s(t)$, and should be a

value less than 0.1 ms[3]. The amount of transmitter substance in the cleft $c(t)$ gives the probability of a nerve fiber firing and is the output of the model.

An important observation is that the amount of transmitter packets in the free pool is high at the onset of a sound, but decreases to a standby level while the sound lasts. Figure 5 shows the response of the MHC ($c(t)$) when a sequence of silence, a sine and silence is used as input, and is a good example of this behaviour.

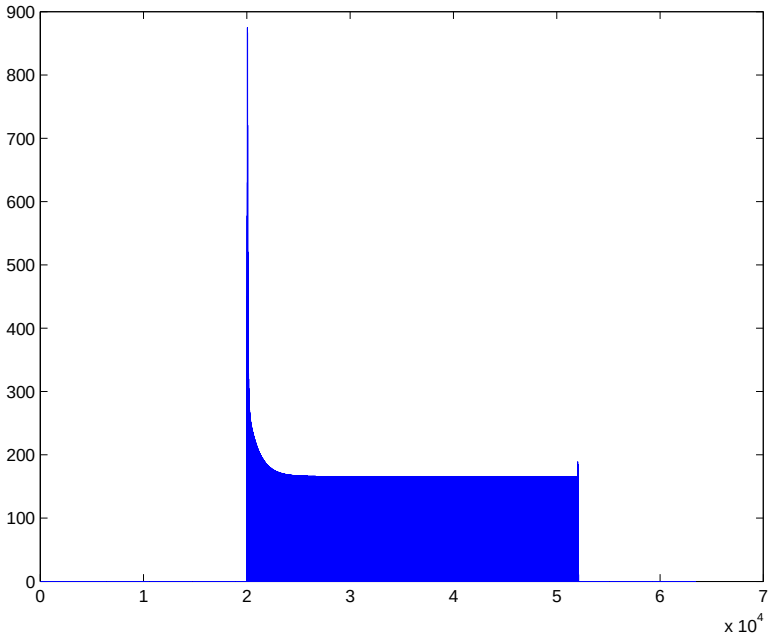

Figure 5: Meddis Hair Cell response of a sine signal

The model has several parameters, which should be set with care. Mainly the parameters suggested in [11] have been used, but also the parameters for medium spontaneous rate fibers proposed in [8] have been tried.

[3]The notation of using t and dt is because the model simulates a continuous process, and is the same as used in [8]

2.5 Cochlea Zero Crossings with Peak Amplitude Features

Zero crossings with peak amplitudes (ZCPA) described in [12] are a set of features used for speech recognition. In this work another feature set based on ZCPA was tried. They are generated by replacing the filter bank in ZCPA with the cochlea feature extractor described in the previous section. The resulting system is shown in Figure 6.

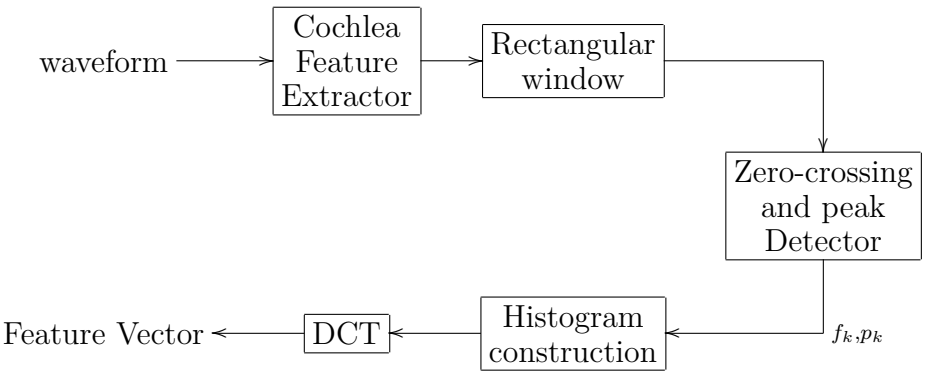

Figure 6: Cochlea ZCPA

The cochlea feature extractor generates feature vectors at the same sampling frequency as the input. All the feature vector sequences are then windowed with a rectangular window before a zero-crossing detector detects the positive zero crossings. Then, for each pair of zero crossings (z_k, z_{k+1}) the inverse of the zero crossing interval is calculated

$$f_k = \frac{1}{z_{k+1} - z_k}, \tag{10}$$

where z_k denotes the time of the kth zero crossing. These frequencies are input to a histogram construction together with the peak amplitude p_k between the two zero crossings. The histogram construction is performed by dividing the frequency axis into frequency bins with equal width on the mel-frequency scale. For each of the frequencies, f_k, the corresponding bin ($b_j = bin(f_k)$) is incremented by the natural logarithm of the peak amplitude

$$b_j^{new} = b_j^{old} + \ln(p_k) \tag{11}$$

Finally, a discrete cosine transform[4] (DCT) is performed on the histogram for decorrelation purposes. Next, the window is shifted for the next ZCPA vector.

2.6 Artificial Neural Networks (ANN)

An Artificial Neural Network (ANN), often called Neural Network, is a type of classifier. The name comes from the similarity with biological neural networks. An ANN also consists of a number of neurons or nodes connected in a network. The neurons performs a processing of the input before sending the result further on, to other neurons. There are several types of ANNs varying in layout and type of processing units. In Figure 7 one of the simplest kinds of ANNs, a feed-forward neural network, is depicted.

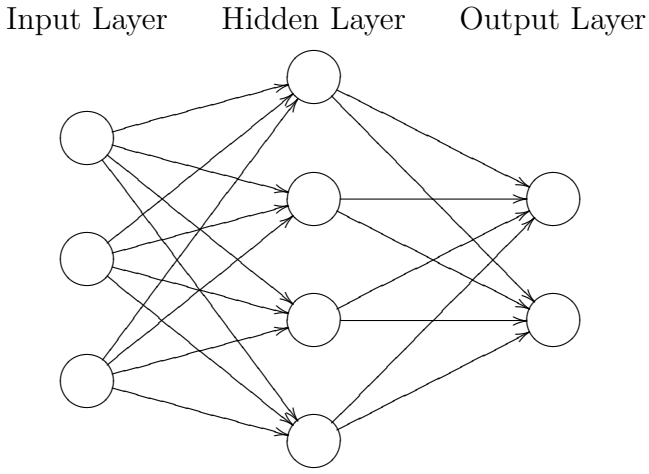

Figure 7: A feed forward neural network with three input nodes, four hidden nodes and two output nodes

In this type of ANNs the information is sent in one direction, forward, from the input layer through hidden layers (if any) to the output layer. The feed-forward neural network shown has three input nodes, one hidden layer with four hidden nodes and two output nodes. The nodes in the input layer accepts three input values, and passes them to the hidden layer, which performs processing on the input values before sending them to the output layer which performs the computations of the final stage.

[4]There are several definition of the DCT. Here the DCT-II is used.

2.6.1 Perceptron Neurons

The neurons in the hidden layer(s) and output layer of a neural network are processing nodes which perform calculations on the input. One of the first neurons invented and the one used in this work was the Perceptron [13], which performs a weighted sum of all the inputs before a threshold is subtracted. To generate binary output the result is often sent through a hard limiting non-linearity to ensure output to either 1/0 (or 1/-1). The Perceptron neuron is shown in Figure 8.

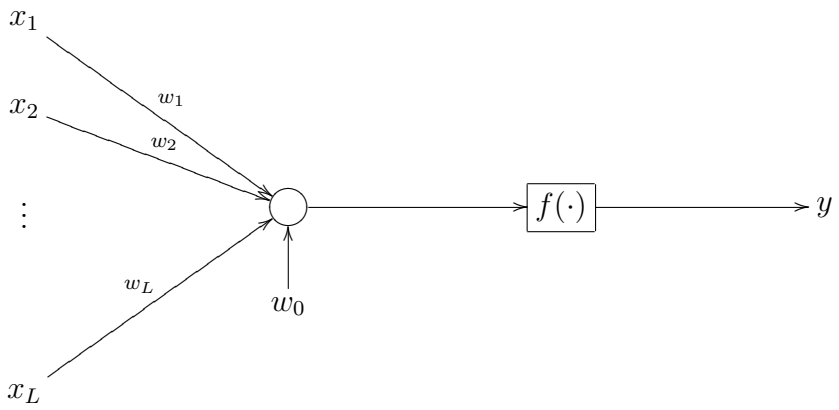

Figure 8: A Perceptron Node

The Perceptron neuron in itself is a two class linear classifier. It divides the input space into two regions by a hyperplane defined by the weights. This can be written as

$$f(\underline{x}) = \underline{x}^T \underline{w} - w_0 < 0 \qquad \text{Assign to class 1} \tag{12}$$

$$f(\underline{x}) = \underline{x}^T \underline{w} - w_0 > 0 \qquad \text{Assign to class 2} \tag{13}$$

where $\underline{w} = [w_1\, w_2\, \ldots\, w_L]^T$ is the vector containing the weights, and $\underline{x}$ is the input vector of the node. To compact the notation the threshold can be moved into the first term by redefining; $\underline{x'} = [1\, x_1\, x_2\, \ldots\, x_L]^T$, $\underline{w'_0} = [w'_0\, w_1\, \ldots\, w_L]^T$ and $w'_0 = -w_0$, which will be used from now on.

2.6.2 The Multi-Layer Perceptron (MLP)

By using several Perceptrons in parallel in the hidden layer, one can create several hyperplanes. This divides the hyperspace into several regions, which

are then mapped to one of the vertices in a N_h dimensional hypercube, where N_h is the number of nodes in the hidden layer. Finally, each of the nodes in the output layer divides these vertices into two classes by drawing hyperplanes. By making sure that only one output node outputs 1 for any vector, each of the output nodes corresponds to one output class. The resulting neural network is a two-layer Perceptron[5]. It is possible to extend with more hidden layers. This will give the ability to divide the vertices of the hypercubes using several hyperplanes instead of one, which means that any union of vertices can belong to a class, in contrast to only neighbouring vertices. However, the ability to separate classes consisting of any union of regions is not always necessary, and the increased complexity of adding another hidden layer is usually not worth the payoff. Hence, a two-layer Perceptron was chosen in this work.

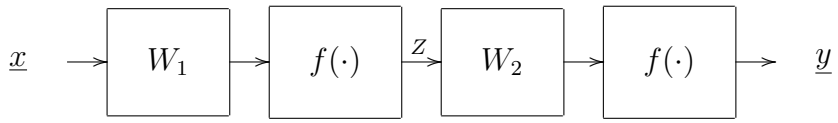

Figure 9: Vectorised view of a two-layer Perceptron

If all the weights from one layer are put into one matrix with the weights from each node as a column, each layer can be viewed as a matrix multiplication combined with an non-linearity to ensure binary output. Hence, the two-layer Perceptron viewed as in Figure 9, where the outputs of the layers are given by

$$Z = f(\underline{x}^T W_1) \tag{14}$$

$$\underline{y} = f(Z^T W_2) \tag{15}$$

where $f(\cdot)$ denotes the non-linearity, and operates on each element. The dimension of Z is given by the input dimension and number of nodes in the hidden layer, while the output vector has a dimension equal to the output nodes - one for each class.

[5]There is some disagreement in how to count the layers of a feed-forward neural net. However, in this work a two-layer Perceptron means a Perceptron with two calculating layers, including the output layer.

