Speech Feature Enhancement using Particle Filters with Class-Based Phoneme Models

Dominic Heger

November 26, 2007

Advisors: Matthias Wölfel, Prof. Alex Waibel

Institut für Theoretische Informatik,
Universität Karlsruhe (TH), Germany

Interactive Systems Laboratories,
Carnegie Mellon University, USA

Abstract

Robust speech recognition in noisy environments can still be seen as a widely unsolved problem, while being enormously relevant in practice. For instance, state of the art systems would likely perform unsatisfactory as speech interface to a ticket machine in a train station or for the task of answering your emails while driving in a car - because of the presence of environmental noise.

In recent years *particle filter* based methods have shown significant performance improvements in this field of research. One of the most crucial subjects in particle filter design for clean speech estimation is, to have an adequate representation of speech (speech model) on which the particle weight calculation is based. Typically, a general, time invariant and phoneme independent speech model is used in this place.

First attempts of research into using a time varying, phoneme specific speech model have already been performed by F. Faubel and M. Wölfel at Universität Karlsruhe (TH) and Carnegie Mellon University, USA which have shown notable gains in word error rate.

This project thesis introduces new approaches to further improve the particle filter performance by *phoneme specific dynamically time varying speech models.* These speech models are based on three different concepts:

1. Building knowledge-driven, heuristic based phoneme clusters
2. Building data-driven, unsupervised hierarchical clustering based phoneme clusters
3. Considering confusability to build phoneme classes

Deutsche Zusammenfassung

Robuste Spracherkennung in geräuschbehafteten Umgebungen kann heute immernoch als ein größtenteils ungelöstes Problem betrachtet werden, obwohl es außeroderdentlich praxisrelevant ist. Beispielsweise ist zu erwarten, dass aktuelle Spracherkennungssysteme als Sprachschnittstelle für einen Fahrkartenautomaten im Bahnhof unzureichende Ergebnisse liefern. Der Grund dafür ist das Vorhandensein von Hintergrundgeräuschen.

In den letzten Jahren erreichten auf Partikelfiltern basierende Verfahren eine signifikante Verbessungungen der Erkennungsleistung auf diesem Forschungsgebiet. Einer der entscheidendsten Faktoren im Entwurf von Partikelfiltern ist, eine geeignete Repräsentation für unverrauschte Sprache (Sprachmodell) zu finden, auf der die Berechnung der Gewichtsfaktoren der Partikel beruht. Typischerweise wird zu diesem Zweck ein allgemeines, zeitinvariantes und phonemunabhängiges Sprachmodell verwendet.

Erste Forschungen ein zeitveränderliches und phonemspezifisches Sprachmodell zu verwenden, wurden bereits von F. Faubel und M. Wölfel an der Universität Karlsruhe (TH) und Carnegie Mellon University, USA unternommen, die deutliche Verbesserungen der Wortfehlerrate zeigten.

Diese Studienarbeit stellt neue Ansätze vor, um die Leistung von Partikelfiltern durch phonemspezifische, zeitveränderliche Sprachmodelle weiter zu verbessern. Diese Modelle basieren auf drei unterschiedlichen Konzepten:

1. Erzeugung von Phonemklassen, durch wissens- und heuristikbasierende Verfahren
2. Erzeugung von Phonemklassen, durch datengetriebene, unüberwachte, hierarchische Clusterverfahren
3. Betrachtung von Verwechselbarkeit, als Mittel zur Bildung von Phonemklassen

Eine Reihe von Verfahren wurde implementiert und anhand von Spracherkennungsexperimenten evaluiert. Trotz deutlicher Variabilitäten in den gemessenen Ergebnissen, konnte in vielen Fällen die Wortfehlerrate durch eines der neu vorgeschlagenen Konzepte, im Vergleich zu den traditionellen Methoden, weiter verbessert werden.

Contents

1 Introduction

The family of algorithms called *Particle Filters* (PF)s (a.k.a. *Sequential Monte Carlo Methods*) is well known in various fields of research including Computer Science, Signal Processing, Statistics and Econometrics. The reason is, that it is becoming more and more important to cope with non-linearity and non-Gaussinanity of dynamic processes that have to be modeled in those fields. Furthermore, the performance of today's computers is sufficient to use them in real-time applications. In research and practice PFs have become most popular in different kinds of tracking tasks, e.g. persuing the position of airplanes on a radar [GSS93] or the movement of persons in a video.

In recent times, they also were applied in the field of automatic speech recognition (ASR), where they can be used for the enhancement of speech features in noisy environments. It is a generally unsolved problem, that the performance of ASR systems decrease rapidly, when operating in an environment which is not completely silent, apart from the speaker's voice. Obviously, this premise cannot hold in many real world applications. An analysis of the problematic effects occurring, when signals are contaminated by noise, can be found in chapter 4 of [Mor96].

Since ASR can be seen as a pattern recognition issue, the problem, which has to be addressed is, to overcome the discrepancy between the environment, in which the acoustic models of the ASR system have been trained and the environmental conditions in which the ASR system has to decode the (noisy) speech features, afterwards. This mismatch results in a drastic decrease of recognition performance. Over the years, various kinds of techniques have been suggested, which try to overcome this drawback.

1.1 Basic components of a statistical speech recognition system

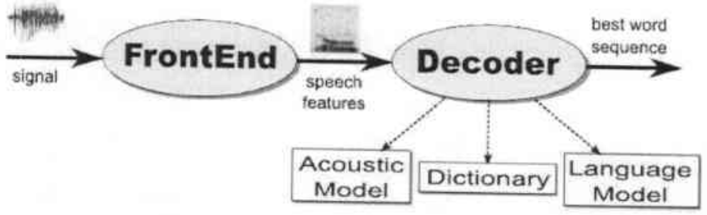


Figure 1: Basic ASR system components

Figure 1 shows the typical components of a statistical ASR system. First, the analog singal is pre-processed in a *front-end* step, which digitalizes the signal, reduces the dimension and extracts relevant speech features. These features are decoded in a following recognition step, which outputs the word sequence with the best probability. The *decoder* basically works on three information sources: First, *acoustic models*, which provide a conditional probability of speech features for a given phonetic unit of speech. Second, a *dictionary*, which maps the phonetic units to words, which can be recognized and third, a *language model* which gives likelihoods for a sequence of words in a specific language.

1.2 Acoustic models - Gaussian Mixture Models

Most state of the art ASR systems represent the acoustic units of speech by Gaussian mixture models (GMM)s, which are actually the linear combination of M single Gaussian distributions:

$$p(x) = \sum_{i=1}^{M} c_i \mathcal{N}(x|\mu_i, \Sigma_i), \text{ where } \sum_{i=1}^{M} c_i = 1,\ 0 \leq c_i \leq 1 \qquad (1.1)$$

A single multivariate Gaussian distribution of dimension n is defined as

$$\mathcal{N}(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu)}$$

Where μ is the mean vector (centroid), Σ is the covariance matrix and $|\cdot|$ is the determinant. In ASR, usually, diagonal covariance matrices are used for modeling acoustic speech.

Gaussian mixtures are known to be universal approximators, which means that any probability distribution can be modeled by a (not necessarily finite) Gaussian mixture.

1.3 Noise compensation techniques

The term 'stationary noise' means, that the background noise is assumed to be a stochastic process, whose probability distribution is fixed over time (e.g. means and variances are set to be time constant). This assumption of stationary noise may hold for white noise, which is a random signal with equal energy for each frequency in the power spectrum, and for colored noise, which also has a fixed power distribution (e.g. pink noise's spectral density is proportional to the reciprocal of the frequency[1]). However, most of the noise we have to face under real world conditions, consists of important parts that vary over time (imagine a car driving past the road, or voices of an auditory in the background). Historically, the first noise compensation techniques tried to cope with *stationary noise*, since it is much easier to handle compared to *non-stationary noise*.

Over the time many different noise compensation approaches have been proposed (e.g. model adaption techniques, or hidden Markov model - decomposition). The technique we use is *speech feature enhancement*. This method tries to clean and emphasize classification relevant characteristics of a noisy speech signal with the goal to retrieve features, which are undistorted, so that the ASR-decoder is able to process them properly.

It is possible to perform speech feature enhancement in an independent pre-processing step (offering an easy potential for parallel implementation), or within the front-end of the ASR system during feature extraction. In both cases it is not necessary to modify the decoding stage and it does not require any changes to the acoustic models of the ASR system.

Traditionally, techniques like *spectral subtraction* or *Wiener filtering* were used for speech feature enhancement. However these methods basically only work for stationary noise compensation. A detailed analysis of traditional methods and further extensions to these systems can be found in [Eph92].

[1] $S(f) = \frac{1}{f}$, where f is the frequency.

Considering the problem of speech feature enhancement as Bayesian parameter estimation, which is also not limited to stationary processes, makes it possible to apply a series of statistical algorithms to estimate the state of dynamical systems. First attempts of research in this field, assumed speech as an autoregressive (AR) linear process, polluted by white noise. Therefore [PB87] proposed to estimate the linear prediction (LP) coefficients and noise variances and then apply a *Kalman filter* (KF) to get an estimate of the clean signal. But there are fundamental problems using LP, since it is known to unsuitably model voiced speech and medium or high pitched voices. Moreover LP coefficients are unstable, which means small changes in the coefficients may not lead to small changes in the speech signal. Furthermore a KF assumes the relationship between the observations and the inner state to be linear and Gaussian, which is not true in practice. That is why [DGW00] proposed to use a *time varying partial correlation model*, and in addition to that to replace the KF by a *particle filter* (PF) (operating in time domain).

Instead of tracking clean speech, [Kim98] proposed to see the problem from the opposite direction and use noise as the state variable, which means that the noise spectrum is 'contaminated' by clean speech. All of the filtering methods mentioned before, do neither operate in the log spectral, nor in the Mel frequency domain. That is why [YN02] presented a PF, operating in the log Mel spectral domain, where the characteristics of the human auditory system are regarded. A PF using the technique of *sequential importance resampling* (SIR), called *Bayesian Bootstrap filter*, was first proposed by Gordon et al. [GSS93] and is basically the approach used for the experiments of this thesis. The advantages of particle filtering compared to other filtering methods are:

- Noise can be *non-stationary*.
- It can cope with the *multi-modality of the proposal density* (i.e. can have more than one maximum).
- It can cope with the *non-linearity* and *non-Gaussianity* of the observation transition.
- PFs are *computational efficient* compared to alternative algorithms (e.g. EKF or HMM-decomposition).

1.4 Basic work

In prior works [Fau06, FW06] a particle filter for speech feature enhancement has already been developed and implemented by our research group at the Institut für Theoretische Informatik, Universität Karlsruhe (TH), Germany and the Interactive Systems Laboratories, Carnegie Mellon University, Pittsburgh, USA. It is part of the *Janus Recognition Tooklit* (JRTk), which is developed and maintained cooperatively by both institutions.

The PF algorithm mainly follows the approach of Singh and Raj [SR03]. Furthermore, a series of refinements to the original approach have been developed to further improve the word recognition accuracy and stability of the PF:

In practice it can happen, that the PF algorithm overestimates a noise hypothesis, so that it exceeds the observed contaminated spectrum. In this case the relationship between contaminate speech, clean speech and noise cannot be calculated in the log Mel spectrum, because of negative logarithm, as shown in the next chapter. Assigning zero weight to such a hypothesis can lead to a decimation of the particle population until its complete annihilation. [FW07] tackled the problem by a so-called *fast acceptance test* and reinitialization procedure.

The second major performance improvement of the particle filter is, to replace the commonly used *vector Taylor series* (VTS) [MRS96] by a new *statistical inference approach* [FW07] to infer clean speech after the noise estimation.

1.5 Pre-processing stage

A speech feature enhancement stage should be placed as close as possible to the feature domain in which the decoder of the ASR system operates, to achieve the best effects. That is why otherwise irrelevant parts of the signal may be cleaned (e.g. a MMSE estimation in the spectral domain does not lead to a MMSE estimation in the log spectral domain) and furthermore there is a general problem that speech feature enhancement may not directly result in a better recognition performance (in terms of word error rate).

Figure 2 shows the pre-processing stage of the speech recognizer (JRTk) used for this project thesis.

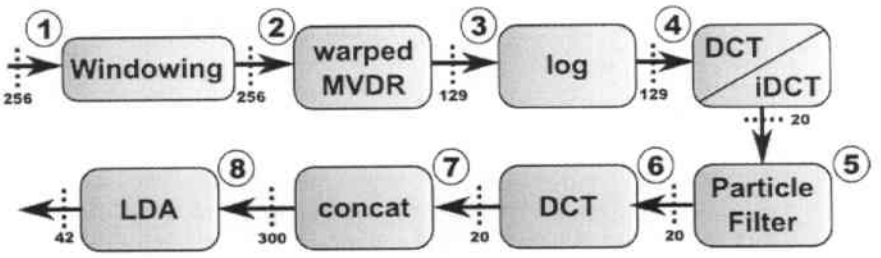


Figure 2: Feature extraction stage

1. The inputed acoustic signal is sampled by 16 kHz, 8 bit. This time domain signal is cut into frames, each of 10 ms length, using a 16 ms *Hamming window*.
2. Instead of the prominent Fourier transformation, we retrieve spectral coefficients, from the 256 samples, by the warped and scaled *minimum variance distortionless response* (MVDR) spectral envelope [WM05]. This results in a 129 dimensional estimation of the power spectrum. It has been shown that spectral envelopes operate more robustly in noisy environments, since they overcome the equal weighting of spectral valleys and peaks by representing energy rich regions with more detailed information than low energy regions, where noise is mainly present. Furthermore, MVDR outperforms *linear prediction* (LP), which is known to model voiced speech and high pitched voices not accurately. To mimic the human auditory system, which percepts frequencies and loudness logarithmically, *Mel-frequency* is approximated by warping the MVDR.

