Acoustical Modeling for Speech Recognition: Long Units and Multi-Modeling

Ronaldo Messina

ACADÉMIE D'AIX-MARSEILLE
UNIVERSITÉ D'AVIGNON ET DES PAYS DE VAUCLUSE

THÈSE

Presentée pour obtenir le grade de Docteur en Sciences de
l'Université d'Avignon et des Pays de Vaucluse

Acoustical Modeling for Speech Recognition: Long Units and Multi-Modeling

SPÉCIALITÉ : Informatique
par

Ronaldo Oliveira Messina

Soutenue publiquement le 05 Décembre 2005 devant le jury composé de :

M.	Pietro Laface	Prof., Politecnico di Torino, Torino, Italie	Rapporteur
M.	Christian Wellekens	Prof., Institut Eurécom, Sophia-Antipolis, France	Rapporteur Président du jury
M.	Stéphane Dupont	Multitel, Mons, Belgique	Examinateur
M.	Denis Jouvet	France Télécom R&D, Lannion, France	Examinateur
M.	Renato De Mori	Prof., Lia, Avignon, France	Directeur de thèse

---

Laboratoire Informatique d'Avignon

Abstract

This thesis' theme is acoustic modeling. We follow the dominant paradigm, hidden Markov models, but we try to explore alternative approaches to increase performance of speech recognition systems. Firstly, we tackle the problem of the choice of modeling unit; for large vocabulary recognition, sub-word units are used so there is no limitation to the size of the lexicon, as long as it remains possible to describe every word with the chosen units. The second approach is the use of multiple models, each adapted to a particular source of variability, to make the recognition more robust to those variabilities.

For the choice of units, we assess the use of "long-units" (sub-word units that span several phonemes); we adopt these units because it is believed that variations in the way each phoneme is spoken due to its "context", i.e. the neighboring phonemes, could be absorbed by the longer time span of the units. Moreover, we make the units dependent on the context as well, to capture the residual variabilities and increase modeling power. To the best of our knowledge, this is the first work that models long-units that are dependent on the context. There are two types of long-units in this work: syllable-like units and an automatically-derived set of units base on mutual information measure. We propose solutions to the problems of estimating the huge number of parameters needed for the longer units and also to the problem of dealing with unseen units (in the case of syllable-like units), that may be needed for a particular lexicon. The results show that high performance can be attained, at the same level or higher than usual allophonic units.

We explore different possible sources of variability that could impair the recognition results: speaker gender, signal-to-noise ratio and we propose the alternative average vocalic energy (AVE) which should be more reliable because there is no need to detect non-speech segments, transmission channel (wired and wireless) and phoneme position within a word. Some sources could be combined, e.g. gender dependent models that are also channel specific. Then we propose

different combination methods to mix the different models in each variation class; note that there is no combination needed for the position dependent modeling, as it is dealt with in the lexical descriptions. We mix models at syntactical, lexical and acoustical levels. The results show that mixing at acoustical level yields the highest performance; regarding the variability, gender+channel dependent models have the lowest error rates, followed by gender and AVE dependent models.

Contents

List of Figures

List of Tables

Acronyms

The following acronyms are used throughout this thesis.

AM Acoustical Model

AMo Acoustical Modeling

ASR Automatic Speech Recognition

AVE Average Vowel Energy

BIC Bayesian Information Criterion

CD Context-dependent

CDHMMs Continuous density HMMs

CI Context-independent

CMS Cepstral Mean Subtraction

DCT Discrete Cosine Transform

EM Expectation-Maximization

FFT Fast Fourier Transform

FSN Finite State Network

GSM Global System for Mobile Communications

HMM Hidden Markov Model

HMMs Hidden Markov Models

iid Independent and identically distributed

LM Language Model

LMo Language Modeling

LU Long Unit

LUs Long Units

MCE Minimum Cumulative Error

MDL Minimum Description Length

MFCC Mel-Frequency Cepstral Coefficients

MI Mutual Information

ML Maximum Likelihood

MLE Maximum Likelihood Estimation

MLLR Maximum Likelihood Linear Regression

MMI Maximal Mutual Information

pdf Probability Density Function

PSTN Public Subscriber Telephone Network

ROS Rate Of Speech

SCA Sequential Clustering Algorithm

SNR Signal-to-Noise Ratio

TD Temporal Decomposition

VTLN Vocal Tract Length Normalization

WER Word Error Rate

Introduction

The desire to make computers recognize, and ultimately understand, speech has been around for some time. Currently, there are many practical applications for speech recognition such as dictation and database access; for these (and forthcoming) applications high recognition performance is required. High performance is a primary requisite because users are very demanding on the level of performance they expect from any automatic system and also because it makes understanding easier as there are fewer word errors in the recognized string, which is later semantically analyzed to retrieve its meaning.

This thesis follows the current dominant paradigm in automatic speech recognition (ASR) that is parametric statistical recognition of speech using Hidden Markov Models (HMMs). Under this paradigm, speech is considered to be a piecewise stationary process that can be well approximated by a first-order Markov model. ASR systems using HMMs deliver high recognition performance in a controlled environment, but there are many factors limiting their use in an unrestricted environment such as:

- *Noise robustness*: Performance degrades when the signal-to-noise ratio (SNR) is significantly different from the one used during parameter estimation (also called *training*);

- *Transmission channel*: Although this item could be placed under *noise robustness* or *speech variability*, we would like to emphasize that the channel also plays a role in limiting performance. It is known that simply changing the microphone could degrade the recognition performance; this happens all the time when the system deals with telephone-quality speech, particularly when mobile or cordless telephones are employed.

- *Speech variability*: Speech characteristics may change up to an extent that makes recognition too difficult for an automatic system; changes can be in speaking rate (normal, slow, fast), accent (regional or non-native), mode (read, spontaneous, prepared);

- *Speaker changes*: ASR systems, most of time, need to be adapted to a particular speaker's voice to attain higher performance.

Modeling and Units

In statistical speech recognition with HMMs, the recognition process finds the most likely sequence of words $\hat{W}$, given the acoustical evidence (or *observation*) A:

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|A)$$

This equation can be transformed into an *a posteriori* formulation (here it is assumed that $P(A)$ is constant for a given sequence of words):

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(A|W)P(W)$$

The term $P(A|W)$ is calculated using HMMs and it is given by the so-called *acoustical model* part of an ASR system; the other term, $P(W)$, represents the *language model*. Acoustical modeling (AMo) involves all that is related to the choice of units, how they are interconnected, which contexts are considered, how the parameters of the models are estimated and adapted (if needed). In this work, continuous density HMMs are employed and the kind of density that is used is a Gaussian mixture model. Even though language modeling (LMo) is an important and active area of research, it will be of small concern here; the language model (LM) gives the probabilities of sequences of words and is represented either by regular or statistical grammars. In the present work the LM is represented by a compiled regular grammar that describes a recognition task, such as the recognition of telephone numbers.