2.6.3 Backpropagation Training of MLP

There are several approaches for training MLPs and according to [14], the most popular training algorithm for non separable problems is the backpropagation algorithm. This algorithm belongs to the class of assisted training algorithms, i.e. the data must be labelled. Based on this data it minimises a cost function with respect to the weights in the structure.

The cost function that is most commonly used is the sum of squared error at the output

$$J = \sum_{n=1}^{N} \|\underline{y}(n) - \underline{\hat{y}}(n)\|^2 \tag{16}$$

where N is the number of data vectors in the training set. The weights in the neural net are found by minimising the cost function. There are several ways of doing this, but a method called gradient descent is a popular choice. The gradient descent is an iterative algorithm which uses all the data, before updating the weights by an equation of the form

$$\underline{w}_{new} = \underline{w}_{old} - \alpha \frac{\partial J}{\partial \underline{w}_{old}} \tag{17}$$

where α is a constant. Now, since the cost function J is a function of the activation function f, we must be able differentiate f. This is not possible if f is chosen to be the unit step function[6]

$$f(x) = \begin{cases} 0 & x < 0 \\ 1 & x > 0 \end{cases} \tag{18}$$

This problem is solved by using functions that approximate the unit step function. A popular choice for such a function is the sigmoid function

$$f(x) = \frac{1}{1 + e^{-ax}} \tag{19}$$

which is displayed in Figure 10 for three choices of the constant a.

The training algorithm begins by updating the weights in the output layer using equation (17), before the error is sent backward to the previous layer using the weights. Hence, the name "Backpropagation algorithm". By using

[6] A neuron that employs the unit step function as activation function is called a McCulloch-Pitts neuron

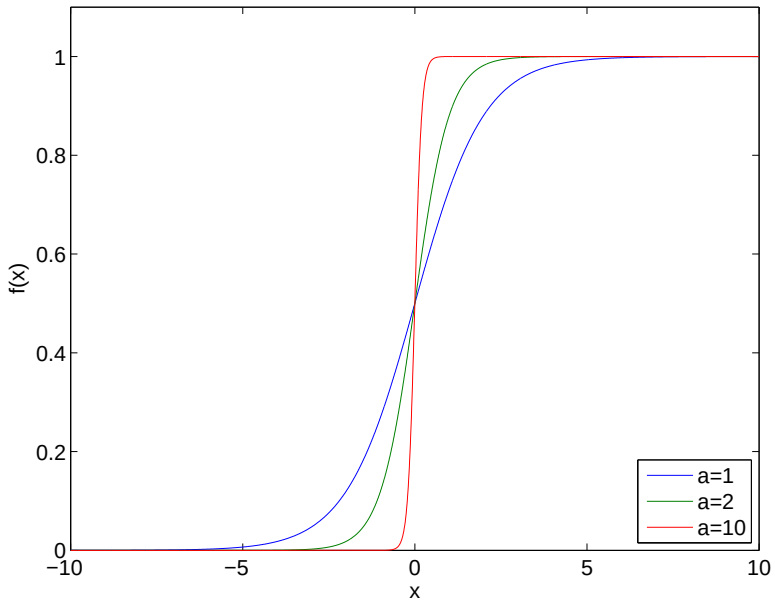

Figure 10: The sigmoid function for different values of a

the error sent back from the next layer, the algorithm continues to update the weights and propagate the error further back.

When all the training data has been applied to the training algorithm a test set is used to evaluate the weights. If the new weights perform better than the best weights so far (highest average class accuracy), they are kept. If a stopping criterion is not met, another iteration is performed. On the other hand, if the criterion is met the weights from the best iteration are returned.

There are several criteria available for determine when to stop iterating. [15] suggests to terminate the iterations either when the cost function J or its gradient with regard to the weights becomes small. In this work however, the training and testing have for simplicity been run 500 times.

3 Experiments

In this section some of the experiments performed will be presented. Since trial and error is the main ingredient in this work, it is impossible to present all the experiments. Hence, only a selection of the experiments is mentioned. The section is not a chronological presentation of the work, but is instead divided into the main subjects: cochlea features in section 3.1, the cochlea ZCPA features in section 3.2 and automatic segmentation in section 3.3.

3.1 Cochlea Features

The process of extracting the cochlea features contains many parameters which need to be tuned. This is a big problem since one is basically forced to perform a trial and error approach to optimise. To evaluate the impact of the different parameters, various plots of the features and the resulting segmentation were used.

3.1.1 Filter bank

An important parameter of the filter bank is the number of filters (also called channels in this work). A large amount of filters will give narrow-band analysis of the signal and produce more features, while fewer filters will result in broader filters and fewer features. Values used in the experiments have varied from 10 to 80.

Also, the outputs of the filters are normalised with regard to the response of the center frequency of the filter, which is also the maximum response. Since the bandwidth of the filters increase with increasing center frequency, the filters with highest frequency will have an advantage. In speech most of the important information is in the lower frequencies. Hence, a normalisation of the filters with regard to the energy of the unit step response seems more beneficial.

3.1.2 Meddis Hair Cell Parameters

The MHC module is definitively the part of the process using most parameters. Fortunately Meddis presents two sets of parameters in [8], these are also given in Table 1 (see Figure 4 on page 13 for explanation of the parameters). Both parameter sets were tried.

Parameter	Set 1	Set2
A	5	10
B	300	3000
g release	2000	1000
y replenish	5.05	5.05
l loss	2580	2500
r recovery	6580	6580
x reprocessing	66.31	66.31

Table 1: Meddis Hair Cell Parameters from [8]

In addition to the parameters inside the model, the scaling of the input can be varied. This will impact the amount of transmitter substance moving from the free transmitter pool to the cleft, which in turn will change the probability of a firing. From equation (9): it is apparent that an increase in energy will give higher output activity.

3.1.3 Signal Processing of the Features

Before the features can be used by the acoustic segmentation algorithm, the sampling frequency has to be reduced. Initially this is the same as the sampling frequency of the input, which is normally 16 kHz. Such a high sampling frequency leads to an unreasonable high number of possible different segmentations, hence it has to be reduced. Two ways of doing this have been tried: ordinary decimation and windowing. The first alternative is simply to keep every D vector and remove the rest after a low-pass filtering, where D is the decimation factor. Both 50 and 100 were tried as decimation factors. This gives respectively a sampling frequency of 320 Hz and 160 Hz, or a frame shift of 3.125 ms or 6.25 ms. This is a higher sampling frequency than what is normal when using MFCCs, where a window shift of about 10 ms seems to be typical. The other alternative of reducing the sampling frequency is performed by first windowing the signal and then calculating the mean inside the window. A Matlab code is given for this in Algorithm 1. The sampling frequency of the output is in this way controlled by the size of the window shifts.

A problem that emerged was the noisiness of the features. They contain a great deal of abrupt changes in time, which results in a high distortion in the acoustic segmentation algorithm. To reduce this the bandwidth of the LP filter in the decimation process were decreased. Bandwidths as narrow

Algorithm 1: Matlab code for windowing operation

```
% Assume data is a matrix containing the features.
% A row corresponds to a channel.
twin=0.010;      % window size (seconds)
tshift=0.00625;  % window shift (seconds)
wsize=Fs*twin;
wshift=Fs*tshift;
w=hamming(wsize)'; % Hamming window
Nfrms=floor((size(data,2)-wsize)/wshift);
nChan=size(data,1);
h=zeros(nChan,Nfrms);
for i=1:nChan
   for j=0:Nfrms-1
      h(i,j+1)=mean(data(i,j*wshift+1:j*wshift+wsize).*w);
   end
end
```

as 1/10th the sampling frequencies were tried. When using the windowing function for reducing the sampling frequency, increased smoothing can be achieved using a larger window.

After studying figures of the features, a hypothesis that most of the information regarding segments is above the mean of the features was produced. To test this, a subtraction of the mean succeeded by a half wave rectifier was applied.

3.2 Cochlea ZCPA Features

The idea of combining cochlea features and ZCPA came in the end of the work. Hence, it was not very much time for exploring the different parameters.

When using the ZCPA algorithm there are four parameters that have to be adjusted:

Window Size: window sizes in the range of 15 ms to 100 ms were used. 15 ms is in the usual range for MFCCs, but this gives small amount of data for the histogram construction. Hence, as large as 100 ms windows were tried.

Window shift: the amount the window is moved between each feature vector. Because of the time consumption of the algorithm the experiments were run with 10 ms window shift and not lower.

Max frequency: the maximum frequency in the histogram construction. Since the sampling frequency of the features is 16000 Hz, the same as the input waveform, the highest frequency is 8000 Hz. However, using a lower value results in a low pass filtering, which might be beneficial considering the noisy behaviour of the features. Thus both 4000 Hz and 8000 Hz were tried.

Frequency bins: the number of the frequency bins in the histogram. A high number of frequency bins gives a high frequency resolution, but requires wider windows or more cochlea channels to get enough data within each bin. Both 30 and 60 bins were used.

Input to the ZCPA were cochlea features with dimension of 10, 20 and 40.

3.3 Automatic Segmentation

In the beginning the features were intended to be used by the segmentation algorithm. Hence, nearly all of the features extracted in the previous sections were tried as input to the acoustic segmentation algorithm. Since the cochlea features from section 4.1 did not give the expected performance with the acoustic segmentation algorithm, another approach for segmentation using ANN was tried.

3.3.1 Acoustic Segmentation

Because the main focus was on the features the acoustic segmentation algorithm was not target for any parameter variation. Instead, the algorithm was used only to evaluate the impact of the different parameters of the feature extraction process. This was done by performing segmentation on some utterances from TIMIT. In previous work Praat[7] has been used for this. Unfortunately, it does not seem like there is an easy way of displaying the cochlea features together with the segmentation in Praat. So to be able to visualise the segmentation and the features a Python script was made.

[7] http://www.fon.hum.uva.nl/praat/

3.3.2 ANN Segmentation

After the process of subtracting mean and half wave rectifying, the features seem to have distinctive behaviour near the boundaries. In an effort to utilise this, an MLP was used to classify the vectors as either a boundary vector or not. The manual labeling was used to decide which vectors belong to the boundary class in the training data. This was done by putting the vectors describing a frame containing a manual boundary into the boundary class. In the experiments with the MLP the TIMIT training set and test set were used for training and testing.

A problem with the first experiments was that too few vectors were in the boundary class. This means that a low cost can be achieved by classifying only a few vectors as boundaries, since this results in a small number of misclassifications. Hence, the result is too few segments in the final segmentation. In an attempt to fix this, the cost function J (see section 2.6.3) in the training step was scaled with the inverse of the priors for the classes. This makes the class with the low prior count more in the error calculation. I.e. a vector from the boundary class classified as non-boundary will result in a larger error than before.

Since the manual labelling does not need to be the only correct segmentation, and a transition between segments is continuous, the two vectors surrounding the boundary vectors can also be put into the boundary class. This will further reduce the problem with the skewed distribution of the classes and therefore give more boundaries.

The amount of hidden nodes in the MLP had to be chosen. The only way to do this is again by trial-and-error. Three values were tried: 20, 50 and 100.

Normally the output node giving the largest value is selected as the correct class. However, since the output nodes give a value in the range $[0, 1]$ it is also possible to compare one of the two nodes with a threshold. In addition, by weighting the output node with a cost function dependent on the amount of frames since last boundary was detected, a problem with succeeding frames being all detected as boundaries (see section 4.3) can be reduced. A weight of the form

$$w = 1 - e^{-an} \tag{20}$$

where n is the number of frames since last boundary, and a was set to about 0.1, was used.