3. By calculating the componentwise logarithm (log) we obtain the *log Mel power spectrum*.

4. The application of *discrete cosine transform* (DCT) takes the spectum into the commonly so-called *cepstrum*. A dimensional reduction to 20 spectral bins is performed by cutting off high cepstral coefficients. This procedure is commonly known as *liftering*. Transforming back again, using a 20 dimensional matrix, results in a smoothing effect of the *spectrum*.

5. The *particle filter*, which is discussed in the next sections, performs the speech feature enhancement.

6. *DCT*, yet without truncation, is applied to operate in the cepstrum, again.

7. To be able to regard variations to adjacent samples in the feature space, for each frame the 7 prior and 7 following spectra (samples) are combinated to form a 300 dimensional vector.

8. Finally, *linear discriminant analysis* (LDA) is applied. This transformation decorelates and arranges the features by their discriminability, while minimizing the variances within each class (i.e. codebook) and maximizing the variance between the classes. By cutting off low order coefficients during the transformation, a further reduction to 42 dimensions is achieved.

Thus, the PF works in the log Mel spectral domain. Maybe it would be appropriate to do speech feature enhancement in the cepstral domain or after the LDA transformation, but there are mathematical problems to maintain the relationship between noise corrupted speech, clean speech and noise throughout the DCT and LDA steps. [Fau06] points out the problems in detail.

1.6 Phones and Phonemes

Phonologists accurately distinguish between *phones* and *phonemes*. *Phones* are all the single acoustic units, which can be distinguished in speech. The *international phonetic alphabet* (IPA) [Ass] aims to list phones of any spoken language. *Phonemes*, however, are the smallest units of speech, which can effect the semantic meaning of a word (e.g. first sound of the words 'rip' and 'lip').

Engineers, seeing these differences from a more technical perspective, tend to mix the terms, since ASR systems only use acoustic models for recognition of the phonetic units, which have to be mapped to words. Throughout this thesis I use the term *phoneme*, although it might not be correct in every situation. Appendix A lists the phonemes and corresponding word examples used for clustering and particle filtering in this work.

2 Speech Feature Enhancement using Particle Filters

This section gives a short introduction into particle filtering theory, which is based on Bayesian tracking, and presents our particle filtering algorithm for clean speech estimation.

2.1 Particle Filter Theory

Many real world processes can be modeled as time-discrete stochastic *dynamic systems* [RN03]. At time step t the output or *observation* y_t of such a system depends on the *current input* u_t, and an *inner state* x_t (which can be seen as a hidden memory). y_t, x_t and u_t can be regarded as n dimensional random vectors.

The evolution of the inner state x_t is described by the so called (dynamic) *state model*. The (static) *measurement model* describes the output y_t upon a given point of time t, inner state x_t and current input u_t.

Usually the state transition probability can be modeled as a (1st-order) *Markov chain*:

$$p(x_{t+1}|x_0, x_1, ..., x_t) = p(x_{t+1}|x_t)$$

The Markovian assumption is basically no restriction, since it can be shown, that every higher order Markov chain can be transformed into a 1st-order Markov chain (by an increase of state space dimensionality).

Figure 3 shows a typical block diagram of a dynamical Markovian system.

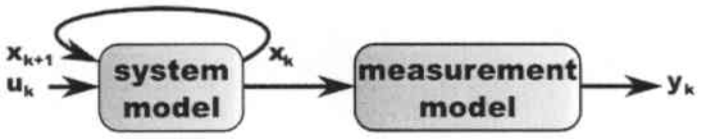


Figure 3: Dynamic Markovian system

The estimation of the inner state x_t, based on the observations y from time step 0 to time step t (the notation $y_{0:t}$ is used in the following), is commonly known as *sequential Bayesian filtering*. *Particle filters* (PF)s are simulation filters, where the probability density mass is modeled by samples from the state space - the so-called *particles*, outputting the *posteriori probability density* of the system state.

Sequential estimations from this hidden system state's probability density $p(x_t|y_{0:t})$ are obtained for each time step by recursively computing, when a new observation is received (i.e. tracking).

Several different PF algorithms have been proposed. In [AMGC02], variants are discussed and compared to estimators, which perform optimal under certain restrictive conditions, like the Kalman filter (KF). Since we use a Bayesian Boostrap filter, the term particle filtering refers to a PF algorithm using the technique of sequential sampling importance resampling (SIR), throughout this thesis.

2.1.1 Bayesian tracking algorithm

A general Bayesian tracking algorithm can be regarded as two simple steps:

1. *Prediction* from the previous density (at $t-1$) one time step ahead to the current time step (t) via the *state transition* density (time update), which can be seen as the evolution (or simulation) of the system:

$$p(x_t|y_{1:t-1}) = \int_{-\infty}^{\infty} p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1} \tag{2.1}$$

   Since this equation is recursive, the system state has to be initially (at $t=0$) estimated by an *a-priori probability* $p(x_0|y_0) = p(x_0)$.

2. A *filtering* step (measurement update), which uses the latest observation to adjust the probability density using Bayes' theorem

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t|y_{t-1})} \tag{2.2}$$

   where the denominator (normalizing constant) can be computed by margialisation of the numerator:

$$p(y_t|y_{t-1}) = \int_{-\infty}^{\infty} p(y_t|x_t)p(x_t|y_{1:t-1})dx_t$$

It can be shown, that the optimal solution to the tracking problem, which is the estimation of the system state, that has a minimal distance to the optimal estimate (*minimum mean square error* estimation) is

$$\mathbb{E}_{p(x_t|y_{1:t})}[x_t|y_{1:t}] = \int_{-\infty}^{\infty} x_t \cdot p(x_t|y_{1:t})dx_t \tag{2.3}$$

where $\mathbb{E}[\cdot]$ is the expectation value. As we are interested in a functional dependency of the estimated noise, corrupted speech and clean speech in our algorithm in the next section, (2.3) can be generalized to

$$\mathbb{E}_{p(x_t|y_{1:t})}[h(x_t)|y_{1:t}] = \int_{-\infty}^{\infty} h(x_t) \cdot p(x_t|y_{1:t})dx_t \tag{2.4}$$

2.1.2 Particle filtering

The Bayesian tracking algorithm derived in the prior section can be seen as the optimal solution of recursively calculating the posteriori density. But in general it cannot be computed analytically, because (2.1) and (2.2) are not necessarily linear. Applying further restrictions, allows to compute an optimal estimation. E.g. a *Kalman filter* (KF) delivers an optimal estimate under the restrictive condition that both, the state transition and the observation transition, are Gaussian and linear functions. But since the relationship between noise and speech is non-linear in nature, a KF is not an adequate algorithm for speech feature enhancement. Alternatively an *extended Kalman filter* (EKF) could be used, which locally linearizes the problem by a first order Taylor series approximation. However it is not mathematically justified to retrieve gains in recognition performance when using an EKF [Kim98].

Particle filters have the ability to cope with non-linear and non-Gaussian problems by approximating the probability distribution with discrete random measures, modeled N samples and weights (called particles):

$$\{x_t^{(j)}, w_t^{(j)}\}_{i=1}^{N}$$

These particles are relocated and weighted in each time step. Particle filtering belongs to the class of Monte Carlo algorithms, therefore it will not perform optimal in general, but as the number of particles goes to infinity, PFs approach the Bayesian optimal estimate.

So PFs approximate the continuous posteriori probability density (2.2) by its discretisized (empirical) counterpart:

$$p(x_t|y_{1:t}) \approx \sum_{j=1}^{N} w_t^{(j)} \delta(x_t - x_t^{(j)})$$

Where N is the number of the particles $x_t^{(j)}$ and $w_t^{(j)}$ their corresponding weights ($\sum_{j=1}^{N} w_t^{(j)} = 1$) as outlined in the next section. $\delta()$ denotes the Dirac distribution.

2.2 PF algorithm for clean speech estimation

A practical particle filtering algorithm for noise tracking and clean speech estimation can be seen as four steps:

1. *Evolution of noise hypotheses* (state transition of particles).
2. Rating the noise hypotheses *likelihood* (particle weights).
3. *Inference* of clean speech based on the noise estimation and observed spectral signal.
4. *Resampling* of noise hypotheses.

The steps 1 - 4 are repeated until each speech frame has been processed.

Note that in this chapter n_t is the state vector (instead of x_t in the preceding section), x_t denotes clean speech and y_t is the noise contaminated speech observation.

1. Initialization and Evolution of noise hypotheses

Initially the N noise hypotheses $n^{(j)}$ $(j = 1 \dots N)$ are sampled from a general probability distribution, which is learned from noise spectra for a specific noise (noise model).

Later $(t > 0)$ the *evolution* of particles (noise hypotheses) can be modeled by a kth-order *autoregressive progress*, as proposed in [RSS04],

$$\begin{aligned} n_t &= A_1 \cdot n_{t-1} + A_2 \cdot n_{t-2} + \cdots + A_k \cdot n_{t-k} + \epsilon_t \\ &= \underline{A} \cdot \underline{n}_{t-1} + \epsilon_t \end{aligned} \qquad (2.5)$$

where A_i are $d \times d$ state transition matrices, which are learned for a specific noise and n_t is a noise sample at time t. The *error term* ϵ_t is i.i.d. zero mean Gaussian (i.e. $\mathcal{N}(n_t; 0, \Sigma_{noise})$). It can be regarded to model parts of the system, which are not expressed by the autoregressive progress. $\underset{\sim}{A}$ denotes the $(k \cdot d) \times d$ by row concatenation of the k A_i matrices and $\underset{\sim}{n}_{t-1}$ denotes the vector by column concatenation of the last k d-dimensional noise samples n_i. This leads to the *noise transition probability* (prediction step)

$$p(n_{t+1}|n_t) = \mathcal{N}(n_{t+1}; \underset{\sim}{A} \cdot \underset{\sim}{n}_t, \Sigma_{noise})$$

Since it has been experienced, that the prediction of noise hypotheses does not significantly improve by using a model order greater than $k = 1$ and a much higher training effort would be necessary, we use a 1st-order model.

A new particle generation is obtained by drawing each particle of the next time step $n_t^{(j)}$ $(j = 1 \ldots N)$ from a so-called importance or *proposal density*, since drawing samples from the *posteriori density* $p(n_t|y_{1:t})$ is usually impossible. This concept, called sequential importance sampling (SIS), can be understood as drawing samples from regions of 'importance'.

When using a sampling importance resampling (SIR) particle filter algorithm, the proposal density of a new particle $n_t^{(j)}$ is its state transition probability $p(n_{t+1}|n_t^{(j)})$. This way, the particles are drawn from a density, which is independent from y_t, however the current observation y_t takes effect to the particle weighting (next step of this algorithm).

As mentioned before, overestimations of the noise hypotheses can cause severe decimation of the particle population, which is also analyzed in [HUS05]. A simple solution to work against the degeneracy would be, to use a very high number of particles, but this would suffer from high computational costs. Therefore we use the so-called *fast acceptance test*, which has been proposed by [FW07]:

While the condition $n_t^{(j)} < y_t$ is not satisfied for all spectral bins, or a certain number B of iterations have passed, the new particle $n_{t+1}^{(j)}$ is sampled from a transition probability $p(n_{t+1}|n_t^{(s)})$, which is the one of a random particle $n_t^{(s)}$ (i.e. random $s \in \{1 \ldots N\}$). This can be regarded as virtually increasing the number of particles up to $N \cdot B$, if necessary.

2. **Calculation of particle weights**

Having an *additive noise* assumption, the time domain relation

$$y_t = x_t + n_t$$

can be expressed in the log Mel frequency domain by the following equation[2], if the *phase is omitted* (a detailed analysis can be found in [FW07]):

$$y_t = log(e^{x_t} + e^{n_t})$$

This makes it possible to express a *functional dependency* of the clean speech estimation calculated from $n_t^{(j)}$ and y_t:

$$\begin{aligned} x_t &= log(e^{y_t} - e^{n_t}) \\ \Rightarrow f(y_t, n_t^{(j)}) &:= y_t + log(1 - e^{n_t^{(j)} - y_t}) \end{aligned} \tag{2.6}$$

Next, we can apply the so-called *fundamental transformation law of probabilities*, which is defined as

$$p_y(y) = p_x(f(y)) \cdot |\frac{df(y)}{dy}|$$

In our case this transformation allows to evaluate the *output probability* $p(y_t|n_t)$ (i.e. $p_y(y) := p(y_t|n_t)$), given the relationship (2.6) and a probability distribution of clean speech $p_x(x_t)$ (speech model).

As noted before, speech is usually modeled by a Gaussian mixture model (equation 1.1, page 5). This gives us the probability distribution $p_x(x_t)$ of clean speech.

So the output probability can be expressed as

$$p(y_t|n_t^{(j)}) = p_x(f(y_t, n_t^{(j)})) \cdot \frac{1}{|1 - e^{n_t^{(j)} - y_t}|} \tag{2.7}$$

$$= \sum_{k=1}^{K} \frac{c_k \mathcal{N}(f(y_t, n_t^{(j)}); \mu_k, \Sigma_k)}{|1 - e^{n_t^{(j)} - y_t}|} \tag{2.8}$$

$$= \sum_{k=1}^{K} \frac{c_k \mathcal{N}(y_t + log(1 - e^{\hat{n}_t^{(j)} - y_t}; \mu_k, \Sigma_k)}{|1 - e^{n_t^{(j)} - y_t}|} \tag{2.9}$$

The likelihood $l(n_t^{(j)}; y_t)$ of the j-th particle is actually the *output probability* $p(y_t|n_t)$ of the dynamical system, i.e. $l(n_t^{(j)}; y_t) = p(y_t|n_t^{(j)})$.