The acoustical evidence A is a sequence of *feature vectors* (see Section 1.1) that is calculated by sliding an analysis window over the speech signal and then analyzing the signal as seen through each window at each time step; this sequence of vectors reflects the piecewise stationarity assumption.

Speech is modeled through a concatenation of basic (atomic) units that builds up more complex constructs such as words and phrases or sentences. One instance of basic unit could be a word; this alternative is adequate for small vocabularies (up to a few hundreds of words) but as the vocabulary grows, it becomes harder to properly estimate the model parameters because

there will be a huge number of parameters to estimate from a limited amount of data. The problem becomes simpler if smaller-than-word units are used. Any word can then be added to the vocabulary by simply "constructing" a model of the word by concatenating the models of its constituent units; whereas in the word-model approach, adding a new word implies collecting data and estimating a model for the new word. Obviously a "pronunciation" lexicon, or a dictionary, is required for the words and their description in terms of those basic units.

As a consequence, phonemes seem to be a good choice as units; but one must keep in mind that phonemes are abstract entities and human speech is not made up of sequences of phonemes. The sounds that are produced are called *phones*, which are the actual physical realization of the phonemes. The production of a phoneme y is subjected to the surrounding phonemes; these phonemes are called the *context* in which y appears. The phone is modified by its context because the articulators (jaws, tongue, lips) cannot move instantaneously from one position to another. The brain coordinates the movements of the articulators, making them move smoothly to produce intelligible speech and we call this phenomenon *co-articulation*. The many different contextual realizations of one phone are collectively known as *allophones* (always related to a given phone). Co-articulation effects make it necessary to model speech with a high level of detail to absorb as much as possible the variation that is observed for each phoneme unit. Therefore, the use of context-dependent (CD) units, such as allophones, is a better choice than the use of context-independent (CI) phones because they enable the system to deal with co-articulation.

However, modeling speech with CD units is not enough to capture all the variability present in the speech signal. As said in the beginning of this chapter, there are factors other than co-articulation which cause the speech signal to vary. Some of the factors can be dealt with by pre-processing the speech signal; noise reduction techniques could also be used to increase the SNR and equalization techniques could reduce the degrading effects of the transmission channel. Those factors can be dealt with at the model side as well; currently, model adaptation seems to be the most efficient technique to cope with variability; models can not only be adapted to a specific speaker's voice, but also to a given channel and even to a different vocabulary (task adaptation).

The following section describes the approaches that are taken in this thesis and what are the problems in speech recognition that we are interested into.

Tackled problems

The main theme of this thesis is acoustic modeling and there are two problems of concern: the choice of units (and their representation) and what types of models can be used to assimilate the variability present in the speech signal.

Phoneme-sized context-dependent units cannot absorb all the co-articulation effects because these effects usually span a longer term than those units. The use of longer units like syllables or multi-phoneme units makes it possible to capture the long term correlations that are present in the sequence of feature vectors. However, this approach is not free of issues; even if the units are longer than phones the co-articulation effect is still present in the transition from one unit to another. Moreover, there is a large number of syllable (and syllable-like) units in most languages; the problem of estimating a huge number of parameters from limited data shows up and has to be circumvented. Another issue that arises is what to do when a unit (syllable) is necessary to model a new recognition task, but due to the limited amount of training data this unit was not present in the data; these units are called *unseen* or *missing* units.

In what concerns the type of models, the use of multiple models (*multi-modeling*) for the units is adopted. The approach that is followed consists in selecting some "classes" (or sources) of variability in an *a priori* way, then estimate specific model parameters from the (homogeneous) data, to create finely tuned models to one specific kind of variability. It is expected that the models "reject" data from other sources, i.e. the likelihood for those data should be much lower than the likelihood for "matched" (coming from the same source) data. The issues that arise are: how to combine the different models to provide one set of "generic" models and which classes are worth modeling. The multi-modeling approach can also be seen as asking what is worthier: to have many models per unit or a single but detailed model. Solutions for all the issues previously listed are presented and evaluated as outlined in the next section.

Thesis outline

This thesis is organized as follows. Chapter 1 describes in detail the acoustic modeling aspects under the paradigm of HMMs and the preliminary work that was done to simplify some of the forthcoming issues, followed by the baseline results that are used to assess the quality of the novel solutions. This chapter also presents the databases used to estimate model parameters and the tasks used to measure the performance of the resulting recognition systems.

The use of longer units is not completely new, so a literature review is presented in chapter 2. This chapter also presents an analysis of the existing solutions and their shortcomings and concludes with the motivations for the present work. Chapter 3 presents how syllable(-like) and multi-phoneme units are employed in an ASR system and how we can circumvent the issues of parameter estimation and unseen units.

Using multiple models is an idea that appeared in some form or another in the literature as presented in Chapter 4. Again, an analysis of the existing solutions and limitations is presented along with the motivations for the present thesis. The choice of *a priori* variability classes and model combination schemes are studied in Chapter 5; performance assessment and analysis of the results are presented as well.

Finally, a summary of the contributions and directions for future work are presented in Chapter 6.

Chapter 1

Acoustic Modeling

This chapter is an overview of Acoustic Modeling (AMo) in speech recognition; the items presented are those of interest for this work and are not exhaustive. The organization of the present chapter is as follows. First we describe the ASR system used in this work and Section 1.1 briefly reviews the feature extraction process performed by the front-end; then the back-end is described in Section 1.2. A brief recall of the theory behind Hidden Markov Models (HMMs) is given in Section 1.3, the most relevant part is parameter estimation. Then Section 1.4 describes the speech corpora that are used to estimate parameters and to assess performance and Section 1.5 presents the training procedure; this procedure is used to estimate parameters for all the different models studied and in Section 1.5.4 we briefly review class-based model adaptation. Context-dependent units and issues that arise with their use are discussed in Section 1.6; finally Section 1.7 presents the reference (baseline) results for the different corpora.

Automatic Speech Recognition (ASR) systems using Hidden Markov Model (HMM) are divided into two main blocks: the *front-end* and the *back-end*. The front-end transforms the speech acoustic waveform into a sequence of observation vectors (also called feature vectors) that are fed into the back-end. There are two modes of operation for the back-end: in the first mode the parameters of the HMMs are estimated and this mode is also called *training*; the second mode represents the actual process of recognizing speech and is called *decoding*. The next sections present these blocks: the whole system used in this work is illustrated in Figure 1.1.