Transitions between unvoiced-unvoiced, voiced-unvoiced, unvoiced-voiced and

voiced-voiced (see Table 2) sounds[8] will act differently, and perhaps distribute the boundary class into several clusters, and therefore increase the need of nodes in the hidden layer. Dividing the boundary vectors into four classes (one for each of the cases) might reduce this problem. Of course, this will again increase the problem with a skewed distribution of the classes.

Class	**Phonemes**
Voiced	b, bcl, d, dcl, g, gcl, dx, z, zh, jh, v, dh, m, n, ng, em, en, eng, nx, l, r, w, y, hv, el, iy, ih, eh, ey, ae, aa, aw, ay, ah, ao, oy, ow, uh, uw, ux, er, ax, ix, axr, ax-h
Unvoiced	p, pcl, t, tcl, k, kcl, q, qcl, ch, s, sh, f, th, hh
Others	pau, epi, h#

Table 2: Voiced and unvoiced phonemes in TIMIT.

Since the boundaries seemed to give a onset of the features, the derivatives should have a large peak. To try to utilise this in the classification also Δ and $\Delta\Delta$ features were added to the feature vectors. These were generated by

$$\Delta \underline{c}_k = \underline{c}_{k+1} - \underline{c}_{k-1} \tag{21}$$

$$\Delta\Delta \underline{c}_k = \Delta \underline{c}_{k+1} - \Delta \underline{c}_{k-1} \tag{22}$$

where $\underline{c}_k$ is the cochlea feature vector at time k. This results in feature vectors with dimension double (only Δ) or triple (both) the size of the original vector. Because of the resource consumption both could not be used together with 40 channels.

[8]Labels from the "Other" class, i.e various pauses, were considered unvoiced.

4 Results

During the experiments in this work, a great deal of results were generated. In this section some of the figures and tables are shown, together with an explanation of the most important observations. The section is divided into subsections to match the subsections of section 3. One exception is the results from the acoustic segmentation which are presented during section 4.1 and 4.2, since they are used as a criterion for evaluating the features.

4.1 Cochlea Features

In Figures 11 and 12 a basic setting of the features and the resulting segmentation is shown. The features are generated with the use of 40 channels and a decimation with a factor of 100. Notice how some parts of the utterance are given a high portion of boundaries as close together as the minimum segment duration given in the algorithm, while other parts are empty only given some boundaries to fulfil the maximum segment duration.

Similarly, the Figures 13 and 14 shows features where each of the filters in the filter bank have been normalised with the energy of its unit sample response. Still the problem with very uneven distribution of the boundaries are present. Also, notice that some of the information in the higher frequency areas have been lost.

The features shown in Figures 15 and 16 have in addition a scaling of the input to the MHC model by a factor of 500 instead of the original 80. In Figure 17 the same features have been used, but the number of segments has been set equal to the number of manual segments. These figures show that the information in the higher frequency is back again. Also, the features have a more natural distribution of the energy in the frequency domain. The segments are now distributed a little bit more even across the utterance, but the problem still needs improvements.

The Figures 18 and 19 shows a plot of 4 of the 40 channels when decimation and windowing respectively are used. These figures shows that the windowing alternative gave a more noisy result than the decimation operation.

In Figure 20 the same settings as in figure 18 are used except from that parameter set 2 given in Table 1 on page 22 is employed. Although these features look less noisy than the previous plots, they gave a segmentation that had a much poorer distribution of the boundaries than the features

generated with parameter set 1.

Another observation worth noting about the channel plots in the previous mentioned figures is that the features seems to vary around a specific standby value. Onsets in the channels give a large spike before the signal goes back down and often below the standby value to recharge back up. Hence, it seems like the information that is of interest in the segmentation process is above this value. This observation gave the idea of half wave rectifying around this standby value. Unfortunately, this did not improve the segmentation output. However, by comparing the features with the manual labelling like in Figure 21, it is evident that the features often either have an onset or offset near a manual boundary. The figure was generated by taking the maximum of the channels corresponding to the subbands [0-1000, 1000-2000, 2000-4000, 4000-8000] for each frame. These four signals were then imported into Praat as two stereo wav files.

Although the figures given here are generated using 40 channels, several other values in the range 10-80 were tried. However, it was hard to get an impression of the impact of the parameter. In some cases a low value seemed to give a slightly better match compared to the manual labelling than a higher value, while in other cases the opposite seemed true.

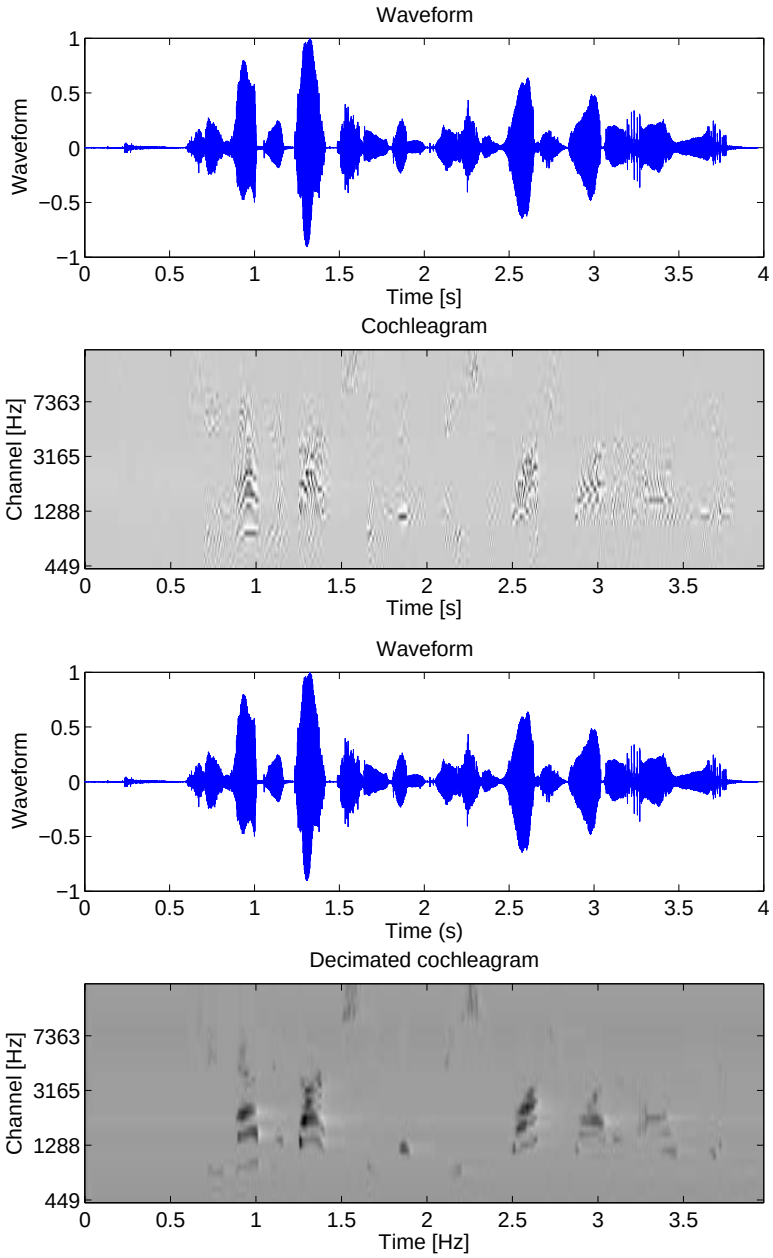

Figure 11: Cochlea features, 40 channels. The raw version on top, and the decimated version below.

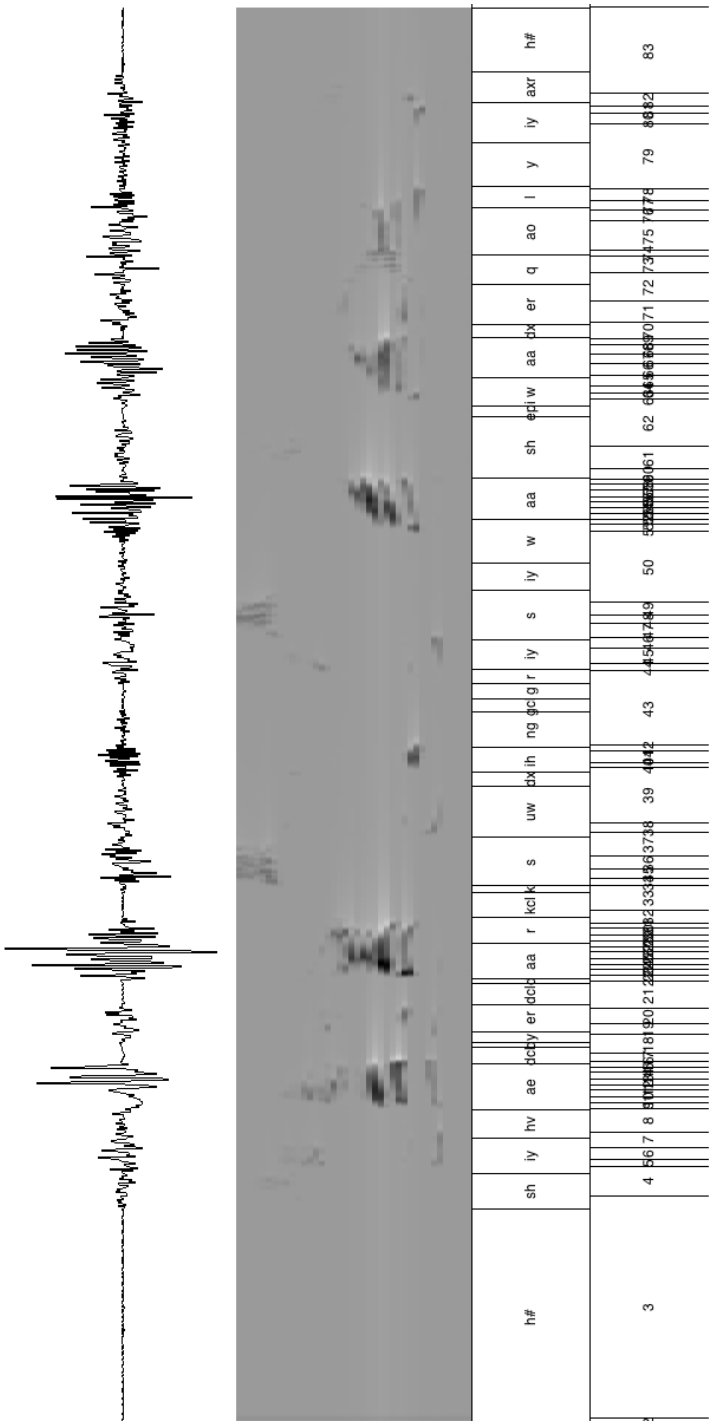

Figure 12: Acoustic segmentation based on features from Figure 11.