The weight of the j-th particle can be calculated as its normalized likelihood

$$w_t^{(j)} := w(n_t^{(j)}, y_t) = \frac{l(n_t^{(j)}; y_t)}{\sum_{i=1}^{N} l(n_t^{(i)}; y_t)}$$

where N is the number of particles.

[2]In our setup, the relationship between corrupted speech, clean speech, and noise only approximates this equation for the log Mel power spectrum, since features derived by warped MVDR are only an approximation to Fourier features, and the liftering step (4. page 8) introduces further non-linearities.

Equation (2.9) is not computable, if $n_t^{(j)}$ exceeds y_t in any spectral bin, since it implies $e^{n_t^{(j)}-y_t} \geq 1$. $p(y_t|n_t^{(j)})$ is set to zero in this case.

3. **Inference of clean speech**

   Usually, PFs for noise tracking deal with the non-linearity of the relation between the system state x and the observation y with a linearization using Taylor series expansions for clean speech inference. This so-called *vector Taylor series* (VTS) approach [MRS96] uses a 0th-order Taylor series to approximate the term $log(1 - e^{n_t-y_t})$.

   In contrast to that, we use a straight forward *direct calculation* (statistical inference approach), derived from equation (2.6):

   (a) Calculation of clean speech hypotheses $x_t^{(j)}$ for each of the N noise samples $n_t^{(j)}$:

   $$x_t^{(j)} = y_t + log(1 - e^{n_t^{(j)}-y_t})$$

   (b) The clean speech estimation $\hat{x}_t$ can be evaluated exploiting equation (2.4), which can actually be calculated by averaging over all N clean speech hypotheses $x_t^{(j)}$, weighted by their normalized likelihoods (particle weights $w_t^{(j)}$):

   $$\hat{x}_t = \sum_{j=1}^{N} w_t^{(j)} \cdot x_t^{(j)}$$

   This simple approach solves the inference problem without approximation and has shown better performance than VTS, while being computationally extremely efficient [FW07].

4. **Resampling of particles**

   A major problem when using a SIR particle filter is the degeneracy of particles. Meaning that after a small number of time steps, all, but only a few particles get insignificant weights and the probability density is described inadequately, since the contribution of most of the particles to the posteriori density is almost zero. Therefore, the *systematic residual resampling* algorithm [BDH03] is used to resample the particles. This is a semi-deterministic variant of the *sequential sampling importance resambling* (SIR) algorithm. SIR can be seen a pruning step, where samples having a low rating in the likelihood function die, and likely samples are multiplied. Figure 4 schematically illustrates the concept.

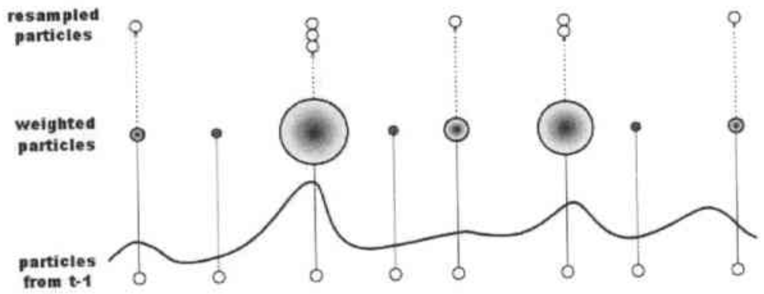


Figure 4: Weighting (indicated by particles' diameter) and resampling of particles

3 Phoneme Specific Particle Filtering

One crucial step to improve the estimation of a particle filtering algorithm, is to get a more precise weighting of the particles (step 2 of the algorithm in the preceding section). This can be achieved by using a better way to model how speech looks like. In other words, we would appreciate to have an output probability distribution (2.9), which models the observed speech signal more precisely.

Traditionally particle filtering for speech feature enhancement is performed using a *general speech model*. This means one generic model, giving a likelihood which states, how far a specific spectral frame has the shape of what is assumed to be clean speech. Faubel and Wölfel [FW06] have proposed a time dependent, phoneme specific clean speech model:

$$p_{phone(t)}(x) = \sum_{i=1}^{M} c_{i,phone(t)} \mathcal{N}(x|\mu_{i,phone(t)}, \Sigma_{i,phone(t)}) \tag{3.1}$$

It is obvious, that a general model for speech is a very unfocused criterion, ignoring the dynamic properties of speech, since different phonemes have different important frequency characteristics (e.g. high frequencies for fricatives like 'S' and low frequent formants for vowels).

Figure 5 shows the probability distribution of a general speech model in comparison to different phoneme models.

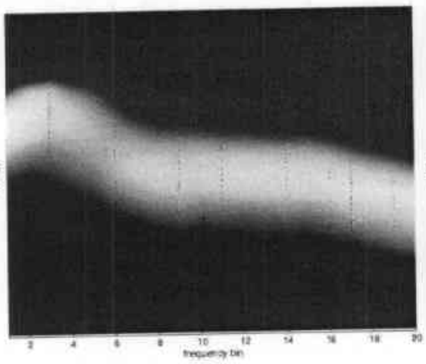
Our research team has developted a PF, which performs a particle weight calculation based on a *phoneme specific speech model*, as further addressed in this thesis, and first experiments showed that the improved estimation can lead to a notable increase of word recognition accuracy [FW06].

To establish a phoneme dependency in the PF, it has to be known, what is spoken at a specific point of time in an utterance. But, since it is the actual job of the ASR system to obtain these phonemes (and reveal word sequences), they are not known in advance. Therefore, a *two pass system* is used (Figure 6).

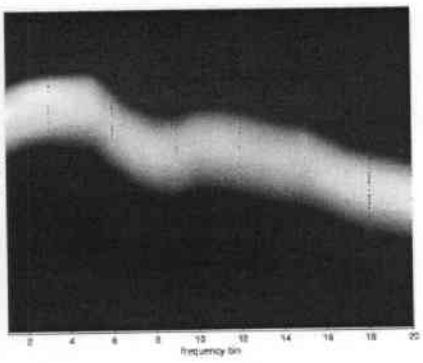
3.1 Two pass Particle Filter

In the *first pass* a phoneme independent, *general acoustic model for speech* is applied to the acoustic signal, as it is commonly used for particle filtering. Afterwards, as usual, the ASR system does its decoding to generate a hypothesis of what has been said. This hypotheses is the basis for the *second pass*. As it is known, at that point, what phonemes have been spoken throughout an utterance, the models can be switched dynamically and particle filtering can be performed using more specific acoustic speech models.

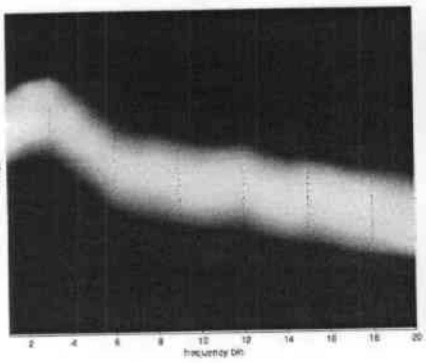
By using a phoneme transcription hypothesis as indicator which phoneme specific speech model to use, the PF's stationary speech model is replaced by a *dynamically time varying* clean speech model. Furthermore, by coupling back the recognition results of the first pass into the speech feature enhancement stage, it benefits from the sophisticated methodologies of the decoder of the ASR system.



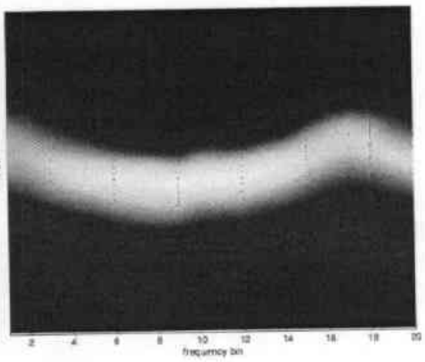
(a) General speech



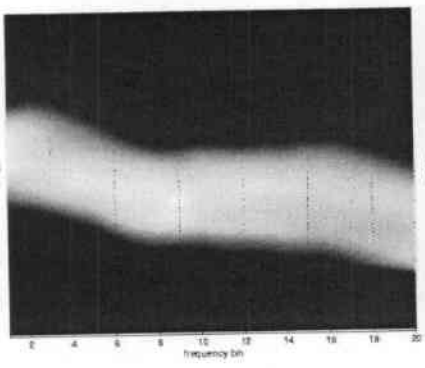
(b) Phoneme „E"



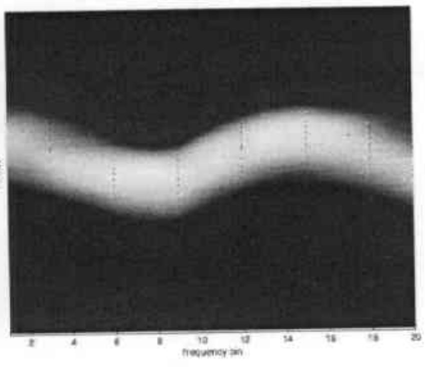
(c) Phoneme „M"



(d) Phoneme „S"



(e) Phoneme „T"



(f) Phoneme „SH"

Figure 5: Probability distribution of general speech model and phoneme models

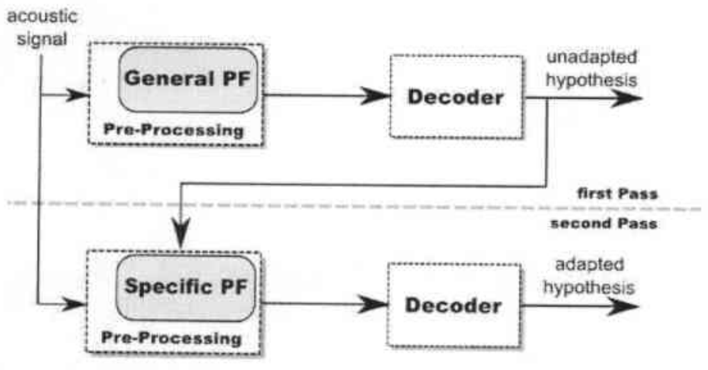


Figure 6: Two pass particle filter

3.2 Problems of the two-pass approach

In preceding experiments [FW06] recognized, that using a phoneme specific speech model for particle filtering is a very promising approach, since the word error rate (WER) could be reduced by more than 5%[3] compared to particle filtering using a general speech model. But only if a *reference hypothesis transcription* is available, specifying which phoneme is spoken at a specific point of time. This can be regarded as the output of a perfect ASR system in the first pass. Of course, in a real world scenario, such a reference is not available in advance, that is why we use the decoded hypothesis of a previous decoding pass. However, such a hypothesis of the first pass may be wrong, which results in wrong probability distributions being used as speech models for particle filtering. The effect is, that in case of a mismatched hypothesis, the PF cleans and enhances the noisy speech features into the direction of a wrong phoneme and the decoder recognizes the wrong phoneme even more likely, which leads to a worse WER than using a general speech model for particle filtering. We call this phenomenon '*model tying*'. This effect is most substantial for similar phonemes which are likely to be confused.

Another problem, when using a phoneme-specific PF, is that dynamically switching the models leads to very sudden changes in the particle's weights, which can destabilize the PF.

To overcome these problems, [FW06] introduced a *mixed model* consisting of the interpolation of a phoneme specific and a general speech model. In addition to reconfirming these experiments, this thesis introduces the following approaches, which will be discussed in the next sections:

- Using a phoneme specific model which is additionally trained by phonemes, which are likely to be interchanged
- Grouping phonemes to classes and using phoneme class-based speech models

[3]SNR 0 dB

3.3 General and phoneme-specific mixed models

Interpolating between the two different approaches - general speech model $p(x)$ and phoneme-specific speech model $p_{phoneme(t)}(x)$ - overcomes the model tying problem, as it was shown in [FW06]. A (phoneme dependent) mixed model can be defined as

$$p_{mix(t)}(x) = \alpha \cdot p_{phoneme(t)}(x) + (1 - \alpha) \cdot p(x)$$

where α is the mixture weight and $phoneme(t)$ is the phoneme in the hypothesis transcription at time t. It has been evaluated, that the increased number of Gaussians in this equation is not relevant for the results.

3.4 Confusability based mixed model

Due to their characteristics, some phonemes are conceivably interchanged by the speech recognition system more often than others. Thus, the model tying effect is assumed to be more severe for those phonemes, which have a higher probability of confusion in the decoding stage. The *confusability based mixed model* tries to tackle this problem by interpolating each phoneme model with a series of phoneme models, which are known to be interchanged often. The goal is, not to focus the enhancements by the PF strongly on the single phoneme given by the hypothesis transcription, but also let it be biased by other possible phonemes, according to the probability of their confusability, so that the decoder can identify the correct word.