It is usual to classify the systems (or tasks) with respect to the "type" of speech they recognize; *isolated* or *continuous* speech. Isolated speech refers to systems that recognize one word at a time, surrounded by some periods of silence (or background noise); continuous speech refers to the recognition of a series of words, with optional pauses in-between them. In this work we

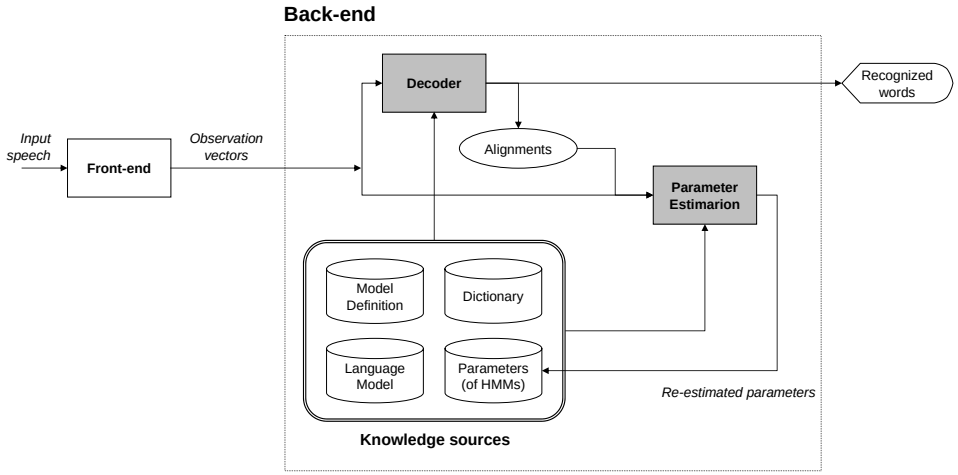


Figure 1.1: Block diagram of a speech recognition system.

have many small vocabulary (up to a few hundreds of words) isolated word tasks such as digit and number recognition and a large vocabulary task (i.e. recognizing city names); there is one continuous speech task, namely telephone numbers; these tasks are not the hardest that we may encounter in practice but the latter two are complex enough to assess the performance of the models. Other large scale continuous speech tasks such as broadcast news or dictation are very dependent on the language model and rejection strategies; so these tasks are not studied to limit the "field of action" of the present work.

1.1 Front-End

The front-end transforms the speech waveform into a sequence of feature vectors that represents the input utterance and that is modeled by a HMM. The feature vectors should, ideally, remove all information that is not important to recognition. Examples of unimportant information are speaker anatomical characteristics, channel frequency response and all kinds of non-speech noise. There are some techniques to reduce the irrelevant information present in the signal. For the anatomical characteristics Vocal Tract Length Normalization (VTLN) [Lee 1996, Zhan 1997] can be used; [Gong 1995] reviews speech recognition in noisy conditions. As the channel is not likely to change during a recognition session, a simple blind equalization scheme called Cepstral Mean Subtraction (CMS) [Rosenberg 1994, Junqua 1995] has become standard in speech recognition; however this technique needs silence/speech detection to be functional.

The speech signal is divided into a series of overlapping windows (*frames*) and then the portion of signal seen through each window is analyzed; a feature vector is the result of the

analysis. To compensate for the negative slope of the spectrum during vowel-like sounds and to boost the energy in high frequency bands (particularly in the case of consonant-like sounds), the signal passes through a pre-emphasis filter. This filter is a first-order high-pass filter given by $1 - az^{-1}$, with a very close to one (the actual value depends on the sampling rate). The window that is employed to select (and weight) a portion of the signal (N points) is usually of the form $w[k+1] = \alpha - \beta \cdot \cos(2\pi \frac{k}{N-1})$, where $\alpha + \beta = 1$; the most common window is the Hamming window which has $\alpha = 0.54$ and $\beta = 0.46$ but the Hanning window $\alpha = \beta = 0.5$ is also used. The usual length of the window is about 10-32 ms and successive windows overlap by a factor in the range 40-60%. In this work the window has a length of 32 ms and the offset is 16 ms (or 50% overlap), which results into windows of 256 points shifted by 128 points (for telephone speech, sampled at 8000 samples per second); this also provides some computational advantage for the Fast Fourier Transform (FFT) calculations.

There are many methods to analyze the speech signal; during the 70's the most popular was linear prediction analysis [Makhoul 1975] and the derived cepstral coefficients [Rabiner 1993]. Since the mid-80's the *de facto* standard representation is the so-called Mel-Frequency Cepstral Coefficients (MFCC) [Davis 1980]; there is another popular front-end that is based on a perceptual linear prediction analysis [Hermansky 1990]. Both methods warp the speech spectrum into a perceptually-oriented scale and then the spectrum is smoothed with a bank of filters that mimic some of the characteristics of the human hearing. Finally the smoothed perceptual spectrum is transformed by a non-linear function and the cepstral (cepstral analysis was introduced in [Tukey 1962]) coefficients are derived. In this work we use MFCC and the sequence of processing steps is described next.

The spectrum of the emphasized and windowed speech signal is calculated with the aid of a FFT (the power spectrum is given by the square of the FFT's magnitude); then the spectrum passes through a bank of triangular filters that are equally spaced on the Mel-scale. The Mel-scale is related to the usual linear scale (in Hz) by the following equation,

$$f_{Mel} = 2595 \log_{10}(1 + f_{Hz}/700)$$

In the linear (Hz) scale, the filters are linearly spaced up to 1 kHz and up from there the center frequency and bandwidth of the filters are logarithmic increasing. Then, the log-energy of each filter is calculated and a Discrete Cosine Transform (DCT) of the log-energies provides the

MFCC. The MFCC are calculated from the log-energies E_k by the equation below [Davis 1980],

$$\mathrm{MFCC}_i = \sum_{k=1}^{N} E_k \cos\left(i\pi \frac{k-\frac{1}{2}}{N}\right)$$

where N is the number of filters (usually in the range 20-25) and i goes from $1, \cdots, M$, M being the number of coefficients.

It was found [Davis 1980] that the direction cosines of the eigenvectors of the covariance matrix of the log-energies and the cosine expansion are very similar. Moreover, the DCT has the nice property of making the coefficients quasi-uncorrelated. It is then possible to retain just a few terms of the cosine transform to represent most of the variation of the log-energies. The result is a compact representation of speech as just as few as 8 "numbers" are enough; note that the zeroth order coefficient is related to the energy of the frame and it is often replaced by another measure of energy; these parameters form the so-called *static* feature vector).

As the frames overlap, the feature vectors are quite correlated, but in HMM modeling it is assumed that the vectors are independent. It is common practice to append estimates of the first and second derivatives (the *dynamical* features) of the static feature vector. This idea was first used in [Furui 1981] in speaker recognition and later adopted as HMMs become the leading modeling paradigm. In [Rabiner 1993] formulas for the first and second derivatives based on polynomial fitting for the statical features are given; those formulas are used because they are less noisy than other approximations. So the feature vectors that are actually modeled are the statical vector (MFCC plus energy) appended by their first and second order derivatives. To make things clear, imagine that the dimension of the vector formed by the energy measure and the MFCC is D, then the dimension of the final vector that is modeled is $3D$. The whole process is illustrated in Figure 1.2; the images on the right side of the blocks illustrate the aspect of the inputs/outputs at each stage for a vocalic sound.