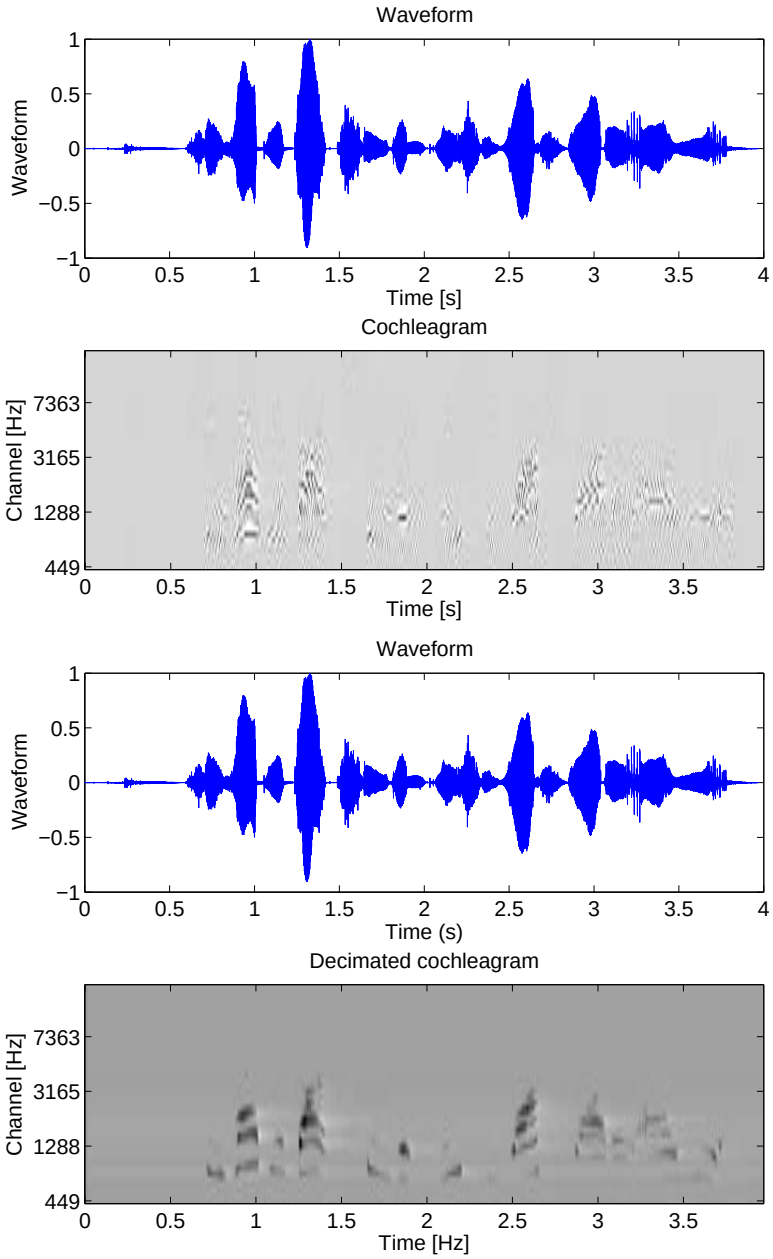

Figure 13: Cochlea features, 40 channels. Filter bank normalised with the energy of the unit sample response.

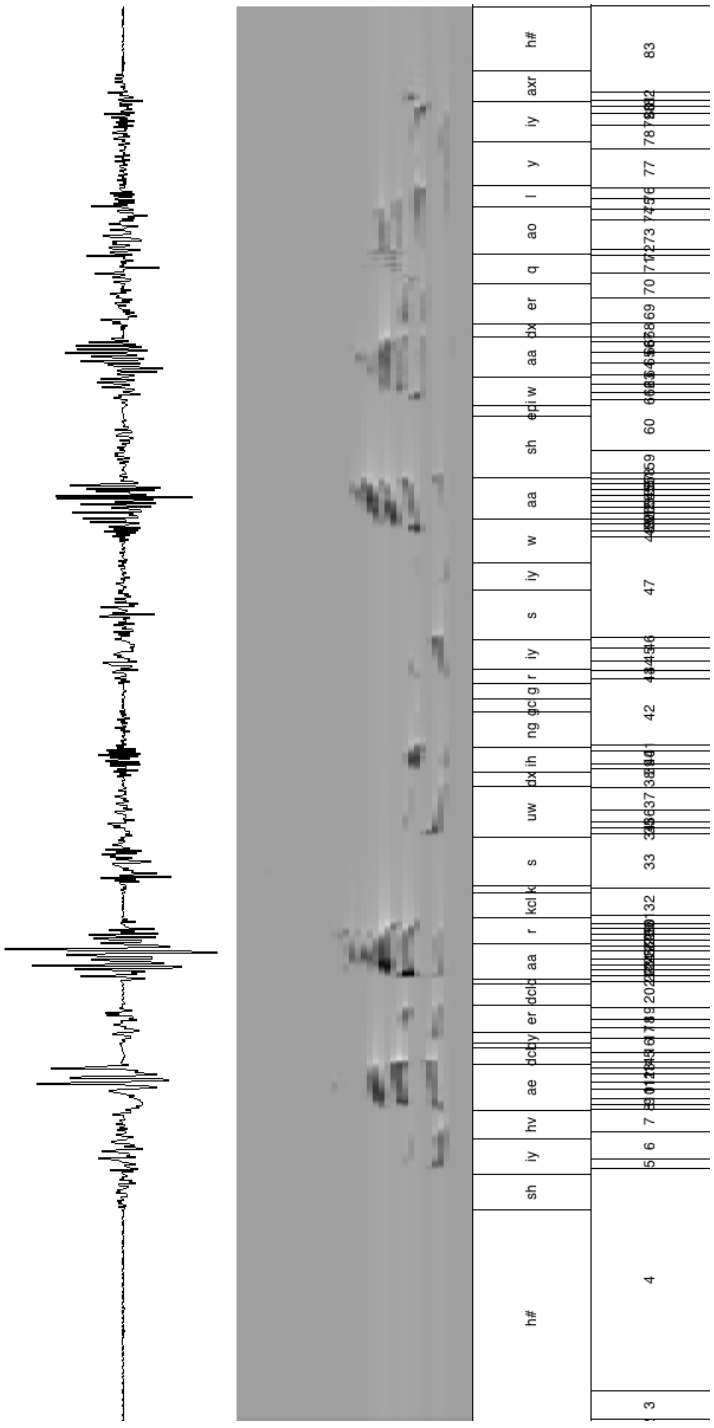

Figure 14: Acoustic segmentation based on features from Figure 13.

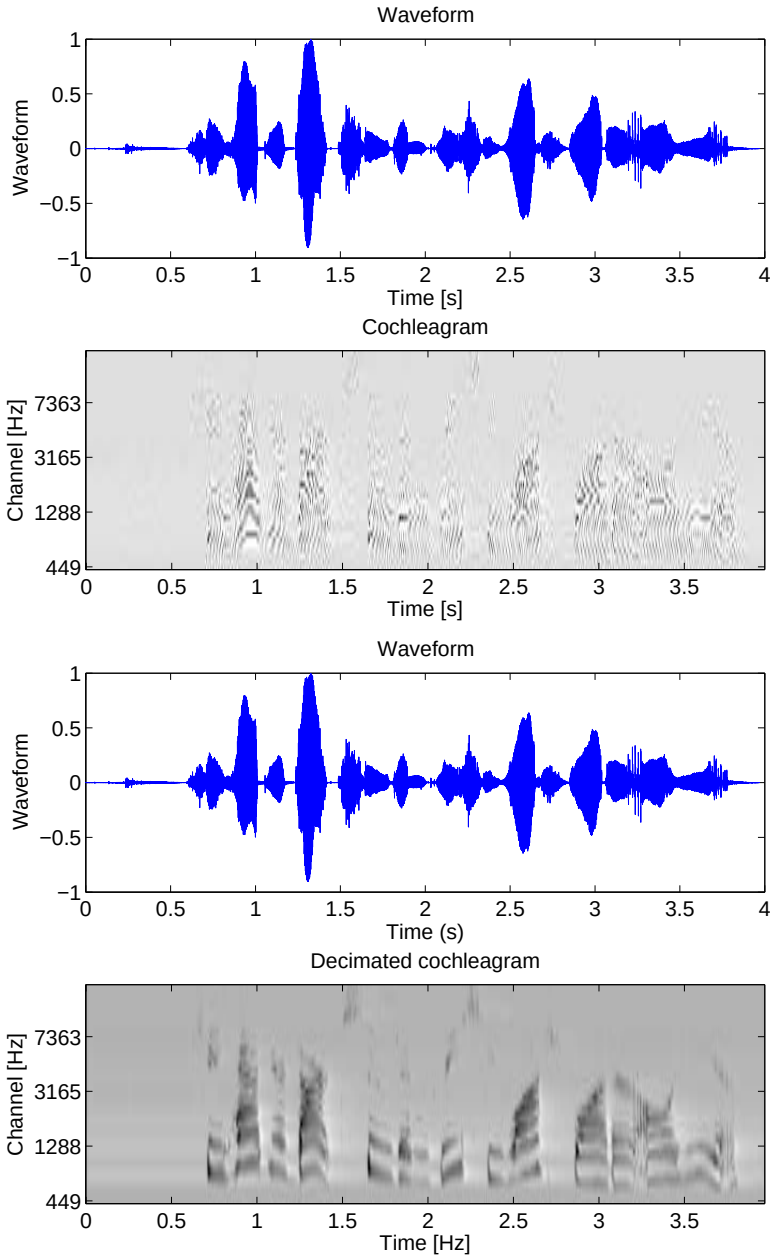

Figure 15: Cochlea features, 40 channels. Filter bank normalised with regard to energy of the unit sample response. Input to MHC scaled by 500.

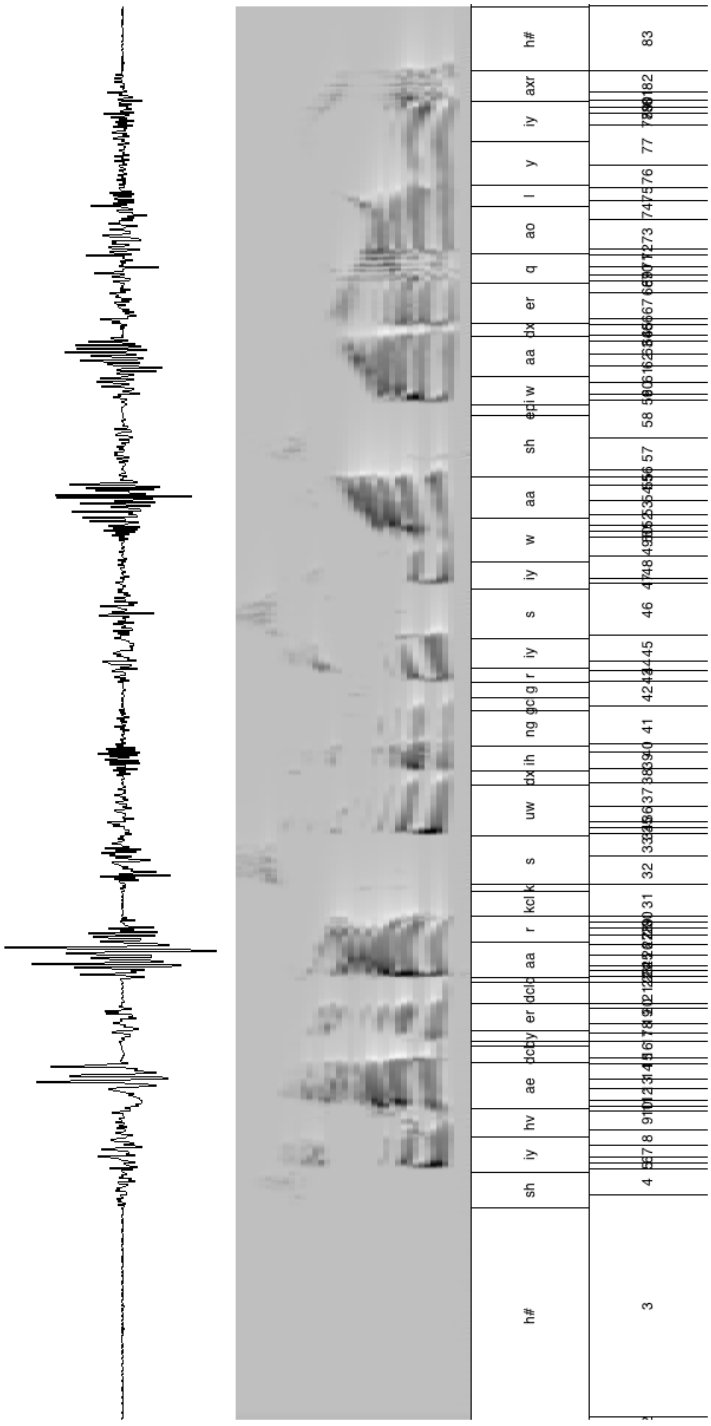

Figure 16: Acoustic segmentation based on features from Figure 15.