The phoneme specific, confusability based mixed model is defined as follows:

$$p_{confusMixed(t)}(x) = \begin{cases} \lambda_{phoneme(t)} \cdot p_{phoneme(t)}(x) & \text{if } A = phoneme(t) \\ \sum\limits_{A \in \Psi} \lambda_{phoneme(t),A} \cdot p_{phoneme(t)}(x) & \text{if } A \neq phoneme(t) \end{cases} \tag{3.2}$$

where

$$\lambda_{phoneme(t)} = \mathbb{P}(phoneme(t) \text{ is not interchanged}),$$
$$\lambda_{phoneme(t),A} = \mathbb{P}(phoneme(t) \text{ is interchange with } A)$$

Ψ is the set of all phonemes and $phoneme(t)$ is the phoneme in the hypothesis transcription at time t. The probabilities $\lambda_{phoneme(t)}$ and $\lambda_{phoneme(t),A}$ were determined by an offline experiment, which messured the number of interchanges of all pairs of phonemes by comparing the result of a recognition pass (i.e. the decoded hypothesis transcription) to a reference transcription of approximately 100 hours of speech data.

Typically $\lambda_{phoneme(t)}$ is a much higher probability than $\lambda_{phoneme(t),A}$, which is zero for many phonemes A in the phoneme set Ψ. However, if $phoneme(t)$ is potentially interchanged with many other phonemes, it leads to a very high number of Gaussians in equation (3.2). Therefore, the experiments for this approach use a fixed number of Gaussians, but are trained by samples, which belong to differrent phonemes, distributed corresponding to $\lambda_{phoneme(t)}$ and $\lambda_{phoneme(t),A}$.

4 Class-Based Phoneme Models

As the phoneme specific PF is focused too much and therefore tends to influence the spectrum towards a maybe wrong hypothesis, the idea is, to tie similar phonemes together into one phoneme cluster. This should also result in smoother transitions between the different models and avoid the dying out of particle, because of rapid changed likelihoods of the particles. The goal is to emphasize the PF performance by exploiting the characteristics of more specific speech models, while dropping the irrelevant characteristics. E.g. phonemes which are similar and likely to be interchanged may be clustered in one class, so that this does not influence the decoder in the way of *model tying*.

4.1 Class-based particle filtering

Generally, there are two different ways how data can be clustered - *knowledge-driven* (supervised) and *data-driven* (unsupervised).

Heuristic, supervised clustering means, that there is a human, who has expert knowledge in the special domain of the data to be clustered (e.g. a linguist, or phonologist for the domain of ASR). This expert can analyze the data and create clusters in a way data can be separated by his knowledge. The methodology of using manually labeled data for clustering has shown good results in many different fields of data clustering (e.g. acoustic model combination for multilingual phoneme set creation [SK06] chapter 4.4). However, it is time consuming and expensive compared to unsupervised clustering. Moreover, many theoretical concepts are difficult to apply to real world data, where various problematic effects have to be considered. E.g. in the case of finding similarities between phonemes, it is complex to deal with fluent, spontaneous speech, where effects like coarticulation occur.

The main focus of this thesis lies in purely data driven unsupervised clustering approaches (without human interaction). So a clustering algorithm has to determine on its own, which pieces of data belong to one class and which do not. We have addressed a hierarchical clustering algorithm to achieve this.

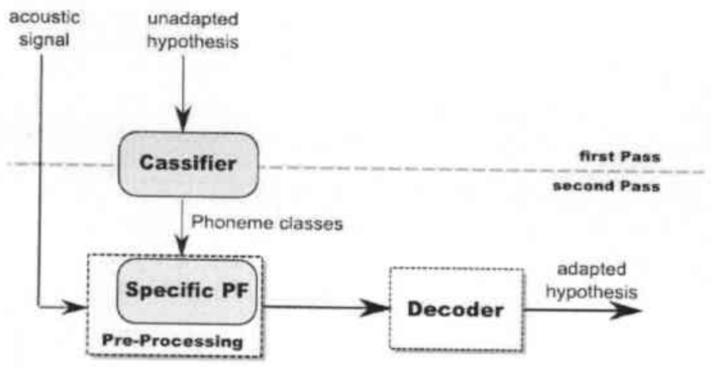


Figure 7: Two-pass particle filter using phoneme classes

Figure 7 shows an extension of the two-pass approach introduced in the previous section having an additional classifier component, which maps the phoneme transcription (hypothesis) of the first pass to phoneme classes, following a certain classification strategy.

The classification strategy realizes the various clustering approaches and parameters we addressed, e.g. the clustering policy (i.e. data-driven or knowledge-based), number of classes, used distance measure, etc.

4.2 Supervised heuristical clustering

First, we focus on the knowledge based approach, and obtain simple phoneme classifications, based on phonologically well known phoneme classes.

This section introduces the evaluated classes and shortly describes their different characteristics.

4.2.1 Voiced and Unvoiced speech

One of the most fundamental classification is the distinction between voiced and unvoiced speech. Voiced speech is quasi-periodic, consisting of a fundamental frequency corresponding to the pitch of the speaker and its harmonics. Unvoiced speech is stochastic in nature and do not consist of a periodic part in the spectrum. It can be modeled as white noise convolved with an infinite impulse response filter.

So the following two phoneme classes (sets of phonemes) have been defined

$$\begin{aligned}
\mathcal{C}_{voiced} = \{&AA, AE, AH, AO, AW, AX, AXR, AY, B, D, DH, EH, ER, EY,\\
&G, IH, IX, IY, JH, L, M, N, NG, OW, OY, R, V, UW, UH, W,\\
&Z, ZH, XL, XM, XN, Y\}\\
\mathcal{C}_{unvoiced} = \{&CH, F, HH, K, P, S, SH, T, TH\}
\end{aligned}$$

4.2.2 Vowels and Consonants

In most languages sounds can be distinct between vowels and consonants. On the one hand vowels are articulated without major constrictions in the vocal tract, on the other consonants are formed by characteristic constrictions in the throat or obstruction in the mouth. Vowels carry most energy of the speech signal, whereas consonants are weak, often looking similar to silence, and therefore are likely to be dominated by noise.

So the following two phoneme classes (sets of phonemes) have been defined

$$\begin{aligned}
\mathcal{C}_{vowels} = \{&AA, AE, AH, AO, AW, AX, AY, EH, EY, IH, IX, IY, OW, OY,\\
&UW, UH\}\\
\mathcal{C}_{consonants} = \{&AXR, B, CH, D, DH, ER, F, G, HH, JH, K, L, M, N, NG, P,\\
&R, S, SH, T, TH, V, W, XL, XM, XN, Y, Z, ZH\}
\end{aligned}$$

4.3 Unsupervised purely data driven clustering

This section presents how to build phoneme clusters based on an unsupervised clustering method. One of the best known unsupervised clustering methods is to build up a hierarchical cluster tree e.g. as presented in [DHS01]. This way of clustering is also popular in finding similar phonemes for multi-lingual automatic phoneme clustering e.g. [BdMCAM99] and performs good for speaker segmentation (e.g. [JS04]).

Hierarchical clustering can be performed in two different ways: agglomerative (bottom-up) and diversive (top-down). In the following we use an agglomerative approach, which means, it starts with n singleton elements to be clustered and merges *similar* elements or clusters into bigger clusters up to one single cluster consisting of all n elements.

The choice, which elements and clusters are most 'similar', and therefore are going to be combined, can be determined by many different kinds of distance measures, which are discussed later. The result of the hierarchical clustering algorithm can be illustrated as a rooted binary tree (a.k.a. *dendrogram*), starting with single elements to be clustered (leafs) and ends in one single cluster (root).

4.3.1 A hierarchical clustering algorithm

Listing 1: Hierarchical clustering algorithm

```
 1: Cluster[] hierarchClustering(Cluster[] clusters,
 2:                int finalNumberOfClusters) {
 3:
 4:    for ( i=1..n; i<=n; i++ )
 5:       for ( j=1..n; j<=n; j++ )
 6:          distances[i,j] := D(i,j);
 7:
 8:    while ( |clusters| > finalNumberOfClusters) {
 9:       (i,j) := getMinDistance(clusters, distances);
10:       clusters := mergeClusters(i, j, clusters);
11:       |cluster|--;
12:    }
13:
14:    return clusters;
15: }
```

Listing 1 shows a typical agglomerative hierarchical clustering algorithm in pseudo-code. The first step in hierarchical clustering is calculating all distances between the elements (line 6) to be clustered. In each of the following steps, the closest clusters are determined (line 9) and combined to a new cluster (line 10), resulting in a decrease of the number of clusters by one. The algorithm terminates when the specified final number of clusters has been reached and returns the resulting clusters.

There remain two important points to make the hierarchical clustering algorithm work:

1. Definition of distances between the clusters
2. Merging of clusters to bigger clusters

4.3.2 Distance definitions

In order to cluster similar phonemes, represented by Gaussian mixture models (GMM)s, the next sections present different distance measures and merging methods.

Most of the distances usually used in the fields of pattern recognition are defined between single Gaussian distributions, which means monogaussian models. E.g. there is no closed form representation of the relative entropy, that could be used as distance between GMMs. So, in many cases, monogaussian distance's definitions must be extended to the use for GMMs.

The following distances, were tested in the hierarchical clustering algorithm, and are discussed further in the following:

- Distances between monogaussian models:
  - Euclidean distance
  - Extended Mahalanobis distance
  - Kullback Leibler distance
- Distances between Gaussian mixture models:
  - Kullback Leibler distance for GMMs
  - Earth Movers distance
- Euclidean distance between speech samples

In this chapter, Φ_i, Φ_j denote Gaussian distributions, with means μ_i, μ_j and diagonal covariance matrices Σ_i, Σ_j (i.e. $\Phi_i := \mathcal{N}(x|\mu_i, \Sigma_i)$, $\Phi_j := \mathcal{N}(x|\mu_j, \Sigma_j)$). Γ_i and Γ_j are Gaussian mixtures, as introduced in (1.1).

4.3.3 Merging classes

After the distance calculation in the hierarchical clustering algorithm, the nearest clusters are merged into a new bigger cluster (listing 1 line 10). As the phoneme classes are represented by GMMs, an easy method achieve this, is to mix the models by adding and adjusting their distribution weights:

$$p_{\mathcal{AB}}(x) = \alpha \cdot p_{\mathcal{A}}(x) + (1 - \alpha) \cdot p_{\mathcal{B}}(x)$$

where $p_{\mathcal{A}}(x)$ and $p_{\mathcal{B}}(x)$ are the GMMs of the two clusters and α is a mixture weight, which can be set, according to the number of phonemes the clusters consist of, to achieve an equal weighting of the intra class distribution weights. The number of Gaussians of the new GMM is the sum of the Gaussians of both clusters.

This method of merging was applied in the experiments for clustering the Kullback Leibler distance for GMMs and the Earth mover's distance.

Apart from this approach, combined models can be created by training new distributions, based on samples of the two phoneme classes to be merged:

There are different ways to train the parameters of a GMM. We use the widespread *split and merge* (SAM) training algorithm (e.g. [UNGH98]), which iteratively improves a model by performing alternately a phase of *split* operations followed by a phase of *merge* operations. Starting with a monogaussian model, based on all training samples, the splitting divides large cluster (of training samples) into two subclusters, whereas the merging combines two nearby clusters. The parameters of the GMMs are improved by several iterations of the expectation maximization (EM) algorithm, after each merge and split phase.

This method of merging was applied in the experiments for clustering using the monogaussian distances.

The Euclidean distance between speech samples does not necessaryly need a special merging step. The distance calculation can just be performed between the samples in each of the clusters.

4.3.4 Eucledian distance

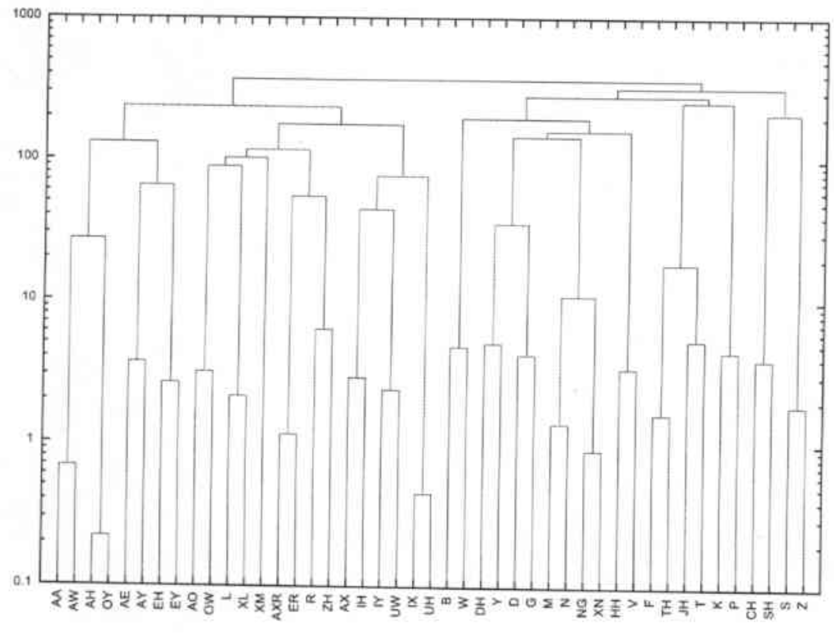


Figure 8: Hierarchical clustering based on Euclidean distance

One of the most popular and simple distances is the Euclidean distance:

$$D^{EUCL}(\Phi_i, \Phi_j) = \sqrt{(\mu_i - \mu_j)^t(\mu_i - \mu_j)}$$

Obviously, this distance is only based on means μ_i,μ_j of Gaussians Φ_i, Φ_j and does not consider their covariances. This can e.g. result in very differently scattered distributions having the same distance.

Figure 8 shows the dendrogram when clustering the phoneme specific models by the hierarchical clustering algorithm.