1.2 Back-End

In the back-end there are two modes of processing; in one mode, the parameters of the HMMs are estimated (training) and in the other mode, the recognition itself takes place (recognition or decoding). In this block the models for the speech units are stored and there are many information sources such as the model description, which includes the number of states and transitions, how the states are connected through the transitions (topology), and the contexts

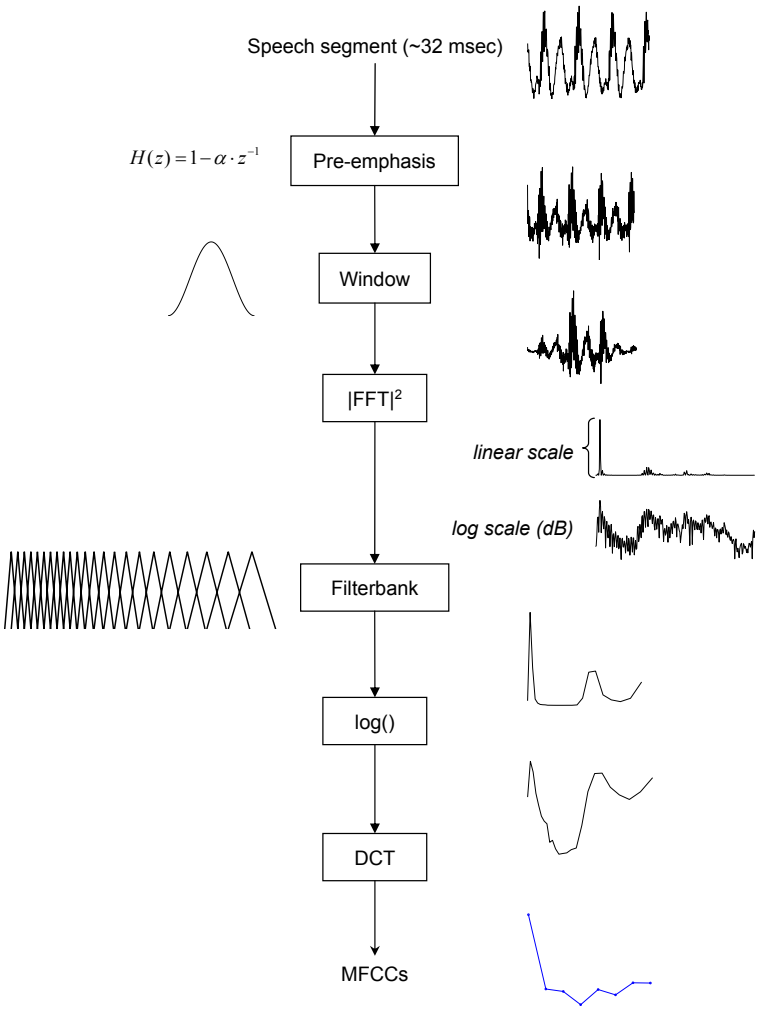


Figure 1.2: Feature extraction: Mel-Frequency Cepstral Coefficients.

that are possible (how the units are connected); another information source is the pronunciation dictionary which indicates how the words are described using the basic units, this dictionary may be just a simple list of word and pronunciations or may be a rule base system (with the aid of an exception list). The last information source that is used is the language model, which describes the possible word sequences. In summary, the language model defines what can be said, the pronunciation defines how it can be said and the model definition says how to model what is said.

In the ASR system used for this work, the language model is represented by a rule-based grammar that describes a given recognition task; a grammar compiler [Brown 1991] integrates the information from the model definition and the pronunciations with the task grammar and builds up a Finite State Network (FSN) that represents the task; note that it is possible to

represent statistical language models like bi-grams [Jelinek 1997] with a FSN (representing tri-grams or longer requires large amounts of memory, even for small vocabularies). This FSN is first determinized and then it is minimized [Hopcroft 2001] to provide a fast and compact representation for computational purposes. There are three levels of abstraction in the FSN:

- Syntactical: how words are connected (Language Model);
- Lexical: how units are connected to build up words (Pronunciation); and
- Acoustical how states are connected to build up units (Acoustical Model).

Label markers are added to the FSN (markers are inside the boxes in Figure 1.3, from "0" up to "9") so the FSN becomes a transducer from the speech signal into words.

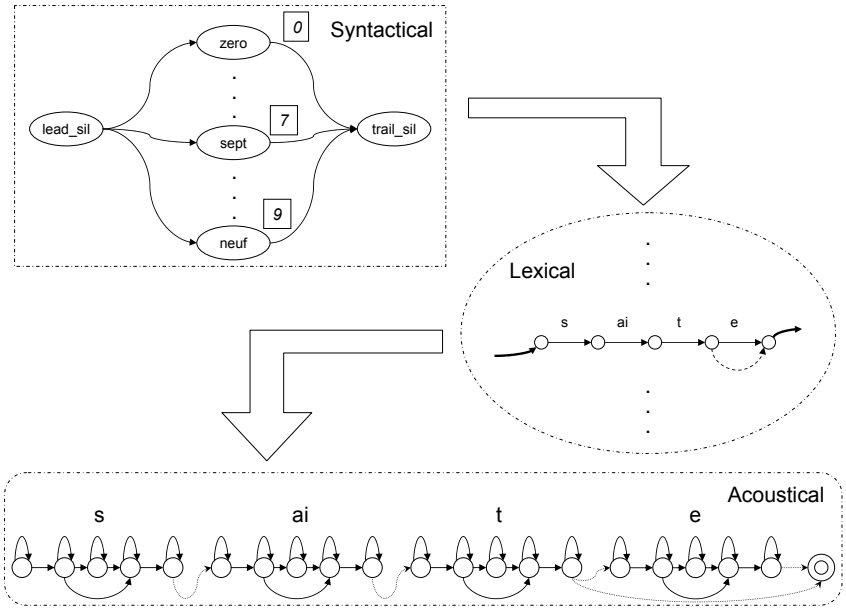


Figure 1.3: Three levels of abstraction of the FSN.

Figure 1.3 depicts the levels for a grammar describing an isolated digit recognition task (given below); each level represents one of the information sources.

```
TERMINALS = {
    0, 1, 2, 3, 4, 5, 6, 7, 8, 9;
}
NON_TERMINALS = {
    VOC = 0 + 1 + 2 + 3 + 4 +
```

```
          5 + 6 + 7 + 8 + 9;
}
SPEECH = LEADSIL . VOC . TRAISIL;
```

Where the separators "+" and "." represent an alternative between two tokens and concatenation, respectively; the symbols `LEADSIL` and `TRAISIL` represents the leading and trailing silence, respectively.

There are two processing blocks in the back-end: the *Decoder* and the *Parameter Estimator*. The "Decoder" block receives the sequence of feature vectors from the front-end and produces the sequence of recognized words and, if desired, the alignment of each vector with the units and the densities of the HMMs; these types of output and their use will be explained later in Section 1.5. Explaining the details of a decoder algorithm is out of the scope of the present work, though we depend on it to train models, ; the details can be found in the two excellent review articles [Ney 2000, Deshmukh 1999].