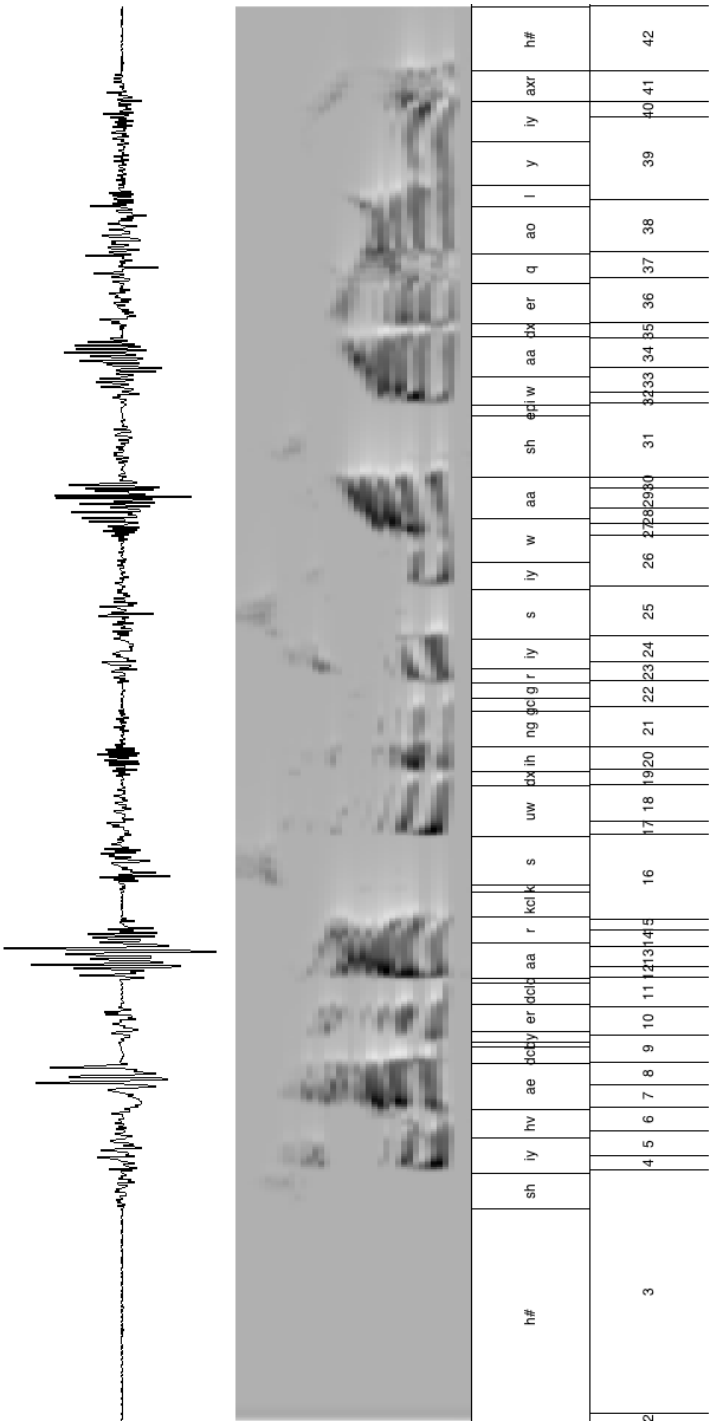

Figure 17: Acoustic segmentation based on features from Figure 15 with number of segments equal to the number of manual segments.

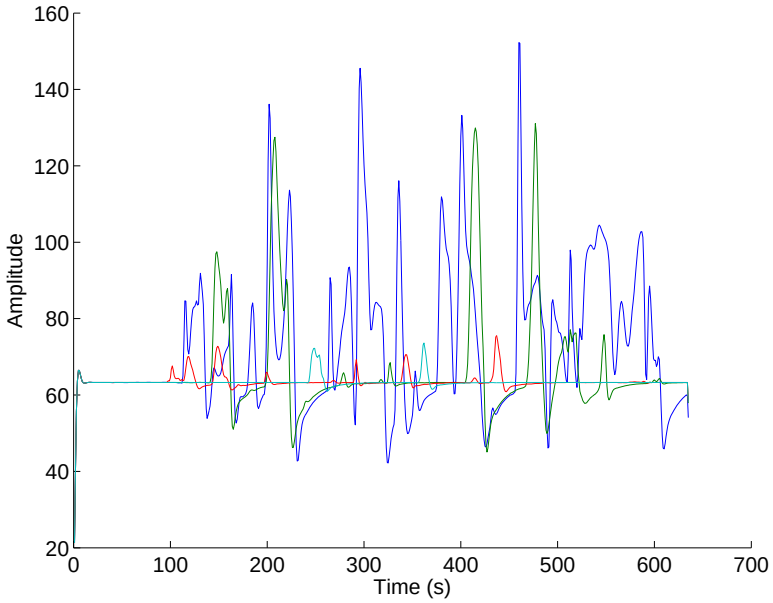

Figure 18: Plot of channel 10, 20, 30 and 40, when 40 channels and decimation with a stricter LP filter are used.

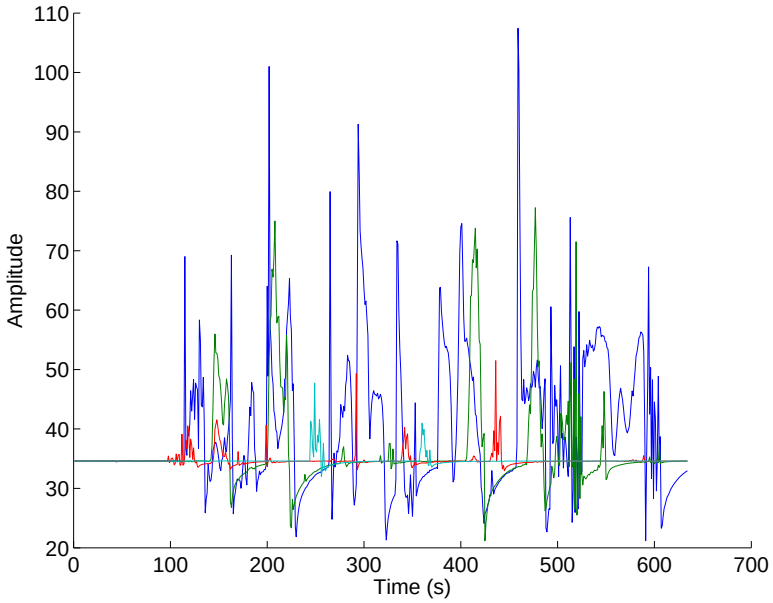

Figure 19: Plot of channels 10, 20, 30 and 40 after windowing operation with window size 5 ms and shift size 6.25 ms.

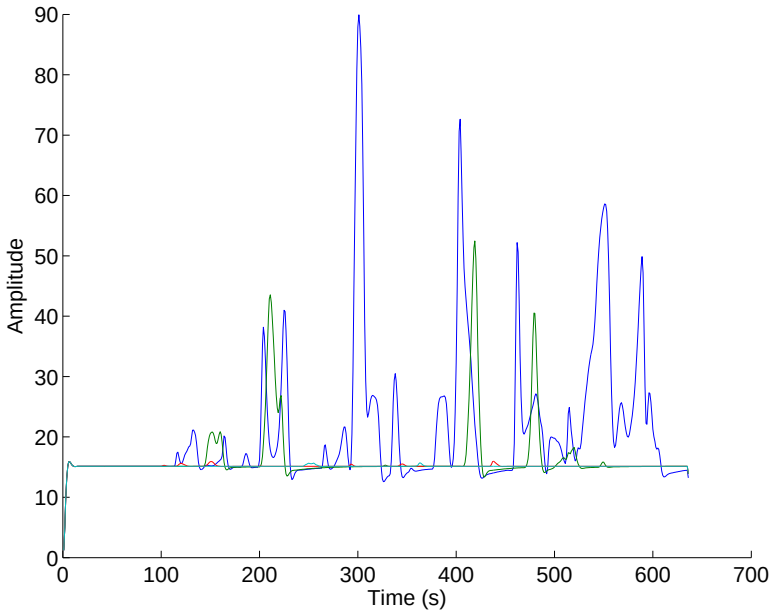

Figure 20: Plot of channels 10, 20, 30 and 40 after MHC parameter set 2 in Table 1 on page 22.