4.3.5 Extended Mahalanobis distance

The well known Mahalanobis distance is defined to messure the distance between two vectors in a vector space:

$$D_{x,y}^{MHN} = \sqrt{(x-y)^t \Sigma^{-1} (x-y)} \tag{4.1}$$

Where x and y are the vectors and Σ^{-1} is the inverse covariance matrix of the vector space. In fact the Mahalanobis distance is a weighted Euclidean distance where the weighting is determined by the covariance matrix (both distances are identical, if the covariance matrix is the identity).

It can be extended to a distance between two Gaussians, as follows:

$$D^{eMHN}(\Phi_i, \Phi_j) = \sqrt{(\mu_i - \mu_j)^t (\Sigma_i + \Sigma_j)^{-1} (\mu_i - \mu_j)}$$

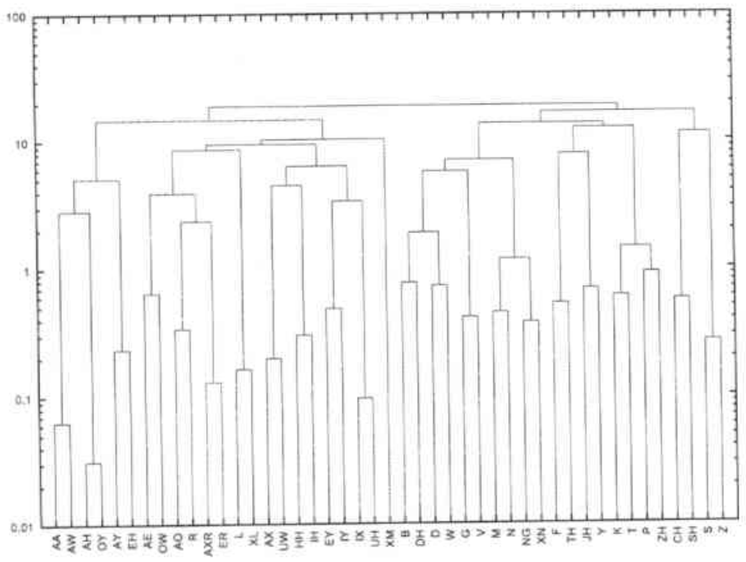


Figure 9: Hierarchical clustering based on extended Mahalanobis distance

4.3.6 Kullback-Leibler distance

The *Kullback Leibler divergence* is usually used as a means to measure the distance between a true probability density $p(x)$ and an approximation to it $\hat{p}(x)$. From an information theoretical point of view, it can also be seen as the *relative entropy* for using $\hat{p}(x)$ instead of $p(x)$:

$$D_{p,\hat{p}}^{KLdiv} = \int_{-\infty}^{\infty} p(x) \cdot \log \frac{p(x)}{\hat{p}(x)} dx$$

This definition is not sufficient as a distance, since it is not symmetric (i.e. $D_{f,g}^{KLdiv} \neq D_{g,f}^{KLdiv}$). So the following symmetrised divergence definition is used in this thesis. It will be called *Kullback Leibler distance* in the following.

$$D_{f,g}^{KL} = D_{f,g}^{KLdiv} + D_{g,f}^{KLdiv} \tag{4.2}$$

It should be noted, that $D_{f,g}^{KL}$ is still not metric, since it does not satisfy the triangle inequality in general.

In the case of $f()$ and $g()$ being multivariate Gaussian distributions the Kullback-Leibler divergence can be computed as

$$D^{KLdiv}(\Phi_i, \Phi_j) = \frac{1}{2} log \frac{|\Sigma_i|}{|\Sigma_j|} + \frac{1}{2} tr \Sigma_i (\Sigma_j^{-1} - \Sigma_i^{-1}) + \frac{1}{2} tr \Sigma_j^{-1} (\mu_i - \mu_j)(\mu_i - \mu_j)^t$$

where tr denotes the trace.

Using diagonal covariance matrices and assuming the coefficients to be statistically independent, allows a simpler calculation of the Kullback-Leibler distance (4.2):

$$D^{KL}(\Phi_i, \Phi_j) = \frac{1}{2} \sum_{l=1}^{d} (\frac{\sigma_{i,l}^2}{\sigma_{j,l}^2} + \frac{\sigma_{j,l}^2}{\sigma_{i,l}^2} - 2 + (\frac{1}{\sigma_{j,l}^2} + \frac{1}{\sigma_{i,l}^2})(\mu_{i,l} - \mu_{j,l})^2) \tag{4.3}$$

where d is the dimension of the feature space and $\sigma_{i,l}^2$, $\sigma_{j,l}^2$ are variances of the dimension l.

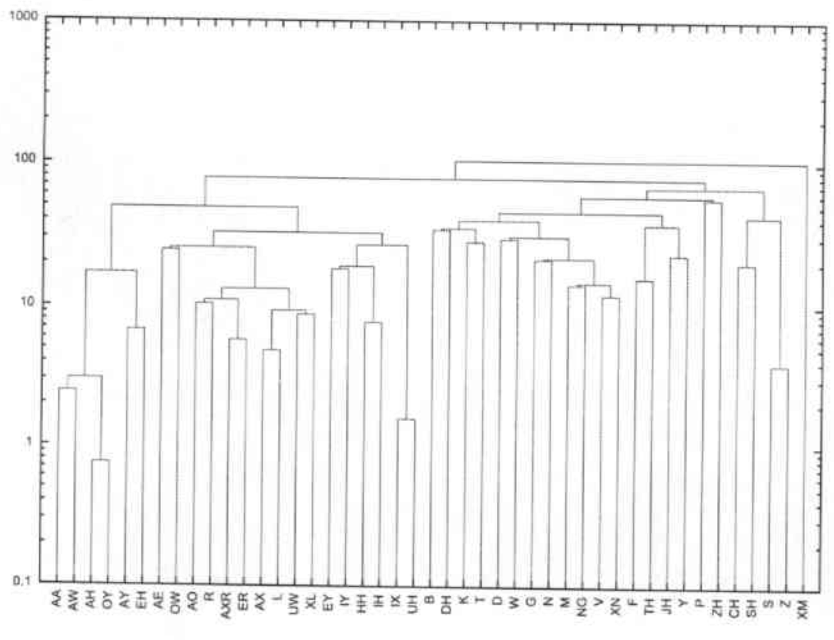


Figure 10: Hierarchical clustering based on Kullback-Leibler distance

4.3.7 Distances between Gaussian Mixture Models

All distances presented in the previous sections, are only defined for single Gaussian distributions. So, we have to face the problem to apply these distances to GMMs. One way to handle this is to find a matching between single Gaussian components of the two GMMs and compute the distance between the mapped pairs (e.g. [GA05]). We have tried the following mapping:

$$D(\Gamma_i, \Gamma_j) = \sum_{\Phi_x \in \Gamma_i} \sum_{\Phi_y \in \Gamma_j} c_x \cdot c_y \cdot D(\Phi_i, \Phi_j)$$

However this approach did not lead to balanced cluster trees and therefore was not considered in our experiments. Another approach to use distances between single Gaussians for GMMs is, simply to compute distances on monogaussian models. These can be generated by a training algorithm.

4.3.8 Earth Mover's distance

The Earth Mover's Distance, proposed in [RTG98], tries to express the minimum amount of work needed to transform the 'distribution mass' of one distribution into another distribution. In [LS01] it is used as distance between GMMs based on the *Kullback Leibler distance* (4.3) as a 'ground distance' between the single Gaussian components of the GMMs. It can be defined as

$$D^{EMD}(\Gamma_i, \Gamma_j) = \sum_{\Phi_x \in \Gamma_i} \sum_{\Phi_y \in \Gamma_j} D^{KL}(\Phi_x, \Phi_y) \cdot F(\Phi_x, \Phi_y)$$

where $F(\Phi_x, \Phi_y)$ is the *flow* from Φ_x to Φ_y. This flow, which also can be regarded as a weighted matching between the single Gaussians of the GMMs, is the solution to the transportation problem [AMO93], where the Gaussians Φ_i with their distribution weight c_i supply the consumers Φ_j with capacity c_j.

4.3.9 Kullback Leibler distance on GMMs

The symetrised Kullback Leibler distance (4.3) can be extended by the distribution weights of the GMMs:

$$D^{KLGMM}(\Gamma_i, \Gamma_j) = \frac{1}{2} \sum_{\Phi_x \in \Gamma_i} \sum_{\Phi_y \in \Gamma_j} 2(w_x - w_y) \cdot log(\frac{w_x \cdot \sigma_y}{w_y \cdot \sigma_x}) +$$
$$\frac{w_x \cdot \sigma_x^2}{\sigma_y^2} + \frac{w_y \cdot \sigma_y^2}{\sigma_x^2} + (\frac{w_y}{\sigma_x^2} + \frac{w_x}{\sigma_y^2})(\mu_x - \mu_y)^2 - (w_x + w_y)$$

4.3.10 Sample distance

A different approach from building phoneme classes based on speech models, is to measure the distance between samples of speech in order to get a classification.

The square of the Euclidean distance can be used as a straight forward metric:

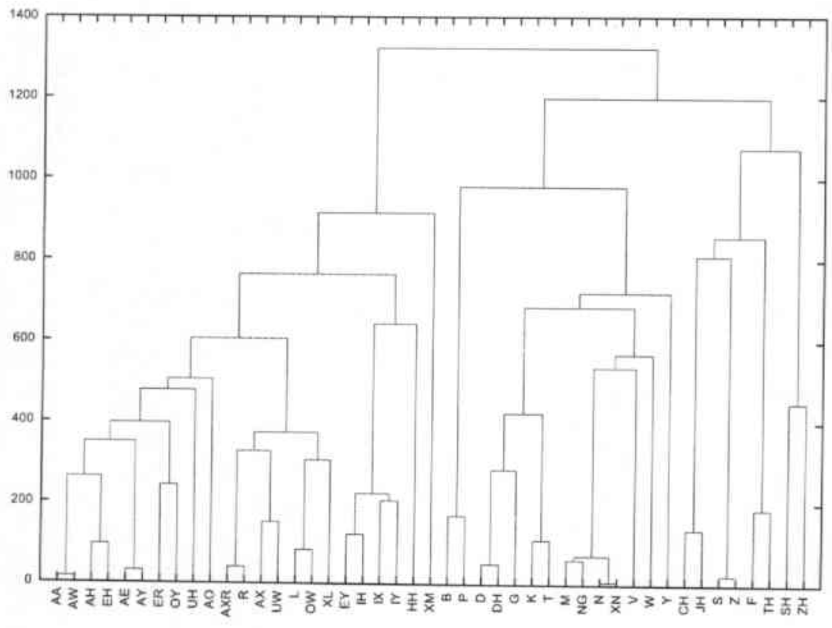


Figure 11: Hierarchical clustering based on Earth Mover's distance

$$D^{euclSMP}(P_1, P_2) = \frac{1}{|P_1| \cdot |P_2|} \sum_{p1 \in P_1} \sum_{p2 \in P_2} \sum_{d=1}^{n} (p_{1,d} - p_{2,d})^2$$

where P_1 and P_2 are sets of speech samples of two phonemes and $|\cdot|$ denotes the cardinality of the set.

4.4 Confusability clustering

In contrast to find similarities between phonemes, the idea behind this clustering method is the fact (similar to the confusability based mixed model in section 3.4), that some phonemes are conceivably interchanged by the speech recognition system more often than others. So the goal of the clustering method proposed in this section, is to try to use the same phoneme cluster and therefore the same (and still correct) speech model for particle filtering, even if the decoded hypothesis of the first pass is wrong (i.e. the hypothesis indicates a phoneme which is known to be interchanged often). This should compensate the effect of *model tying* in a more direct way, than by the approach using a distance measures between GMMs, which was proposed in prior sections, because it gains from the complex evaluations and optimizations in the decoding stage of the ASR system, in addition to considering similarities between GMMs. Thus, regarding the interchanges between phonemes offers a potentially good indicator for building phoneme classes.

To achieve this, we introduce a messure for clustering which answers the question "How likely will two phoneme classes $\mathcal{X}$ and $\mathcal{Y}$ be interchanged?". This measure is called Phoneme Interchange Rate (PIR) in the following.