The "Parameter Estimator" block has as inputs the feature vectors and the alignments from the "Decoder", and from a set of initial parameter values (initial models) it estimates new values for the parameters and uses these values as initial values for a new iteration. The process iterates until a stopping criterion is satisfied; usually the stopping criterion is a threshold on likelihood increase. Training is also stopped if a maximum number of iterations is reached. The most common estimation scheme is Maximum Likelihood Estimation [Rabiner 1993, Jelinek 1997] (MLE) which gives parameter values that maximize the likelihood of the training data given the models. In this scheme each model has its parameters estimated in isolation from the others (with the data corresponding to the modeled unit), and the models do not discriminate between units. Discriminative parameter estimation calculates parameter values with the data from all units; the parameters are calculated so the models can better "separate" one unit from another. There are many criteria for discriminative parameter estimation, the most popular are Minimum Cumulative Error (MCE), Maximal Mutual Information (MMI); for further information on discriminative methods in parameter estimation for speech recognition see [Kapadia 1998, Valtchev 1995, Gopalakrishnan 1991, Juang 1997]. We will not further discuss discriminative techniques because it is out of the scope of this thesis. The next section describes Hidden Markov Models and also discusses MLE.

1.3 Hidden Markov Models

Even for the same person uttering the same word, the duration of the resulting signal will be very variable. Hidden Markov Models (HMMs) are good models because they can accommodate sequences of different lengths that are the result of the same process.

HMMs are statistical models which consist of states connected by transitions; there is a probability attached to each transition. HMMs can be seen as generative models, where at each time step, an output symbol (*observation*) is produced (emitted) with a given probability. During decoding it is easier to interpret the process as "consuming" one symbol with a given probability. The observation production (consumption) can be attached either to the states either to the transitions. Both views are valid and completely dual [Jelinek 1997], but in the literature the emission is commonly attached to the states. Attaching the emission to the transitions is more appropriate within the FSN/transducer view of the recognition process (inputs are feature vectors and outputs are sequences of words); hereafter the terms related to the emission/consumption of the observations are going to be used interchangeably, the meaning should be clear from the context. It is also possible to have non-emitting transitions (resp. states) that do not produce any observation. The non-emitting (or empty) transitions (resp. states) allow to change state within the same time step; this is also used in a transducer when it is not desired to put a label marker on some of its transitions. In the rest of the discussion we will keep the emissions on the transitions point of view, as implemented in our system.

Due to the sequential nature of the speech signal, it is usual to have transitions only from left-to-right (as in the Bakis' model [Bakis 1976]). The topology of the model should be consistent with the type of unit that is modeled and also with the time step of the observations, given by the front-end. Short term units (like phones) should have a model that can be traversed by consuming a smaller number of observation than word models that should consume more observations depending on its average (and also on the minimal) duration. So phone models can be traversed with as few as 3 to 5 observations and word models take a number of observations proportional to the number of phonemes in the word. Note that if the units are context-independent (CI) or context-dependent (CD), the topology has to be adapted accordingly. CD units have more entry/exit states, depending on the context width, i.e. to the number of preceding/following phonemes that are considered to have an effect on the current phoneme. Figure 1.4 illustrates the topology that is adopted in this work for CI phone models; note that the transitions are

named (e.g. "*fl*", "*fc*" etc.) those names are used to identify the density functions associated to the transitions; the full naming convention for the densities is "*unit name.transiton name*".

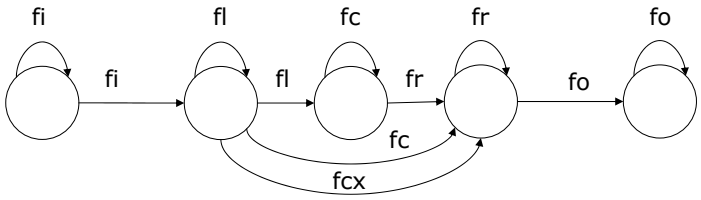


Figure 1.4: Topology of a CI phone model.

The transitions with densities named "*fc*" and "*fcx*" are used to shorten the model, i.e. the model can be traversed consuming a minimum of 3 observation vectors. There are also models for "silence" (leading and trailing silence have their own models) and there is a model for short pauses between words. There is no temporal structure to model during silence (actually, non-speech or background noise) segments, therefore the topology can be just a single state (with a self-loop) or a sequence of few states to enforce a minimal duration.

It was said before that a density function is associated with each transition; this function gives the values of emission probabilities. When the set of observation symbols is discrete, this density gives a true probability value; when continuous valued vectors are used, probability density functions are used in the place of the (true) probabilities, and they give the likelihood of emitting that vector. The most common probability density function that is used (in the speech recognition community) is a mixture of Gaussian densities; the mixture can have just a single Gaussian (mono or single Gaussian) or up to a maximum number of mixture components. It is accepted that the more Gaussian densities a mixture contains, the better (more detailed) the modeling is, but the issue of estimating a larger number of parameters from a limited amount data makes the quality of estimates critical. The parameters of a HMM are then the probabilities associated with each transition and the parameters of the Gaussian mixtures.

It should be stated that the true sequence of transitions is not accessible, only the observation symbols are available; in mixture models, the components are also hidden; that is why the models are called "hidden" and why the symbols are called "observations". There are actually two stochastic processes that are modeled by an HMM, one that is responsible for the change of state and another that is responsible for emitting the observations according to the density (or probability) attached to the transition that is taken.

Formally a HMM is defined by the following parameters:

- π_j: probability that the initial state is j;
- a_{ij}: transition probability to go from state i to j; and
- $b_{ij}(o_t)$: emission probability for the observation o_t when the transition arrives at state j coming from the state i.

The set of parameters of a model are collectively represented by $\lambda = (\Pi, A, B)$, where Π are the set of initial state probabilities, A is the set of transition probabilities and B represents the parameters used to calculate the emission probabilities.

To make HMMs useful for speech recognition, there are some problems [Rabiner 1993] that have to be solved:

- **Problem 1** How to calculate the probability $P(\mathbf{O}|\lambda)$ of a sequence of observations $\mathbf{O} = (o_1, o_2, \cdots, o_T)$ with length T, given the model parameters λ ?
- **Problem 2** Given the observation sequence $\mathbf{O}$, how to find the "best" (most likely) sequence of transitions $\mathbf{Q} = (q_1, q_2, \cdots, q_T)$?
- **Problem 3** How to adjust the parameters to maximize $P(\mathbf{O}|\lambda)$?

The following sections briefly review the solutions to those problems, also presented in detail in [Rabiner 1993, Jelinek 1997].

1.3.1 Problem 1 – Calculating the Probability of an Observation

This problem can be stated as follows: given a model and its parameters λ and a observation sequence $\mathbf{O} = (o_1, o_2, \cdots, o_T)$, how well does this model match the observations ? "matching" being in the sense of "most likely".

It is computationally infeasible to directly calculate the probability $P(\mathbf{O}|\lambda)$ by enumerating all possible transitions sequences of length T because the resulting number of computations is proportional to N^T, where N is the number of transitions. Fortunately, there is a procedure called *Forward-Backward* or *Baum-Welch algorithm* that enables the computation of $P(\mathbf{O}|\lambda)$ with complexity proportional to N^2T. This procedure is simply a lattice organization of the computations; as there are only N nodes in the lattice, any state sequence will re-merge onto those N nodes, despite its length.