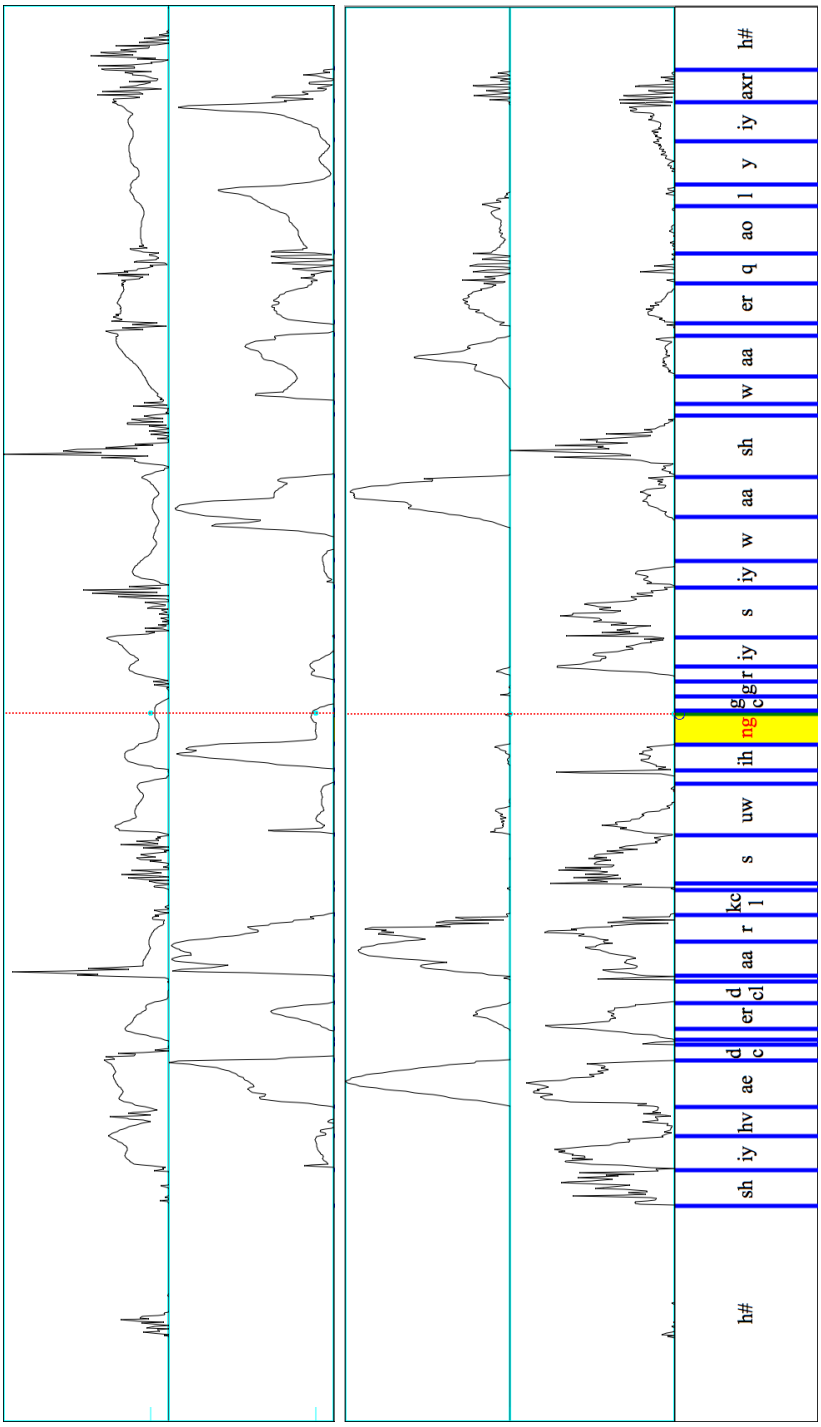

Figure 21: The maximum value of the channels corresponding to [0-1000,1000-2000,2000-4000,4000-8000] Hz after mean subtraction and half wave rectifier compared with the manual labelling using Praat.

4.2 Cochlea ZCPA Features

In Figure 22 the cochlea ZCPA features before the DCT are depicted. The features are generated with 20 channels, max frequency 8000 Hz, 60 frequency bins, window size of 30 ms and window shift of 10 ms. Figure 23 shows the resulting segmentation (after the DCT has been applied). It is evident that this result is much better than the results using the normal cochlea features. The problem of segments with minimum and maximum duration is nearly gone, and it seems that the boundaries have a high correspondence with the manual boundaries.

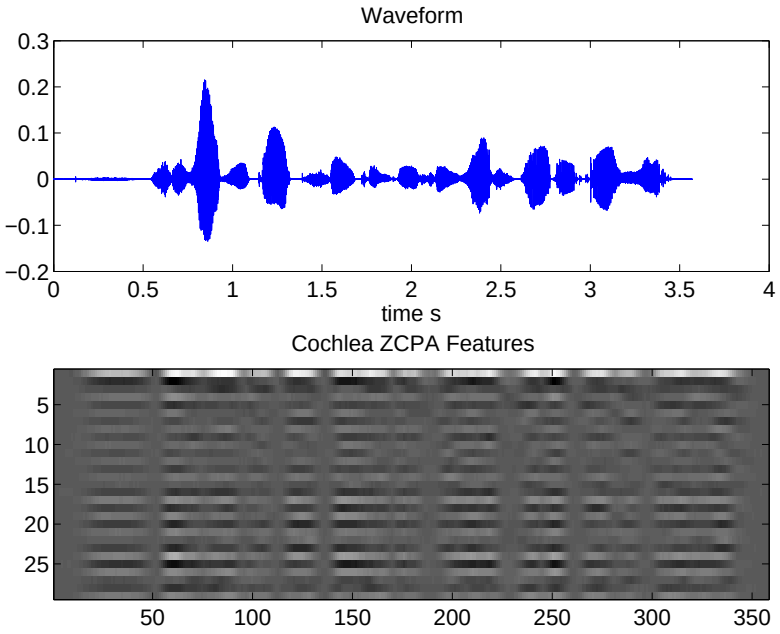

Figure 22: Cochlea ZCPA Features before DCT

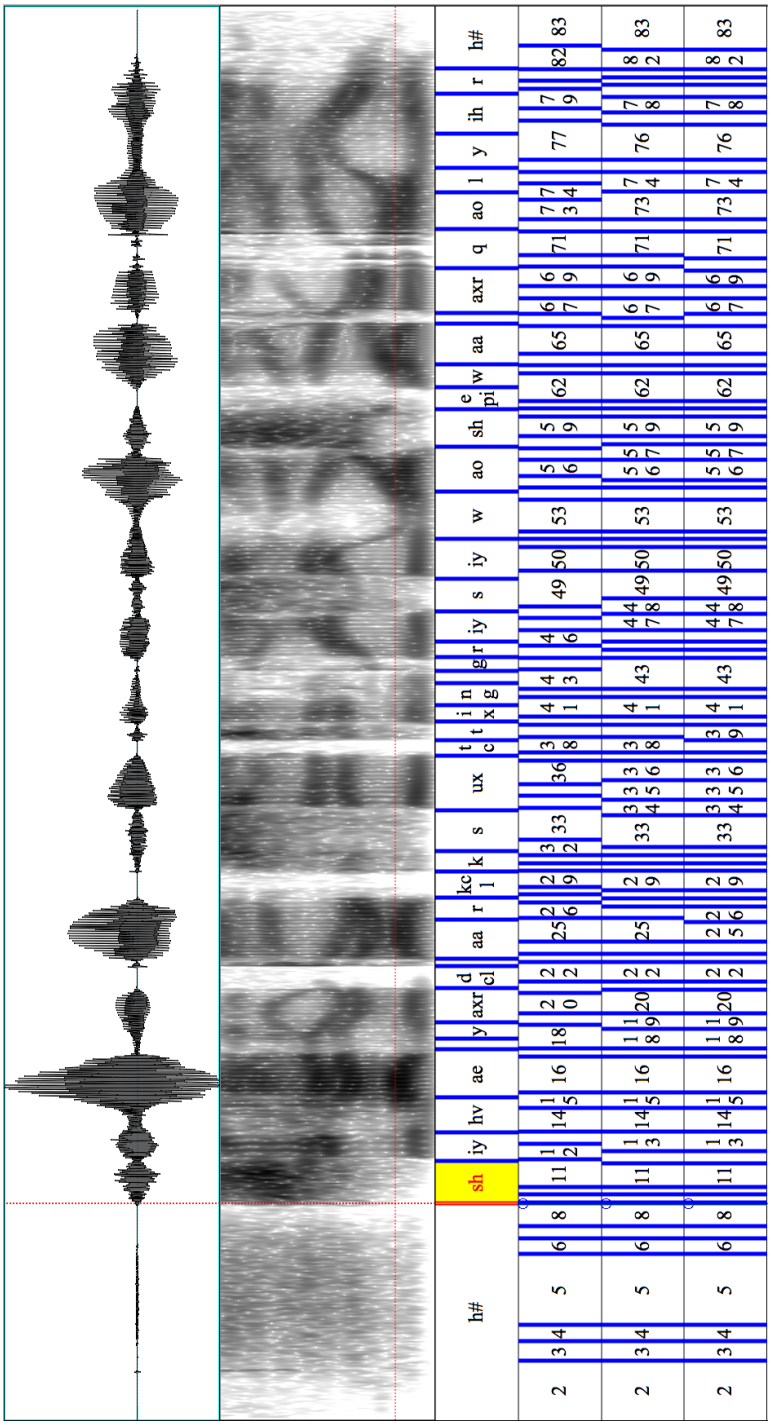

Figure 23: Segmentation using cochlea ZCPA as input. The numbered labellings corresponds to features generated using (from the top) 10, 20 and 40 channels. The features are extracted with max frequency of 8000 Hz, 60 frequency bins, window size 100 ms and window shift 15 ms.

4.3 ANN Segmentation

Figure 24 shows the values at the output node of the boundary class for one utterance in the TIMIT test set. This is a great example of a common problem with ANN: too many hidden nodes will result in large output values. In fact the values input to the sigmoid function had magnitudes of the order of 10^5 for the case of 100 hidden nodes.

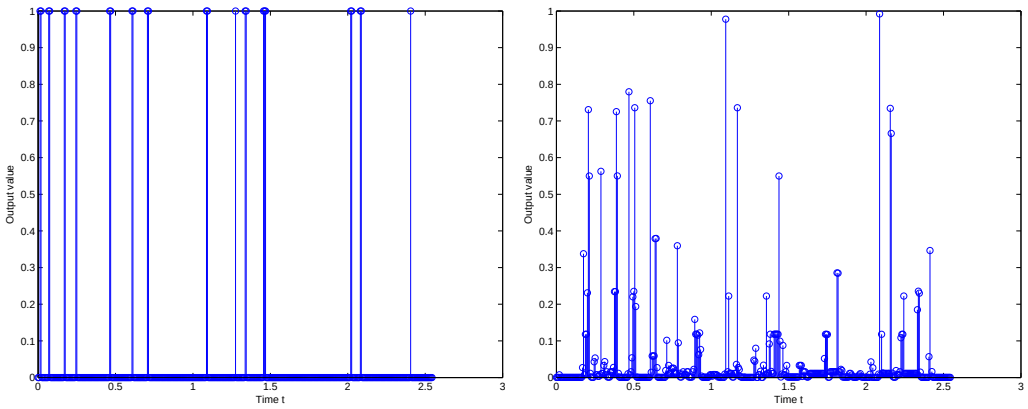

Figure 24: Values at output node using 100 hidden nodes (left) and 50 hidden nodes (right).

The Figures 25 and 26 shows segmentation results for four data sets: 10 channels+Δ+$\Delta\Delta$, 20 channels+Δ, 20 channels+Δ+$\Delta\Delta$, 40 channels+Δ all with a decimation factor of 100. The classification is done by selecting the class that had the maximum output value. The figures are good examples of the problem with succeeding frames all being classified as boundaries. This is of course expected and quite natural, as succeeding frames will be correlated.

In the Figures 27 and 28 the boundaries have been divided into 4 classes: voiced-voiced (v-v), voiced-unvoiced (v-u), unvoiced-voiced (u-v) and unvoiced-unvoiced (u-u). Also, the decision was performed by comparing the maximum of the four output nodes to a threshold, while the 5th output node corresponding to the non-boundary (nb) was ignored. The number of segments could then be controlled by adjusting the threshold. Also a penalty for succeeding boundaries was inserted, something that is shown clearly in the figures.