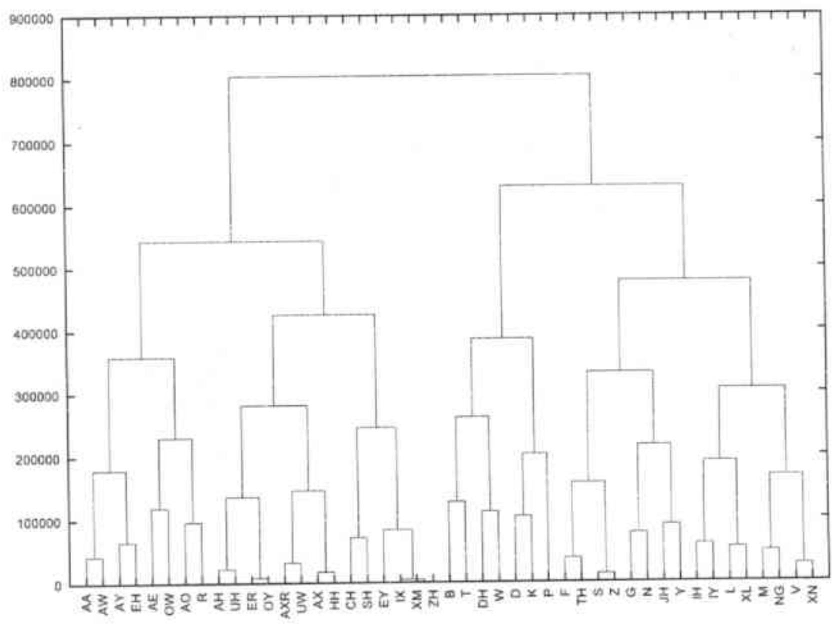


Figure 12: Hierarchical clustering based on Kullback-Leibler distance between Gaussian mixtures

Phoneme interchange rate

The offline experiment, which was already mentioned in section 3.4, provides us the number of interchanges of all pairs of phonemes by comparing the result of a recognition pass (i.e. the decoded hypothesis) to a reference transcription of the speech data. The *interchange rate* between the phonemes X and Y (PIR) can be computed as follows:

$$PIR(X,Y) = \mathbb{P}(\text{ } X \text{ and } Y \text{ are interchanged } | \text{ } X \text{ occurs and } Y \text{ occurs })$$

$$= \frac{\mathbb{P}(\text{ } X \text{ and } Y \text{ are interchanged })}{\mathbb{P}(\text{ } X \text{ occurs }) \cdot \mathbb{P}(\text{ } Y \text{ occurs })}$$

$$= \frac{\dfrac{\#ic(X,Y) + \#ic(Y,X)}{\sum\limits_{A,B \in \Psi} \#ic(A,B)}}{\dfrac{\#oc(X)}{\sum\limits_{C \in \Psi} \#oc(C)} \cdot \dfrac{\#oc(Y)}{\sum\limits_{C \in \Psi} \#oc(C)}} \tag{4.4}$$

Where $\#ic(X,Y)$ means the number of interchanges of phoneme X by phoneme Y. $\#oc(C)$ is the number of occurrences of phoneme C in the ref-

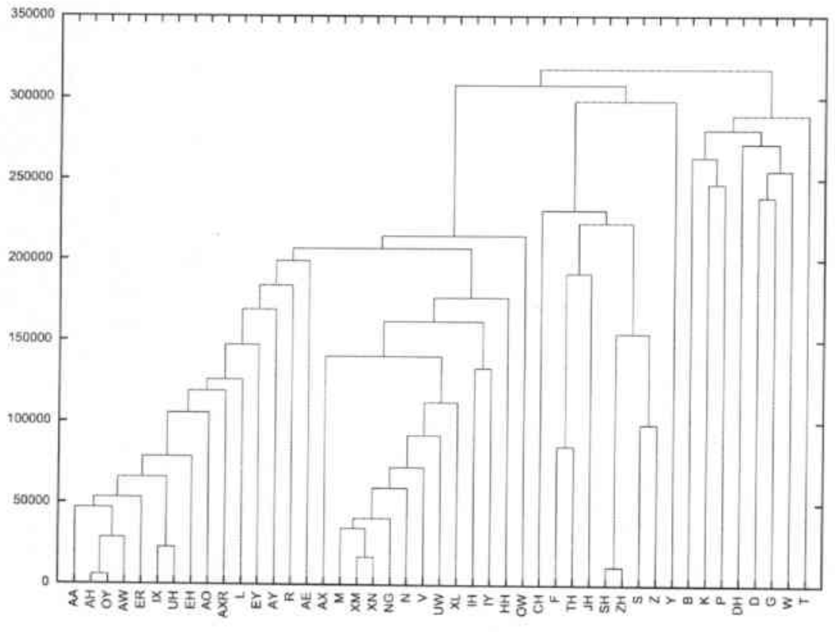


Figure 13: Hierarchical clustering based on Euclidean distance between samples

erence transcription, Ψ is the set of all phonemes.

In the hierarchical clustering algorithm two phonemes X and Y are merged to one new cluster, when their interchange probability $PIR(X, Y)$ is the largest. Therefore, the PIR is defined as a symmetric measure (i.e. $PIR(X, Y) = PIR(Y, X)$). The denominator of (4.4) can be regarded as elimination of the a-priori probability. Not considering the a-priori probability would lead to a cluster tree, where nearly in each step of the algorithm a phoneme is added to one single big class. This would result in very unevenly distributed class sizes. Since phonemes, which occur very often, have a potentially high number of interchanges, and therefore are merged first. With the effect of getting an even higher number of occurrences and a probable higher number of interchanges in the following step.

Above, the definition of the phoneme interchange rate is between two single phonemes X and Y. To be able to use it in the hierarchical clustering algorithm it has to be extended to work for two sets of phonemes (i.e. phoneme classes) $\mathcal{X}$ and $\mathcal{Y}$:

$$PIR(\mathcal{X},\mathcal{Y}) = \mathbb{P}\big(\bigcup_{X\in\mathcal{X},Y\in\mathcal{Y}} X \text{ is interchanged with } Y \mid \bigcup_{X\in\mathcal{X},Y\in\mathcal{Y}} X \text{ occurs and } Y \text{ occurs } \big)$$

$$= \frac{\displaystyle\sum_{X\in\mathcal{X},Y\in\mathcal{Y}} \frac{\#ic(X,Y)+\#ic(Y,X)}{\displaystyle\sum_{A,B\in\Psi}\#ic(A,B)}}{\displaystyle\sum_{X\in\mathcal{X},Y\in\mathcal{Y}} \frac{\#oc(X)}{\displaystyle\sum_{C\in\Psi}\#oc(C)}\cdot\frac{\#oc(Y)}{\displaystyle\sum_{C\in\Psi}\#oc(C)}}$$

$$= \frac{\left(\displaystyle\sum_{C\in\Psi}\#oc(C)\right)^2}{\displaystyle\sum_{A,B\in\Psi}\#ic(A,B)} \cdot \frac{\displaystyle\sum_{X\in\mathcal{X},Y\in\mathcal{Y}} \#ic(X,Y)+\#ic(Y,X)}{\displaystyle\sum_{X\in\mathcal{X},Y\in\mathcal{Y}} \#oc(X)\cdot\#oc(Y)}$$

The following diagram shows the corresponding dendrogram, as outputted by the hierarchical clustering algorithm:

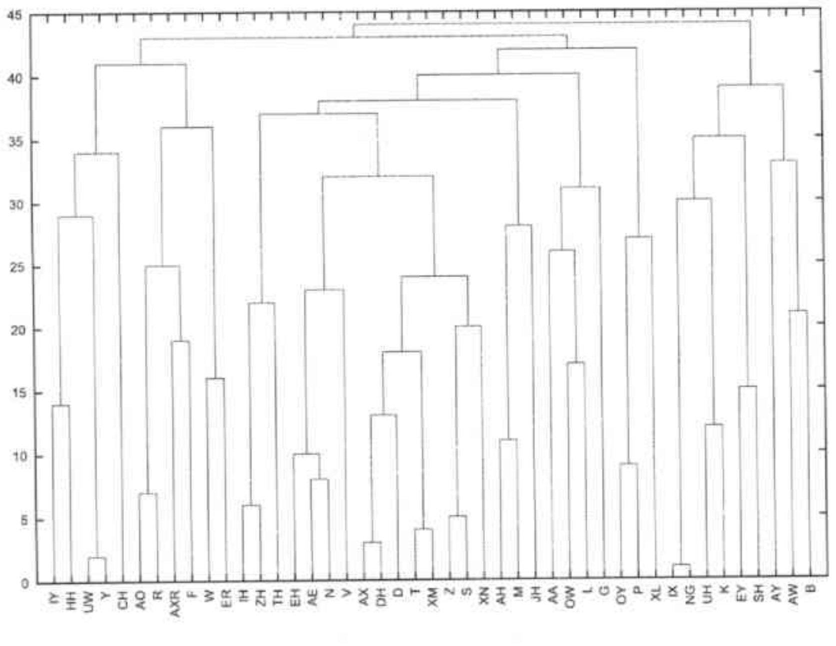


Figure 14: Hierarchical clustering based Phoneme Interchange Rate

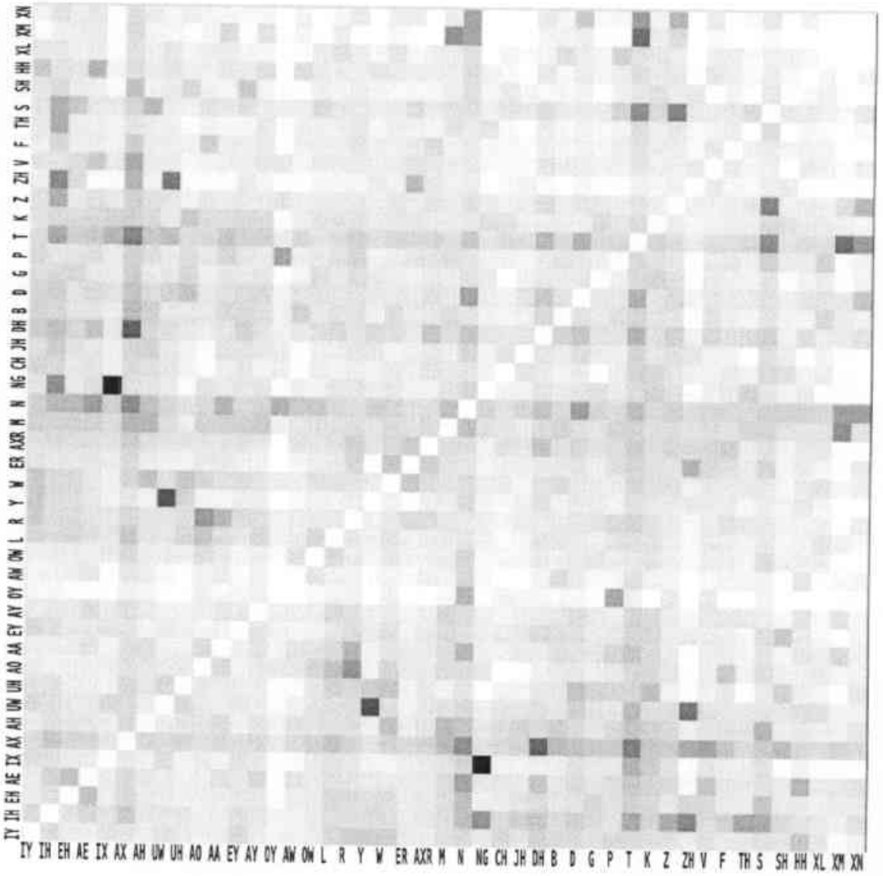


Figure 15: Phoneme Interchange rate (darker fields denote higher PIR)

5 Experiments

After the theoretical part of this thesis, we now try to evaluate and compare the discussed particle filtering enhancements by recognition experiments using the system setup described in the following subsection. The different experiments are quantified in terms of word error rate (WER).

5.1 System setup

To analyze the performance of the discussed particle filtering improvements, all of the proposed techniques were tested on 45 minutes of seminar talk, which presents a very challenging task to all parts of the ASR system: The acoustic models have to cope with data that contains spontaneous, non-native and disfluent speech. The language model has to deal with demanding technical topics, which are focused on automatic speech recognition and signal processing.

Dynamic noise was artificially added to the clean speech signal (additive noise). We selected to evaluate all of the proposed techniques under the *signal to noise ratios* (SNR)s 0 dB, 5 dB and 10 dB. The artificially added noise [Pro] consists of a collection of different kinds of environmental noise with a high degree of non-stationary sounds. It contains noise of trucks driving past, slamming containers and human sounds like distant voices, shouts and coughing, etc.

The acoustic models were generated based on approximately 100 hours of training data, which was taken from ICSI, NIST, CMU meeting corpora, as well as Translanguage English Database (TED), leading to 3500 context dependent codebooks, which consist of up to 64 Gaussians with diagonal covariances.

We use a 3-gram language model, trained on approximately 23000 words with a perplexity of 125.

In the second pass, we also used adaption. Namely, maximum likelihood linear regression (MLLR) and constrained MLLR (feature space adaption) adapted the corresponding first pass hypothesis.

In each experiment the PF uses $N = 100$ particles, since it has been discovered, that a much higher number of particles only leads to slight performance improvements, while being computationally significantly more expensive. A static noise variance $\epsilon_t = 10$ (error term in equation (2.5)) has been chosen in each of the experiments, however different values have shown comparable results.

The traditional general speech model is represented by one single Gaussian Mixture for speech, consisting of up to 64 Gaussians. Each cluster of the clustered phoneme models (presented in chapter 4) also consists of up to 64 Gaussians. The mixed models (section 3.4 and 3.3) consist of up to 64 Gaussians for each of the 45 phonemes. The clean speech model for the purely phoneme specific model (section 3.1) consists of 16 Gaussians per phoneme.

5.2 Experimental results

The experiments using a particle filter with the traditional general speech model can be seen as the baseline to the following improvements by mixed models or phoneme classes.

SNR 0 dB				
	Unadapted		Adapted	
	Hypothesis	Reference	Hypothesis	Reference
No PF	61.3%		43.9%	
General PF	56.6%		41.9%	
PSM	55.1%	52.9%	41.4%	39.8%
Mix	55.3%	54.8%	41.9%	40.0%
ConfusMix	56.5%	54.6%	41.5%	39.5%

Table 1: Experimental results SNR 0 dB

SNR 5 dB				
	Unadapted		Adapted	
	Hypothesis	Reference	Hypothesis	Reference
No PF	50.1%		35.4%	
General PF	46.3%		36.1%	
PSM	45.5%	44.9%	34.5%	33.5%
Mix	45.9%	45.6%	35.3%	33.5%
ConfusMix	45.9%	43.9%	34.3%	33.9%

Table 2: Experimental results SNR 5 dB

SNR 10 dB				
	Unadapted		Adapted	
	Hypothesis	Reference	Hypothesis	Reference
No PF	42.9%		31.2%	
General PF	40.5%		31.2%	
PSM	41.4%	39.4%	32.2%	31.2%
Mix	40.6%	39.7%	32.5%	32.3%
ConfusMix	41.0%	40.0%	31.7%	32.0%

Table 3: Experimental results SNR 10 dB

First, Table 1, Table 2 and Table 3 show the *general particle filter* results (General PF) in comparison to the results without a speech feature enhancement step by particle filtering (No PF). The values with adaption as well as without, show significant improvements compared to the recognition experiments without PF (only the adapted experiment at SNR 5 dB appears to be an outlier).