The *Forward* procedure defines an auxiliary variable $\alpha_i(t)$ that represents the probability of the partial observation sequence from 1 up to time t and being at state i in time t, given the (current) model parameters λ:

$$\alpha_i(t) = P\left(o_1 o_2 \cdots o_t, q_t = i|\lambda\right)$$

The procedure is inductively solved following the steps:

1. Initialization
$$\alpha_j(0) = \pi_j \qquad 1 \leq j \leq N$$

2. Induction
$$\alpha_j(t) = \sum_{i=1}^{N} \alpha_i(t-1) a_{ij} b_{ij}(o_t) \qquad 1 \leq t \leq T$$

3. Termination
$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_i(T)$$

The induction step is interpreted as enumerating all possible ways to reach state j at time step t with the observation o_t in consideration. The calculation of the probability $P(\mathbf{O}|\lambda)$ is simply the sum of all possible state sequences ending at each one of the states i.

The *Backward* procedure is quite similar. The auxiliary variable $\beta_i(t)$ is the probability of the partial sequence from $t+1$ up to the end given state i at time t and the model λ:

$$\beta_i(t) = P\left(o_{t+1} o_{t+2} \cdots o_T|q_t = i, \lambda\right)$$

Solving for $\beta_i(t)$ boils down to a similar induction as used in solving the forward procedure:

1. Initialization
$$\beta_i(T) = 1 \qquad 1 \leq i \leq N$$

2. Induction
$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} b_{ij}(o_{t+1}) \beta_j(t+1) \qquad t = T-1, T-2, \cdots, 1, 0 \qquad 1 \leq i \leq N$$

3. Termination

$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \beta_i(0)\pi_i$$

The interpretation of the steps is similar the one given to the forward procedure.

1.3.2 Problem 2 – Calculating the Most Likely Path

It is not possible to give an exact solution to this problem because the true sequence is hidden; the best solution has to be found according to an optimality criterion. If the individually most likely transition at each time is chosen as criterion, the problem can be solved with the aid of the previously define forward-backward variables and the auxiliary variable $\gamma_i(t)$:

$$\gamma_i(t) = P(q_t = i|\mathbf{O}, \lambda)$$

that is the *a posteriori* probability of being at state i at time t given the observation and the model parameters. The posterior can be rewritten as

$$P(q_t = i|\mathbf{O}, \lambda) = \frac{P(q_t=i,\mathbf{O}|\lambda)}{P(\mathbf{O}|\lambda)}$$

and finally with the forward-backward variables:

$$\gamma_i(t) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{l=1}^{N} \beta_l(0)\pi_l} \tag{1.1}$$

The most likely state q_t^* at each time step is then given by:

$$q_t^* = \underset{1\leq i\leq N}{\operatorname{argmax}}\, \gamma_i(t) \qquad 1 \leq t \leq T$$

But there is a problem with this procedure; as the probability of successive states is not considered, the most likely sequence obtained through this procedure may not even be valid, i.e. when $a_{ij} = 0$. The optimality criterion has then to be changed; the most widely used solution is to maximize $P(\mathbf{Q}, \mathbf{O}|\lambda)$ where $\mathbf{Q} = (q_1, q_2, \cdots, q_T)$ is the sequence of states. We can retrieve the transitions from the state sequence, as we are assuming to have a single transition between two states (but the procedure can be augmented to track multiple transitions between the same pair of states).

Maximizing $P(\mathbf{Q}, \mathbf{O}|\lambda)$ is equivalent to finding the single best state/transition sequence,

which is easily solved with the Viterbi [G. David Forney 1973] algorithm, described next.

Viterbi algorithm

This algorithm is based on dynamic programming. The single best path (state or transition sequence) can be found with a procedure similar to the forward calculation. Let $\delta_i(t)$ be the best scoring (highest probability/likelihood) partial path from 1 up to time t and the (partial) observation given the model:

$$\delta_i(t) = \max_{q_0,q_1,\cdots,q_{t-1}} \left[P\left(q_0, q_1, \cdots, q_{t-1}, q_t = i, o_1 o_2 \cdots o_t|\lambda\right)\right]$$

It is only needed to keep track of the state i^* that maximizes:

$$\delta_j(t+1) = \max_i \left[\delta_i(t) a_{ij} b_{ij}(o_{t+1})\right]$$

for all times t and then backtrack from the end to retrieve the most likely sequence. The procedure goes as follows.

1. Initialization
$$\delta_i(0) = \pi_i \qquad 1 \leq i \leq N$$
$$\psi_i(0) = 0$$

2. Recursion
$$\delta_j(t) = \max_{1\leq i\leq N} \left[\delta_i(t-1) a_{ij} b_{ij}(o_t)\right] \qquad 1 \leq t \leq T \quad 1 \leq j \leq N$$
$$\psi_j(t) = \text{argmax}_{1\leq i\leq N} \left[\delta_i(t-1) a_{ij} b_{ij}(o_t)\right]$$

3. Termination
$$P^* = \max_{1\leq i\leq N} \left[\delta_i(T)\right]$$
$$q_T^* = \text{argmax}_{1\leq i\leq N} \left[\delta_i(T)\right]$$

4. Sequence backtracking
$$q_t^* = \psi_{q_{t+1}^*}(t+1) \qquad t = T-1, T-2, \cdots, 1$$

This procedure also presents an advantage from the computational point of view; it can be processed in the logarithm domain, avoiding the successive products of numbers that are smaller

than one. Thus underflow problems that have to be dealt with when using the forward-backward procedure are not met.

Again, the presented procedure supposes a single transition between states, but it can be easily expanded in the case of multiple transitions.

1.3.3 Problem 3 – Estimating Model Parameters

The algorithm called *Expectation-Maximization* (EM) [Dempster 1977, Hartley 1958, Bilmes 1998, Xu 1996] can be used to estimate the model parameters that maximize the probability $P(\mathbf{O}|\lambda)$. This algorithm is used in the case when the data are incomplete (when there are hidden parameters), or can be seen as incomplete; it is also used in the case where the maximization of the likelihood function (here $P(\mathbf{O}|\lambda)$) can be simplified if the values of additional parameters were known. Then from initial values for the complete data (or parameters), two steps are repeated: first the **E**xpected value for the likelihood of the complete data given the observed data and current parameter estimates is calculated and second, the expected value is **M**aximized with respect to the parameters.

This algorithm is proven to converge to a *local* maximum of the likelihood function, so it is sensible to provide adequate values for the initial parameters. It does not present numerical issues as other maximization techniques such as gradient descent. The **E** and **M** steps are repeated until the increase in likelihood between two iterations is smaller than a threshold or when a maximum number of iterations is reached. Actual training of the models needs additional setup and is described with more detail in Section 1.5; the rest of the section briefly reviews the theoretical aspects of estimating parameters for HMMs.