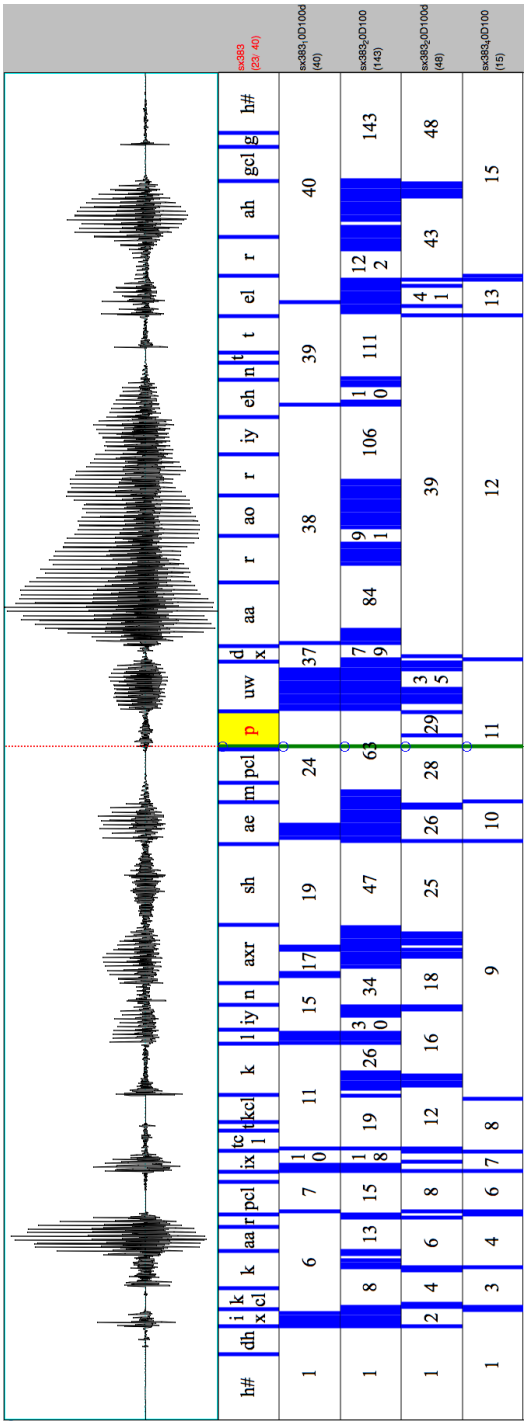

Figure 25: ANN Segmentation viewed in Praat. Data sets from top: manual labeling, 10 channels+Δ+ΔΔ, 20 channels+Δ, 20 channels+Δ+ΔΔ, 40 channels+Δ all with a decimation factor of 100. 20 hidden nodes. Maximum of output nodes used for deciding.

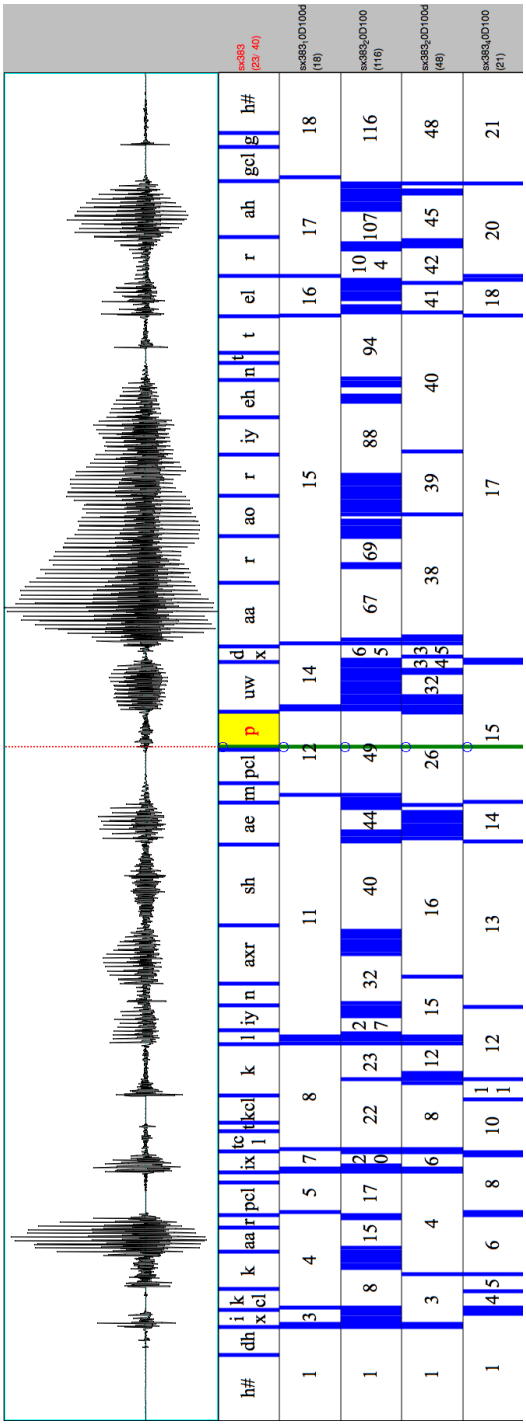

Figure 26: ANN Segmentation viewed in Praat. Data sets from top: manual labeling, 10 channels+Δ+ΔΔ, 20 channels+Δ, 20 channels+Δ+ΔΔ, 40 channels+Δ all with a decimation factor of 100. 50 hidden nodes. Maximum of output nodes used for deciding.

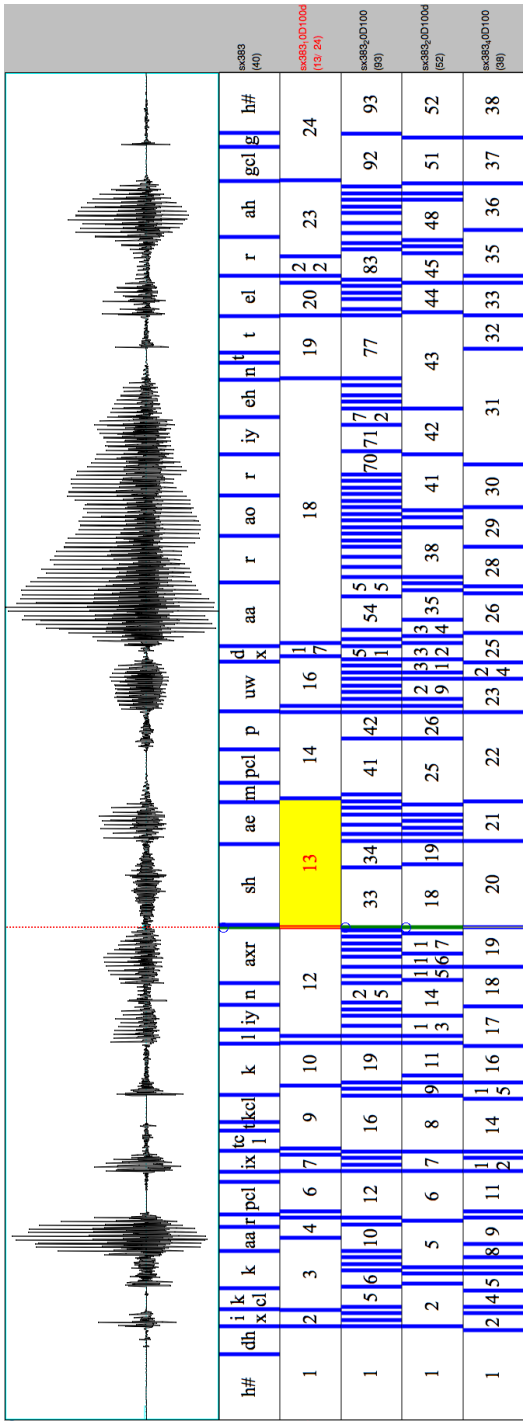

Figure 27: ANN Segmentation viewed in Praat. Data sets from top: manual labeling, 10 channels+Δ+ΔΔ, 20 channels+Δ, 20 channels+Δ+ΔΔ, 40 channels+Δ. 50 hidden nodes. Output of boundary node larger than threshold used for deciding. Penalty for succeeding boundaries added.

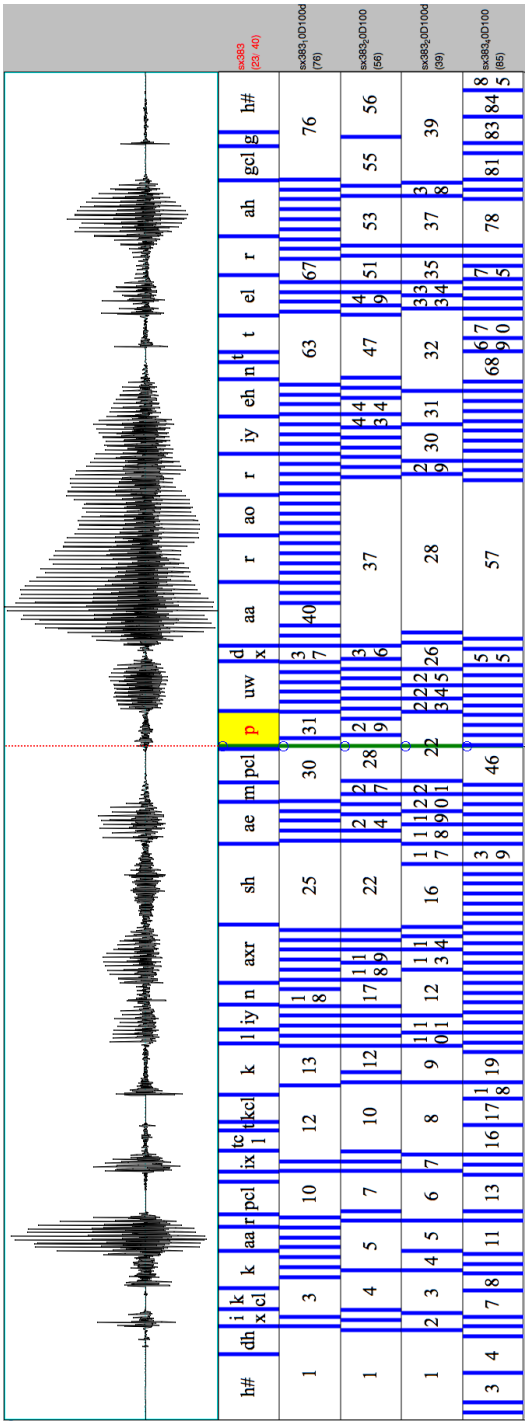

Figure 28: ANN Segmentation viewed in Praat. Data sets from top: manual labeling, 10 channels+Δ+ΔΔ, 20 channels+Δ, 20 channels+Δ+ΔΔ, 40 channels+Δ. 50 hidden nodes, 5 output nodes. Maximum of output nodes 2-5 larger than threshold used for deciding. Penalty for succeeding boundaries added.

One of the most interesting results in the experiments with 5 output nodes, is the confusion matrices shown in Table 3. These show the portion of vectors of a class (row) being classified as another class (column), i.e. element (i, j) gives the share of vectors from class i being classified as class j. Especially notice how well the classifier does for the (u-v) class. Although the rest of the results are not very good.

-	nb	u-u	v-u	u-v	v-v
nb	92.6	0.8	0.2	4.4	2.0
u-u	95.0	0.6	0.9	2.9	0.7
v-u	98.6	0.2	0.2	0.6	0.4
u-v	60.3	17.9	0.3	19.6	1.9
v-v	87.9	3.6	0.1	7.0	1.4

	nb	u-u	v-u	u-v	v-v
nb	92.5	0.1	1.9	4.9	0.6
u-u	93.6	0.0	4.7	1.1	0.5
v-u	98.5	0.0	0.9	0.6	0.0
u-v	77.7	0.1	2.3	19.5	0.3
v-v	94.3	0.0	0.5	5.0	0.1

	nb	u-u	v-u	u-v	v-v
nb	90.8	2.6	0.2	4.9	1.6
u-u	79.3	11.0	0.2	8.6	1.0
v-u	78.7	3.3	1.1	2.7	14.1
u-v	43.8	5.6	0.1	49.7	0.8
v-v	79.4	1.8	0.3	15.5	3.1

	nb	u-u	v-u	u-v	v-v
nb	80.6	1.5	0.7	16.2	1.0
u-u	81.2	0.3	0.0	18.6	0.0
v-u	94.8	0.3	0.0	4.8	0.1
u-v	44.0	1.5	0.1	54.0	0.5
v-v	77.7	1.0	0.2	20.6	0.6

Table 3: Confusion matrix for (from left to right, top to bottom) 10 channels+Δ + $\Delta\Delta$, 20 channels+Δ, 20 channels+Δ + $\Delta\Delta$,40 channels+Δ. The classes: No boundary (nb), unvoiced-unvoiced (u-u), voiced-unvoiced (v-u), unvoiced-voiced (u-v) and voiced-voiced (v-v). For each manual boundary three frames are said to be of that boundary class.