The unadapted general PF experiments act as the first pass of the two pass approach. All of the other experiments are based on the hypothesis phoneme transcription of a corresponding first pass (i.e. their related General PF experiment) and can be considered as second pass. Additionally, as mentioned in the previous section, the *adapted* experiments used adaption techniques, based on these first pass recognition results.

In the next row Table 1, Table 2 and Table 3 show the results of the purely *phoneme specific model* (PSM). It appears to generate good results for the evaluated experimental conditions (with and without speaker adaption and for the reference as well as the hypothesis transcription) for SNR 0 dB and SNR 5 dB. Thus, as in previous analyzes we can clearly confirm, that particle filtering greatly benefits from phoneme specific speech models. The problem of *model tying* becomes evident at SNR 10 dB (i.e. the hypothesis experiments of the purely phoneme specific model perform worse than the general model, and its reference experiments perform better). However, in contrast to experiences from previous experiments, this effect does not appear for SNR 5 dB and SNR 0 dB in our system setup.

Mixed Speech Models

Next, let us have a look at the *mixed speech models* (see section 3.3). Contrary to earlier experiments (using different system setups), the phoneme specific results perform better than the results using the general model even for most of the hypothesis based experiments. Therefore, mixing the phoneme specific model with the (worse performing[4]) general model is not very promising. We tested a series of different values for the mixture weight α. The experiments use $\alpha = 0.5$ (equal weighting of both models), which appeared to be a good value. The results can also be found in Table 1, Table 2 and Table 3. The performance of the basic mixed model is slightly better than the general model approach (except for SNR 10 dB), but cannot reach the performance of the purely phoneme specific model in nearly all of the cases.

Confusability based mixed model

Most of the experiments using the *confusability based mixed model* PF (see section 3.4) show slightly better results, than the basic mixed model experiments. Even though, for the unadapted experiments the WER performances is mostly located between the one of the two mixture sources (i.e. general model and phoneme specific model), the adapted experiments show slightly better results than the general model as well as the phoneme specific model.

[4]We discovered, that the mixed models approach delivers good results for this system setup, if the training of the models is slightly modified by using samples of phoneme mids only (i.e. samples which do not belong to the beginning or the end of the articulation of a phoneme, and therefore are assumed to be more "precise"). For better comparability to the other approaches, the results presented here are based on the unmodified training procedure, using all of the extracted training samples.

SNR 0 dB				
	Unadapted		Adapted	
	Hypothesis	Reference	Hypothesis	Reference
General PF	56.6%		41.9%	
PSM	55.1%	52.9%	41.4%	39.8%
VoicedUnvoiced	55.1%	56.6%	41.3%	40.4%
VowelConsonant	57.0%	55.8%	41.2%	40.7%

Table 4: Experimental results SNR 0 dB

SNR 5 dB				
	Unadapted		Adapted	
	Hypothesis	Reference	Hypothesis	Reference
General PF	46.3%		36.1%	
PSM	45.5%	44.9%	34.5%	33.5%
VoicedUnvoiced	45.9%	45.7%	34.4%	35.1%
VowelConsonant	45.6%	45.9%	34.9%	34.4%

Table 5: Experimental results SNR 5 dB

SNR 10 dB				
	Unadapted		Adapted	
	Hypothesis	Reference	Hypothesis	Reference
General PF	40.5%		31.2%	
PSM	41.4%	39.4%	32.2%	31.2%
VoicedUnvoiced	41.3%	40.6%	32.3%	31.7%
VowelConsonant	40.9%	40.7%	31.8%	32.2%

Table 6: Experimental results SNR 10 dB

Knowlodge-based phoneme clustering

The knowledge based clustering into two classes, for *voiced and unvoiced phonemes* (see section 4.2.1), as well as for *vowels and consonants* (see section 4.2.2) reaches the performance of the purely phoneme specific particle filter results (see Table 4, Table 5, and Table 6). Therefore, it can be stated, that improvements over the traditional general model approach can be achieved with a much lower number of classes. So, it is possible to improve the PF performance up to a level, that is comparable to the results of the purely phoneme specific approach (consisting of 45 phonemes), only by a little more training effort (two classes instead of one).

Data-driven phoneme clustering

To evaluate the phoneme classes based methods, we start by analyzing the outputs of the hierarchical clustering algorithm (the corresponding dendrograms have already been shown in each of the distance's sections). In the following, we discuss their performance.

Clustering results
First, it becomes apparent, that the distance measures are a critical factor in building dendrograms by agglomerative hierarchical clustering. Some distances tend to generate a tree, which is very unbalanced, because many phonemes are merged to the same cluster over a series of continuous iterations of the algorithm (i.e. it results in very unevenly distributed class sizes). This problem depicts itself as *stairs-effect* in the dendrogram (e.g. see the Euclidean distance between samples, Figure 13). Such a case limits the number of potential phoneme clusters to a small number (i.e. we only evaluated two and three classes for the Euclidean distance between samples). There are distances where this effect is so severe, that it is even not possible to build a useful cluster tree, because in each iteration of the hierarchical clustering algorithm only one single cluster grows (e.g. the distance proposed in [HV07], and the generalization of the monogaussian distances to GMMs as described in section 4.3.7 turned out to be inoperative for clustering phoneme models).

Having a closer look at the clustering results shows, that the arrangement of phonemes differs significantly for the different distance measures. However, it can be recognized, that there are several characteristics that many cluster trees have in common. For example in each of the investigated cluster trees the phonemes S and Z are merged at leaf layer, furthermore the class { {AA, AW}, {AH, OY}, {AY, EH} } appears in all of the monogaussian dendrograms (i.e. D^{EUCL}, D^{KL}, D^{eMHN}) and is constructed exactly in the same order for each of the distances.

All in all it can be stated, that in most cases the phoneme model based clustering results show structures, which can (intuitively) recognized as similar sounding.

The confusability based clustering (Figure 14) reveals problems due to deviations of the reference alignment from the hypothesis. Thus, the two phonemes IX and NG which together form the suffix 'ing', or Y and UW which compose to the word 'you', appear to be interchanged very often. However, S and Z, which also show a high interchange rate, are certainly phonemes which are hard to be kept apart by the decoder.

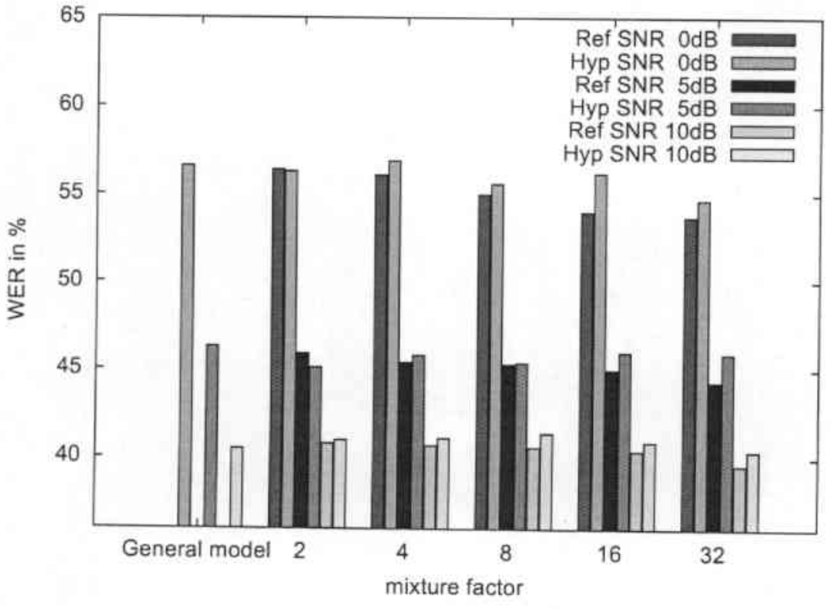


Figure 16: Kullback-Leibler distance on Gaussian mixture models, unadapted, 2, 4, 8, 16 and 32 classes in comparison to the traditional general model

Monogaussian Distances

For the distances based on the monogaussian approach (see sections 4.3.4, 4.3.5 and 4.3.6), we evaluated the *Euclidean distance* (D^{EUCL}), and the *extended Mahalanobis distance* (D^{eMHN}) for two classes, and the *Kullback Leibler distance* (D^{KL}) for three classes, because of the structure of its dendrogram (it actually consists of two approximately equal sized classes and the phoneme XM). The results of the monogaussian distances outperform the general model in all of our experiments for SNR 0 dB and SNR 5 dB, but can mostly not reach the performance of the purely phoneme specific approach. At SNR 10 dB, where the model tying effect occurs, they cannot compete with the general model, however the results are better than those of the purely phoneme specific approach. Thus, the monogaussian distances show performance improvements for evident noisy environments in comparison to the general model, and appear to be more robust against the model tying effect than the phoneme specific approach, only by enhancing the particle filter with one additional class.

Distances between Gaussian Mixtures

For the class of distances based on Gaussian mixtures, we first evaluated the *Earth Mover's distance* for two and three classes. The results in terms of word error rate can be found in Table 8, Table 9 and Table 10 at rows D^{EMD}.

AA, AW, AY, EH, AE, OW, AO, R, AH, UH, ER, OY, AXR, UW, AX, HH, CH, SH, EY, IX, XM, ZH															B, T, DH, W, D, K, P, F, TH, S, Z, G, N, JH, Y, IH, IY, L, XL, M, NG, V, XN																
AA, AW, AY, EH, AE, OW, AO, R							AH, UH, ER, OY, AXR, UW, AX HH, CH, SH, EY, IX, XM, ZH								B, T, DH, W, D, K, P							F, TH, S, Z, G, N, JH, Y, IH, IY, L, XL, M, NG, V, XN									
AA, AW, AY, EH			AE, OW, AO, R				AH, UH, ER, OY, AXR, UW, AX, HH				CH, SH, EY, IX, XM, ZH				B, T, DH, W				D, K, P			F, TH, S, Z, G, N, JH, Y						IH, IY, L, XL, M, NG, V, XN			
AA, AW, AY, EH			AE, OW		AO, R		AH, UH, ER, OY		AXR, UW, AX, HH		CH, SH		EY, IX, XM, ZH		B, T		DH, W		D, K		P	F, TH, S, Z		G, N		JH, Y		IH, IY, L, XL		M, NG, V, XN	
AA, AW	AY	EH	AE	OW	AO	R	AH, UH	ER, OY	AXR, UW	AX, HH	CH	SH	EY	IX, XM, ZH	B	T	DH	W	D	K	P	F, TH	S, Z	G	N	JH	Y	IH, IY	L, XL	M, NG	V, XN

Table 7: Kullback-Leibler distance between Gaussian mixtures - 2, 4, 8, 16, and 32 classes

They appear to be comparable to those based on the monogaussian distances, although exploiting some additional information, that the GMMs provide (i.e. the mixture weights are not considered by the monogaussian distances and the Euclidean distance even ignores the covariance matrices).

Anyhow, the overall best absolute value of a word error rate in our experiments can be found for the adapted reference of the Earth Mover's distance for three classes at SNR 10 dB (31.0%).

We further investigated a deeper analysis on the *Kullback Leibler distance on GMMs*, since among all studied distances the shape of its dendrogram is most similar to a balanced binary tree. So we evaluated 2, 4, 8, 16 and 32 classes. The compositions of these classes can be found in Table 7. Table 8, Table 9 and Table 10 show the corresponding results as D^{KLGMM}.

A potentially best number of classes cannot be determinated clearly. For the unadapted results on the reference, a definite trend for better performance with more classes can be recognized, however this mostly does not translate to the adapted experiments. Nevertheless, for the (most expressive) hypothesis experiments, nearly all of the best performing results (among all studied particle filter enhancement techniques) can be found in one of those experiments based on Kullback Leibler distance on GMMs. For example at SNR 5 dB on the hypothesis with speaker adaption using 8 classes, it could achieve a gain of 2.6% WER compared to a particle filter using the traditional general model. Especially it can outperform the traditional general approach and the purely phoneme specific approach (except for the hypothesis experiment at SNR 10 dB, where the general model could not be surpassed by any other approaches' hypothesis experiment).