In the case of HMMs, the state sequence completes the data; the details of the procedure are well known [Dempster 1977, Hartley 1958, Bilmes 1998, Xu 1996] and for brevity will not be repeated here, only the solutions are presented. The initial state probabilities are given by (with the aid of Equation 1.1):

$$\hat{\pi}_i = \gamma_i(0) \qquad 1 \leq i \leq N$$

and the state transition probabilities by:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \xi_{ij}(t)}{\sum_{t=1}^{T} \gamma_i(t)}$$

where the auxiliary variable $\xi_{ij}(t)$ is given by:

$$\xi_{ij}(t) = P(q_{t-1} = i, q_t = j|\mathbf{O}, \lambda) = \frac{P(q_{t-1} = i, q_t = j, \mathbf{O}|\lambda)}{P(\mathbf{O}|\lambda)}$$

which can be calculated with the forward-backward variables:

$$\xi_{ij}(t) = \frac{\alpha_i(t-1)a_{ij}b_{ij}(o_t)\beta_j(t)}{\sum_{i=1}^{N}\alpha_i(t)\beta_i(t)}$$

So far, the form of the probability density attached to the transitions $b_{ij}(o)$ was not specified; it is now needed for the EM algorithm. The emission probability densities are modeled with Gaussian mixtures (with M Gaussian densities $\mathcal{N}$) as given by:

$$\begin{aligned} b_{ij}(o_t) &= \sum_{m=1}^{M} w_{ijm} \cdot \mathcal{N}\left(o_t, \mu_{ijm}, \Sigma_{ijm}\right) \\ &= \sum_{m=1}^{M} w_{ijm} \cdot \mathcal{N}_{ijm}(o_t) \end{aligned}$$

The parameters of a mixture are the M mixture weights $w_{\cdot m}$, and M Gaussian mean vectors and covariance matrices ($\mu_{\cdot m}$ and $\Sigma_{\cdot m}$, respectively). It is usual to consider the covariance matrices to be diagonal; that is because it is assumed that the components of the feature vector are independent and also to reduce the number of parameters. Moreover, it can be proved that a mixture of diagonal covariance Gaussian densities can arbitrarily approximate any probability density function, given a number of Gaussian components that is large enough. Let us first consider a single Gaussian density:

$$\mathcal{N}\left(x, \mu, \Sigma\right) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}} \exp\left(-\frac{1}{2}\left(x-\mu\right)^t \Sigma^{-1}\left(x-\mu\right)\right)$$

The EM algorithm maximizes the log-likelihood (it is simpler in the log-domain); the log-likelihood of a Gaussian density is:

$$\log\left(\mathcal{N}\left(x, \mu, \Sigma\right)\right) = -\frac{D}{2}\log 2\pi - \frac{1}{2}\log|\Sigma| - \frac{1}{2}\left(x-\mu\right)^t \Sigma^{-1}\left(x-\mu\right)$$

In the case of diagonal covariance matrix with diagonal elements σ_d^2, the likelihood simplifies to:

$$\log\left(\mathcal{N}\left(x,\mu,\Sigma\right)\right) = -\frac{D}{2}\log 2\pi - \frac{1}{2}\sum_{d=1}^{D}\log\sigma_d^2 - \frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_d-\mu_d}{\sigma_d}\right)^2$$

The mean vector element μ_{ijd} for dimension d of the transition between states i and j is re-estimated as:

$$\hat{\mu}_{ijd} = \frac{\sum_{t=1}^{T}\xi_{ij}(t)\cdot o_{td}}{\sum_{t=1}^{T}\xi_{ij}(t)}$$

and the diagonal elements (variances) σ_{ijd} using:

$$\hat{\sigma}_{ijd}^2 = \frac{\sum_{t=1}^{T}\xi_{ij}(t)\left(o_{td}-\hat{\mu}_{ijd}\right)^2}{\sum_{t=1}^{T}\xi_{ij}(t)}$$

Now, when mixtures of Gaussian are used, the variable $\xi_{ijl}(t)$ is defined to take into account which component (l) is selected at time t; note that this information is also part of the incomplete data used in the EM algorithm:

$$\xi_{ijl}(t) = P(q_{t-1}=i, q_t=j, C_{ijt}^*=l|\mathbf{O},\lambda) = \xi_{ij}(t)\frac{w_{ijl}\cdot\mathcal{N}_{ijl}(o_t)}{\sum_{k=1}^{L}w_{ijk}\cdot\mathcal{N}_{ijk}(o_t)}$$

Where C_{ijt}^* (see equation 1.3.3, Section 1.3.3) indicates which mixture component is selected at the transition at time t. Now the elements of the mean vector for the l-th Gaussian are given by:

$$\hat{\mu}_{ijld} = \frac{\sum_{t=1}^{T}\xi_{ijl}(t)\cdot o_{td}}{\sum_{t=1}^{T}\xi_{ijl}(t)}$$

And the variances:

$$\hat{\sigma}_{ijld}^2 = \frac{\sum_{t=1}^{T}\xi_{ijl}(t)\left(o_{td}-\hat{\mu}_{ijld}\right)^2}{\sum_{t=1}^{T}\xi_{ijl}(t)}$$

New values for the Gaussian weights w_{ijl} are:

$$\hat{w}_{ijl} = \frac{\sum_{t=1}^{T}\xi_{ijl}(t)}{\sum_{t=1}^{T}\xi_{ij}(t)}$$

We notice that the constraints are met for the transition probability estimates as well as for the mixture weights. The procedure described so far considers estimating the parameters for one model and its corresponding observation sequence. To estimate parameters for a composite model, i.e. for the multiple phone models corresponding to the pronunciation of a given word, a "super" HMM is constructed by concatenating phone models and the same procedure is used.

When context-dependent models are employed, the appropriate contextual states have to be used when connecting the phone models to correctly assign data to those states; this will be explained further in Section 1.5.

Multiple observation sequences

The re-estimation formulae given in the previous section are valid for a single observation sequence $\mathbf{O}$; when there are multiple independent observation sequences $\mathbf{O}_1, \mathbf{O}_2, \ldots, \mathbf{O}_K$, with lengths $T_1, T_2, \ldots, T_K$, the new parameters are estimated simply by the weighted average over all sequences:

$$\hat{\pi}_i = \frac{\sum_{k=1}^{K} \gamma_i^k(0)}{K} \tag{1.2}$$

$$\hat{a}_{ij} = \frac{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ij}^k(t)}{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \gamma_i^k(t)} \tag{1.3}$$

$$\hat{w}_{ijl} = \frac{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ijl}^k(t)}{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ij}^k(t)} \tag{1.4}$$

$$\hat{\mu}_{ijld} = \frac{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ijl}^k(t) \cdot o_{td}^k}{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ijl}^k(t)} \tag{1.5}$$

$$\hat{\sigma}_{jld}^2 = \frac{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ijl}^k(t) \left(o_{td}^k - \hat{\mu}_{ijld}^k\right)^2}{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \xi_{ijl}^k(t)} \tag{1.6}$$