5 Discussion

One of the largest difficulties when experimenting with automatic segmentation is how to evaluate the results. Both because there is no right solution to the problem, and because a good segmentation for one task does not need to be a good segmentation for another task. Hence, without knowing in detail what kind of errors that are crucial for the next part of the system, it is hard to tell if one segmentation is better than another. However, it is possible to see if the resulting segmentation corresponds to the speech signal at all. If the segmentation agrees with the manual labelling on many of the boundaries, it is probably a good segmentation.

5.1 Cochlea Features

When trying to evaluate the results of a given set of parameters, three criteria have been used: firstly, the features have a tendency to being noisy, a behaviour it is beneficial to suppress. Hence, features with less noisy behaviour were desired. Secondly, a reasonable distribution of energy in the frequency domain is important in order to avoid losing essential information. Finally and most importantly, a subjective evaluation of the resulting segmentation has been performed.

The first criterion has been a big problem, since a large degree of abrupt changes gives a bad segmentation when using the acoustic algorithm. Hence, much time and effort have been used to get smoother features. The main reason for this being so difficult is because one of the main properties of the features is in fact that they give a large spike for onsets. Of the two approaches used to suppress this noisiness, the LP filtering seem to give slightly better segmentation results than the windowing operation given in Algorithm 1. This might be because the windowing operation was not as accurate in removing the high frequencies of the features, or because the right setting with window shift and window size was not found. Nevertheless, the smoothing algorithm did not seem to give features with better properties than a LP filtering. Hence, it is unlikely that the window operation can give any better results than the LP filtering.

Another problem with the features was that the distribution of energy in the frequency domain seemed strange in several of the experiments. Originally it appeared to be too little energy both in the frequencies below 1000 Hz and above 3500 Hz (see Figure 11 for an example). This was improved by

normalising the filter bank and by increase the scaling of the input to MHC. However, it is still a problem that higher frequencies are being ignored. A good example of this is segment number 16 in Figure 17 on page 35. This is a "k" followed by an "s", which should be segmented into three segments ("k"-closure, "k"-burst and "s"). However, there are none or very little energy in frequencies below 7000 Hz (see decimated cochleagram in Figure 15 on page 33) during this part of the signal. This results in only one segment.

One parameter that was hard to see the actual impact of, was the number of channels. However, it seems that the optimal value of this parameter is dependent on the other parameters and operations performed on the features. Hence, this parameter should be optimised after a given feature extraction setting has been chosen.

Among all the different parameter setups for the feature extraction process it is hard to tell which one that is the best for acoustic segmentation. Especially since the knowledge of the details of the detectors is unknown. One of the parameters with highest impact on the result, is the scaling of the input to the MHC. When using the original 80 suggested in the Auditory Toolbox documentation, the activity of the output was very low. However, a value of about 500 gave much better results (compare Figure 13 and figure 15 on page 31). Further, a normalisation of each channel with regard to the energy of its unit sample response, gave somewhat better results than the original normalisation. This can be explained by that the latter gives higher energy in the channels with wide bandwidth and high frequency, while much of the information in a speech signal is in the lower frequency channels. Hence, by normalising with regard to unit step frequency response, the information contained in the lower frequencies of the speech signal will have more impact in the resulting segmentation than the information in the higher frequencies. By looking at the Figures 11 and 15, it is evident that the latter is better. It has a large degree of energy in the low frequencies, but still has some information left in the higher frequencies. This is confirmed by investigating Figure 17, which is a pretty good segmentation, in fact it is the best that were produced using the cochlea features as input to the acoustic segmentation algorithm (not including the ZCPA experiments).

5.2 Cochlea ZCPA Features

Because of time constraints, an optimising of the parameters of the cochlea ZCPA feature extraction was not performed at all. The features were therefore only tried for a couple of chosen parameter settings. Hence, the results

from these experiments can only be used as an indication for whether these features are usable or not.

When looking at the resulting segmentations in Figure 23 on page 40 it seems that a high degree of the manual segment boundaries are hit, meaning that they have an automatically generated boundary close by. Also for these features the number of channels seems to have a small impact on the resulting segmentation. However, a pattern is hard to see other than that 20 and 40 channels gives results that are similar compared to the result for 10 channels. This is probably because 10 channels give less data for the frequency histogram construction.

One thing is clear from these experiments: the results given by the cochlea ZCPA are the best results generated in this work. It is definitively promising results indicating that a more thorough research on these features are in order.

5.3 ANN Segmentation

The idea of trying to use classification as an approach for segmentation came from plots like those in Figure 21, where the boundaries seem to have characteristic feature vectors. It gave some promising results but could not really compare with the acoustic segmentation algorithm.

When looking at the segmentation based on the ANN, it seems like it does a good job for some of the boundaries. However, it is very unstable in the way that it leaves some parts of the utterance unsegmented, while other parts are segmented into very small pieces. This indicates that it might be impossible to utilise the information this kind of segmentation provides. Nevertheless, when studying the confusion matrices (Table 3 on page 46) it does not look all bad. First, remember that the resulting segmentation is only depended on non-boundary/boundary, so a misclassification between the four boundary classes ("u-u", "u-v", "v-u", "v-v") ends up being right. Hence, the unvoiced-voiced boundary vectors are in reality only misclassified in 43.8% of the cases for the data set with 20 channels$+\Delta+\Delta\Delta$. In addition, remember that for each manual boundary there are three frames said to be a boundary. This means that with only 33.3% correct, it is still possible to cover all, but very unlikely though. Of course, it is possible that many of these boundaries are from parts that are very rich on segments, but it does not seem that way from the segmentation in Figure 28 on page 45.

A hypothesis from this might be that the ANN approach performs more sta-

ble on some specific subword units than the acoustic variant. If that is true, the ANN approach is something that is possible to utilise in the corresponding detector in the SIRKUS recogniser. However, further experiments have to be done in order to confirm this.

5.4 Future Work

The cochlea features contain one important peculiar property: They produce a high spike to mark onsets. Since this results in high distortion in the acoustic segmentation, it might be that the algorithm is not the optimal approach for using the cochlea features to segment speech. It could however be possible to change the algorithm to better handle the cochlea features. Since the big problem is that the features are not stationary for a stationary part of a signal (see Figure 5 on page 14), a better adapted approximation trajectory could be beneficial.

The use of the cochlea ZCPA features seemed to give good results. A deeper look into these features would certainly be of interest. First of all an optimisation of the feature extraction process must be performed. Especially, the window length, number of histogram bins and number of channels need to be coordinated to ensure that each bin gets enough data.

A problem with the ANN approach is that succeeding frames are all being detected as boundaries. In this work a penalty function has been inserted to give some memory to the process. The problem with that solution is that it might not always be the first boundary that is the right. Imagine as an example 5 succeeding frames being detected as boundaries, and that only the middle on in fact is a boundary. By inserting the penalty function, this might result in frame number 1 and 5 being detected, instead of only number 3. Hence, if the approach is to be used for segmentation, a better solution to the problem should be developed. For example a post processing step finding the optimum segmentation based on some kind of cost function could be utilised.

Something that might be worth trying is to perform an evaluation of how each of the segmentation approaches are performing for different phonemes. This can be done by comparing the segmentation result with the manual labels and calculate the success rate of segments belonging to the phoneme in question. These results can be valuable when deciding what approach that is best for each of the detectors.

6 Conclusion

It is evident from the results shown in this work, that the cochlea features contain a great deal of usable information. The problem however, seems to be how to utilise this information when performing automatic segmentation. Of the approaches that have been presented, the cochlea ZCPA features gave the best results. They are definitively a subject for further exploration. The question is however if the features give any new information that is not contained in the frequently used MFCCs. This is of course impossible to answer based on the results from this work, since MFCCs have not been used at all. Hence, a comparative study should be performed before any conclusion regarding this question is made. Of course, the ZCPA features need an optimisation before this is done.

Since the SIRKUS recogniser consists of several parallel detectors trying to detect if a certain phonological feature is present, it might be that the cochlea features are better than the MFCCs for some of the detectors. It is therefore possible that the features can be used in the SIRKUS project, despite the fact that they seem to perform slightly poorer than the MFCCs in general.

Further, the features seem to contain a great deal of information not only needed for segmentation, but also for recognition in general. Much of the information seems to be similar to that of the MFCCs. They are both containing information regarding energy in different frequency bands. However, the cochlea features are different in that they emphasise onsets in a frequency band.

This work can not be used to reject the use of cochlea features in speech recognition or automatic segmentation. Neither can it be used to argue that they are better than alternative features. Hence, it is necessary to perform more research before making a final conclusion. However, the features give some promising results which indicates that they should be further explored. Especially promising are the results from the cochlea ZCPA, which definitively should be a subject of further investigation.

References

[1] R.A. Obrecht. A new statistical approach for the automatic segmentation of continuous speech. *IEEE Trans. Acoust. Speech Signal Processing*, Jan 1988.

[2] Abhinav Sethy and Shrikanth Narayanan. Refined speech segmentation for concatenative speech synthesis. In *Proc. ICSLP'02*, pages 149–152, Denver, Sept. 2002.

[3] J.R. Glass and V.W. Zue. Multi-level acoustic segmentation of continuous speech. In *Proc. ICASSP'88*, pages 429–432, 1988.

[4] Torbjørn Svendsen and Frank K. Soong. On the automatic segmentation of speech signals. *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '87*, 1987.

[5] Torbjørn Svendsen. Articulatory features and segmental information for automatic speech recognition. ESF workshop, November 2007.

[6] A.V. Brandt. Detecting and estimating parameter jumps using ladder algorithms and likelihood ratio tests. In *Proc. ICASSP'83*, 1983.

[7] Vivek Tyagi, Hervé Bourlard, and Christian Wellekens. On variable-scale piecewise stationary spectral analysis of speech signals for asr. *Speech Communication*, 48(9):1182–1191, Sept. 2006.

[8] Ray Meddis. Implementation details of a computation model of the inner hair-cell/auditory nerve synapse. *Journal of Acoustic Society of America*, 1990.

[9] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett. DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. *NASA STI/Recon Technical Report N*, 93:27403–+, February 1993.

[10] R.D. Patterson, K. Robinson, J. Holdsworth, D. McKeown, C.Zhang, and M.Allerhand. Complex sounds and auditory images. *Auditory Physiology and Perception*, 1992.

[11] Ray Meddis. Simulation of auditory-neural transduction:further studies. *Journal of Acoustic Society of America*, 1988.

[12] Bojana Gajic. Auditory based methods for robust speech feature extraction. *Telektronikk*, pages 45–58, 2003.

[13] Richard P. Lippman. An introduction to computing with neural nets. *IEE ASSP Magazine april*, 1987.

[14] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, Third Edition*. Academic Press, February 2006.

[15] Alan H. Kramer and A. Sangiovanni-Vincentelli. Efficient parallel learning algorithms for neural networks. pages 40–48, 1989.