SNR 0 dB					
		Unadapted		Adapted	
	#classes	Hypothesis	Reference	Hypothesis	Reference
General PF	1	56.6%		41.9%	
PSM	45	55.1%	52.9%	41.4%	39.8%
D^{EUCL}	2	55.7%	56.1%	41.5%	40.3%
D^{eMHN}	2	56.3%	56.3%	41.0%	40.8%
D^{KL}	3	55.8%	56.0%	42.2%	41.0%
D^{EMD}	2	56.3%	56.3%	41.0%	39.5%
D^{EMD}	3	56.4%	56.6%	40.5%	40.2%
D^{KLGMM}	2	56.3%	56.4%	41.6%	40.0%
D^{KLGMM}	4	56.9%	56.1%	40.9%	41.7%
D^{KLGMM}	8	55.6%	55.0%	41.4%	41.1%
D^{KLGMM}	16	56.2%	54.0%	41.4%	40.4%
D^{KLGMM}	32	54.7%	53.7%	41.3%	39.1%
$D^{euclSMP}$	2	57.0%	56.9%	41.9%	41.0%
$D^{euclSMP}$	3	56.2%	56.2%	42.2%	40.6%
D^{PIR}	2	55.9%	56.4%	42.0%	40.9%
D^{PIR}	3	56.1%	55.5%	41.7%	41.3%

Table 8: Experimental results SNR 0 dB

SNR 5 dB					
		Unadapted		Adapted	
	#classes	Hypothesis	Reference	Hypothesis	Reference
General PF	1	46.3%		36.1%	
PSM	45	45.5%	44.9%	34.5%	33.5%
D^{EUCL}	2	45.9%	45.0%	34.8%	34.7%
D^{eMHN}	2	45.6%	46.0%	35.0%	34.2%
D^{KL}	3	45.9%	46.1%	34.6%	34.2%
D^{EMD}	2	45.6%	46.0%	35.0%	34.4%
D^{EMD}	3	46.0%	44.7%	34.7%	34.0%
D^{KLGMM}	2	45.1%	45.9%	34.4%	34.0%
D^{KLGMM}	4	45.8%	45.4%	33.9%	34.3%
D^{KLGMM}	8	45.4%	45.3%	33.5%	34.4%
D^{KLGMM}	16	46.0%	45.0%	34.6%	34.0%
D^{KLGMM}	32	45.9%	44.3%	35.0%	33.8%
$D^{euclSMP}$	2	45.9%	46.6%	34.2%	34.2%
$D^{euclSMP}$	3	46.7%	45.7%	35.3%	34.8%
D^{PIR}	2	46.5%	45.7%	34.6%	34.3%
D^{PIR}	3	46.0%	45.4%	34.3%	34.3%

Table 9: Experimental results SNR 5 dB

Sample distance
The *Euclidean distance between samples* (see section 4.3.10) has already shown flawed clustering results. We evaluated the clustering into two and three classes, which have relatively unevenly distributed class sizes (e.g. in the case of two classes, one class consists of 37 phonemes and the other consists of 8 phonemes). Table 8, Table 9 and Table 10 again present the results ($D^{euclSMP}$). For nearly all of the analyzed experimental parameters, the performance of $D^{euclSMP}$ can be found among the worst results (e.g. it delivered 57.0% for unadapted, hypothesis with two classes at SNR 0 dB, which is the overall worst performance in the PF experiments test set). Therefore, in our experiments the Euclidean distance between samples is the least successful approach to build clustered phoneme models, but this points out, that the quality of clustering has an impact on the results of the recognition experiments. Nevertheless, $D^{euclSMP}$ can excel the traditional approach in single experiments.

Confusability clustering
The *confusability clustering* experiments, which are based on the phoneme interchange rate (introduced in section 4.4) can also be found in Table 8, Table 9 and Table 10 (D^{PIR}). Again, we evaluated two and three classes. The performance is below-average, which appears to be consistent to the problems, that have already been indicated by the clustering results. However, for most of the experiments, the word error rate can still be identified to be better than either the traditional general, or the purely phoneme specific model.

SNR 10 dB					
		Unadapted		Adapted	
	#classes	Hypothesis	Reference	Hypothesis	Reference
General PF	1	40.5%		31.2%	
PSM	45	41.4%	39.4%	32.2%	31.2%
D^{EUCL}	2	41.3%	40.6%	31.9%	31.3%
D^{eMHN}	2	41.0%	40.8%	32.3%	31.5%
D^{KL}	3	40.8%	41.5%	31.9%	31.7%
D^{EMD}	2	41.0%	40.8%	32.3%	32.0%
D^{EMD}	3	41.1%	40.0%	31.6%	31.0%
D^{KLGMM}	2	41.0%	40.8%	31.6%	31.7%
D^{KLGMM}	4	41.1%	40.7%	32.3%	32.1%
D^{KLGMM}	8	41.4%	40.6%	31.8%	31.7%
D^{KLGMM}	16	40.9%	40.4%	32.4%	31.7%
D^{KLGMM}	32	40.4%	39.6%	31.6%	31.9%
$D^{euclSMP}$	2	40.8%	40.7%	32.9%	32.2%
$D^{euclSMP}$	3	41.2%	40.9%	31.9%	32.0%
D^{PTR}	2	41.1%	40.1%	31.8%	32.3%
D^{PTR}	3	41.2%	40.5%	31.7%	31.7%

Table 10: Experimental results SNR 10 dB

6 Conclusion and Outlook

In this thesis, a series of new approaches to further improve the particle filter performance by phoneme specific, dynamically time varying speech models have been introduced and an experimental scenario has been set up to evaluate their performance. The focus has been set to data-driven, unsupervised phoneme clusters, which were generated by an agglomerative hierarchical clustering algorithm using different distance measures. In addition to that knowledge-based and confusability-based approaches have also been examined.

We tried a lot of different approaches and parameters to evaluate the performance of the proposed particle filter enhancements. The results show a high rate of variability, which makes it hard to derive clear qualitative statements for comparing the different approaches. These problems seem to be only partly inherent in the indeterminism of the particle filter approach. Additional variabilities come from the language model, the speaker adaption techniques and also the pre-processing stage appears to be a highly critical factor. For that reason, the experimental results might not directly translate to alternative scenarios or processing in different dimensionalities of the feature space. To get more stable results, further analyses, using a much larger data set, had to be used, which was not possible within the scope of this thesis.

The different distances lead to very different clustering results, but this does not directly transfer to the word error rates and no single best solution could be determined, that stands out from the other approaches. Hence, it might be necessary to perform an additional investigation, to find out which approach achieves the best possible speech recognition performance for the application's special environmental conditions and setup.

However, we clearly reconfirmed, that a particle filter based approach does a very good job to improve the recognition accuracy in environments afflicted with highly non-stationary, additive noise. To conclude we can state, that in nearly all cases, the performance of one of the proposed approaches could outperform either the traditional general or the purely phoneme specific model, and the overall best performance could be meliorated by the use of class-based phoneme models.

Outlook

Finally, this section introduces some additional ideas, to develop further the phoneme specific particle filter design.

The articulation of a phoneme can vary within its duration, so that the onset and offset might differ considerably. Additionally, for spontaneous and fluent speech, phonemes can not be treated as separate units, but flow into each other (coarticulation). Because of that, a phoneme is often represented by a 3-state Hidden Markov Model in ASR systems (modeling its begin, middle and end). Furthermore context dependent probability distributions are used for each state. By now, our speech models for particle filtering neglected the fact of fluctuating articulation within a phoneme and its context. A phoneme clustering with respect to begin, middle and end of a phoneme, or clustering based on context dependent sub-phonemes could be more fine granular, reduce variabilities, and lead to better matching to the acoustic models used in the decoding pass.

Another refinement can be applied to manage the classification of the particle filter: Instead of using a single best hypothesis transcription of a first recognition pass, the classification can be made by using phoneme transcriptions based on the n best hypotheses or lattice, derived from the wordgraph of the (first pass) decoding stage. This way, additional information about alternative hypotheses and their uncertainty can be transfered back from the ASR system into the particle filter.

It even might be possible to overcome the two phase approach by fast heuristics to detect classes without a previous recognition pass (e.g. use of an voiced/unvoiced speech detection as classifier), since we experienced improvements in the performance of the particle filter even with a small number of classes.

The methods proposed in this section show, that there are still many opportunities to exploit the full potential of phoneme specific particle filters for speech feature enhancement. Future research efforts can improve their performance and stability and help to further reduce the impact of noisy environments on speech recognition.

A Phonemes in JRTk

Phoneme	Words
AA	**ar**m, **ar**ticle
AE	**a**venue, **a**xe
AH	**a**bout, **a**bove
AO	**aw**esome, f**or**ce
AW	b**ou**nce, d**ow**n
AX	**a**ccount, **a**lert
AXR	capt**ure**, lit**er**, ...er
AY	m**i**ke, ps**y**cho
B	**b**rain, a**b**out
CH	**ch**ain, **ch**icken
D	**d**evelopment, **d**estiny
DH	**th**e, **th**ank
EH	**e**rror, **e**xcellent
ER	v**er**sus, t**er**m
EY	w**eigh**t, t**a**ke
F	**f**ilter, **f**lag
G	**g**old, **g**un
HH	**h**ack, **h**ammer
IH	h**i**story, **i**mage
IX	**i**llusion, **i**ntensive, ...ing
IY	jewler**y**, magaz**i**ne, ...ty
JH	ma**j**or, mer**ge**
K	mi**c**ro, **k**ill
L	**l**ong, **l**ife
M	**m**an, **m**anual
N	**n**ovel, **n**ice
NG	la**ng**uage, ba**n**k, ...ing
OW	b**o**ld, c**o**de
OY	depl**oy**, app**oi**ntment
P	**p**ittsburgh, **p**arty
R	**r**eason, **r**ecord
S	**s**enior, **s**etup
SH	**sh**ield, **sh**ort
T	**t**ime, **t**oday
TH	**th**umb, **th**eatre
UH	w**ou**ld, l**oo**k
UW	y**ou**, l**oo**se
V	o**v**er, pro**v**ider
W	q**u**een, **w**ay
XL	ab**le**, ang**le**
XM	rhyth**m**, touris**m**
XN	certa**in**, but**on**
Y	**y**oung, **y**ear
Z	advi**c**e, i**s**, ...s
ZH	mea**s**ure, u**s**ual

References

[AMGC02] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[Ass] International Phonetic Association. The international phonetic alphabet. http://www2.arts.gla.ac.uk/IPA/ipachart.html.

[BDH03] M. Bolic, P. Djuric, and S. Hong. New resampling algorithms for particle filters. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2:589–592, April 2003.

[BdMCAM99] P. Boula de Mareuil, C. Corredor-Ardoy, and Adda-Decker M. Multi-lingual automatic phoneme clustering. *International Congress of Phonetic Sciences*, pages 1209–1212, 1999.

[DGW00] A. Doucet, S. J. Godsill, and M. West. Monte carlo filtering and smoothing with application to time-varying spectral estimation. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2:701–704, June 2000.

[DHS01] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley, 2. ed. edition, 2001.

[Eph92] Y. Ephraim. Statistical-model-based speech enhancement systems. *Proceedings of the IEEE*, 80(10):1526–1555, Oct 1992.

[Fau06] F. Faubel. Speech feature enhancement for speech recognition by sequential monte carlo methods. Master's thesis, Universität Karlsruhe (TH), 2006.

[FW06] F. Faubel and M. Wölfel. Coupling particle filters with automatic speech recognition for speech feature enhancement. *Interspeech*, 2006.

[FW07] F. Faubel and M. Wölfel. Overcoming the vector taylor series approximation in speech feature enhancement - a particle filter approach. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007.

[GA05] J. Goldberger and H. Aronowitz. A distance measure between gmms based on the unscented transform and its application to speaker recognition. *Proceedings of Interspeech 2005*, September 2005.

[GSS93] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140:107–113, April 1993.

[HUS05] R. Haeb-Umbach and J. Schmalenstroeer. A comparison of particle filtering variants for speech feature enhancement. *Proceedings of Interspeech 2005*, pages 913–916, 2005.

[HV07] M. Helen and T. Virtanen. Query by example of audio signals using euclidean distance between gaussian mixture models. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007.

[JS04] Qin Jin and T. Schultz. Speaker segmentation and clustering in meetings. *Proceedings of International Conference of Spoken Language Processing*, October 2004.

[Kim98] Nam Soo Kim. Imm-based estimation for slowly evolving environments. *IEEE Signal processing letters*, 5(6):146–149, June 1998.

[LS01] B. Logan and A. Salomon. A music similarity function based on signal analysis. *IEEE International Conference on Multimedia and Expo*, 2001.

[Mor96] P. J. Moreno. *Speech Recognition in Noisy Environments.* PhD thesis, Carnegie Mellon University, May 1996.

[MRS96] P. J. Moreno, B. Raj, and R. M. Stern. A vector taylor series approach for environment-independent speech recognition. *Proceedings of ICASSP*, 1996.

[PB87] K. Paliwal and A. Basu. A speech enhancement method based on kalman filtering. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 177–180, April 1987.

[Pro] The Freesound Project. garbage.coll.serv.ds70p.mp3. http://freesound.iua.upf.edu/samplesViewSingle.php?id=6986.

[RN03] S. J. Russell and P. Norvig. *Artificial intelligence.* Prentice Hall, 2. ed., internat. ed. edition, 2003.

[RSS04] B. Raj, R. Singh, and R. Stern. On tracking noise with linear dynamical system models. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:I–966 – I–968, May 2004.

[RTG98] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 59, Washington, DC, USA, 1998. IEEE Computer Society.

[SK06] T. Schultz and K. Kirchhoff. *Multilingual speech processing.* Academic Press, 2006.

[SR03] R. Singh and B. Raj. Tracking noise via dynamical systems with a continuum of states. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:I–396 – I–399, 2003.

[UNGH98] N. Ueda, R. Nakano, Z. Ghahramani, and G. E. Hinton. Split and merge em algorithm for improving gaussian mixture desity estimates. *Neural Networks for Signal Processing VIII, 1998. IEEE Signal Processing Society Workshop*, 1998.

[WM05] M. Wölfel and J. McDonough. Minimum variance distortionles response spectral estimation. *IEEE Signal Processing Magazine*, pages 117–126, September 2005.

[YN02] K. Yao and S. Nakamura. Sequential noise compensation by sequential monte carlo method. *Advances in Neural Information Processing Systems 14. MIT Press*, 2002.