Viterbi training

It is also possible to estimate the best transition (and Gaussian component) sequence for each observation with the Viterbi algorithm and use those values for estimating the new parameter values. It is equivalent to the formal EM algorithm, where the **E** step is replaced by a Viterbi decoding. The values for the best transition sequence and Gaussian component can be seen as an *alignment* of the training data with the models. This procedure is also guaranteed to converge [Picone 1990] and the same formulae presented in Section 1.3.3 are used; the difference is in the definitions of $\gamma_j(t)$ and $\xi_{ijl}(t)$ (the superscript k is dropped for convenience) as given by the following equations.

$$\gamma_j(t) = \begin{cases} 1 & \text{if the state at time } t \text{ is } q_t^* = j \\ 0 & \text{otherwise} \end{cases}$$

$$\xi_{ijl}(t) = \begin{cases} 1 & \text{if the mixture component indicator is } C^*_{ijt} = l \\ 0 & \text{otherwise} \end{cases}$$

Due to the exponential decay of the likelihood for the different mixture components, the emission probability can be approximated by the single best Gaussian (best *score*); this is the same as if the information about which component is used was available:

$$b_{ij}(o_t) \approx \max_m \left[w_{ijm} \cdot \mathcal{N}_{ijm}(o_t) \right]$$

with the best component indicated by:

$$C^*_{ijt} = \operatorname*{argmax}_m \left[w_{ijm} \cdot \mathcal{N}_{ijm}(o_t) \right]$$

This type of training is much more computationally efficient than the formal forward-backward variables and does not suffer from numerical problems, as the Viterbi search can be done entirely in the logarithmic domain. A variant that uses K-means [Duda 2000] for Gaussian mixture estimation is given in [Rabiner 1986]. The values $\gamma_j(t)$ and $\xi_{ijl}(t)$ are now simple indicators; their sum (*counters*) represents the number of times state j was visited and the number of times the transition between states i and j was taken with the l-th Gaussian component presenting the best score, respectively. The latter value is also know as the *mass* of the Gaussian and it is the number of feature vectors that are "aligned" with that Gaussian.

In this work a further simplification of the segmental K-means training is used. Instead of re-calculating the best state/transition sequence at each iteration, it is calculated once and it is kept fixed for the rest of the training procedure. To save computations, mono Gaussian densities are used to calculate the alignments. When Gaussian mixtures are trained, it is only necessary to calculate the component indicators C^*_{ijt} (along with the estimates for means and variances). A comparison between training schemes is presented in Section 1.5 where it is shown that fixing the alignments issued from single Gaussian models results in no degradation or just a slight decrease in performance, but the training time is reduced significantly. The next section describes the speech corpora that is used to train and evaluate the models.

1.4 Speech Corpora

In the development of an ASR system, two speech databases (corpora) are needed; one is used to estimate parameters (training corpus) and the other is used to assess performance (evaluation corpus). A third corpus is sometimes used to verify the stopping criterion for the training iterations (cross-validation), but it is completely optional for the approach taken here. It should be noted that for performance evaluation in speaker-independent speech recognition, speakers that are present in the training corpus should not appear in the evaluation corpus to avoid biased results. It is desirable to have as many different speakers as possible to provide data representing the differences between speakers.

The vocabulary content (phonetic coverage) plays an important role in the design of the training corpus. This corpus should contain as many samples as possible for the most part of speech sounds (phones) and also cover the contextual variations of the phonemes to provide a modeling that is detailed and complete. Note that when there is an overlap between the vocabularies of training and evaluation corpora, there is a tendency of the recognition rate to be higher for this particular corpus, because the same "kind" of speech is properly modeled (see Figure 1.8 for an illustrative example of training and testing of the same vocabulary and training on a "generic" vocabulary but testing on another one).

It is clear that a training corpus should be very large to represent the various speakers and also to cover the contextual variation of speech. Nowadays there are some huge training corpora, like the one used in Broadcast news transcription which can amount to some thousands of hours of speech data! There is a well known saying which goes as:

> "There is no data like more data"

The drawbacks of huge speech corpora are the high cost to collect the data and specially to annotate it; the minimum annotation is the transcription into words, but it could go into detailed phonetic transcription with annotation of some characteristics of interest like extra events (background speech, different types of noise such as radio, tv, door slam, etc.) But not everybody follows the "more data" credo; there is some recent work on *data selection* [Hasenjäger 2000], where it is shown that the same level of performance can be attained with a relatively small fraction of the training data (around 20-30%). It seems that the credo should be that "there is no data like *better* data". The idea is to select relevant data, which cannot be explained by the current models, and discard already well modeled data. Even if this

approach is not explicitly followed here, it seems worth to mention it. In the part dealing with multi-modeling (Chapter 5) we train models on specific chunks of the training data trying to specialize the models to some types of variation.

We use four speech databases to train or adapt models. Three of them were collected from the Public Subscriber Telephone Network (PSTN) and the other from the Global System for Mobile Communications (GSM) network; data was sampled at 8 kHz in all cases and the language is French (from France). All databases are proprietary material of France Télécom R&D.

1.4.1 Training

The first PSTN training corpus is called `Allophones`; it covers most of the contexts in French and it is composed of a mix of short and long phrases; there are about 13 hours of speech (plus silence) and 1700 speakers. This database is used in some preliminary experiments and also to calculate initial values for model parameters.

The second PSTN corpus is an extension of `Allophones` and is called `PSTN`; it is complemented by isolated words and short phrases (digits, some city names, pairs of numbers from 00 to 99, spelled letters and commands). This corpus has about 300 hours of signal (speech and silence) and there are about 61600 speaker sessions; a speaker session is a recording session of the same speaker.

The last training database for PSTN channel consists only of telephone numbers and it is used to validate fixed alignment (cf. Section 1.5) training; this database is called `TelNum_TRN`.

The GSM training corpus comprises a mix of short and long phrases and is similar to `PSTN`; it contains digits, pairs of numbers, and commands but there are no city names. This database is called `GSM`; it amounts to 100 hours of signal and there are 19100 speaker sessions. Table 1.1 summarizes the sizes of the training corpora.

Corpus	Utterances	Speaker sessions
`Allophones`	47571	1700
`PSTN`	906863	61600
`TelNum_TRN`	39425	3599
`GSM`	249945	19100

Table 1.1: Total number of utterances and speaker sessions for the training corpora.

1.4.2 Evaluation

The models are evaluated in three tasks that are common to both PSTN and GSM data; the same tag is used to identify the task and the corresponding test set:

- `Digits` : isolated digits from 0 to 9; this is a quite easy task and is included to be sure that there are no obvious modeling problems;
- `Numbers` : 2-digits numbers from 00 to 99 (denoted by "XX"); the numbers are modeled as compound words and pronunciation alternatives are included;
- `TelNum` : telephone numbers spoken in the usual French way: either as 5 instances of 2-digits numbers "XX XX XX XX XX" or as "0 80Y XX XX XX", where "Y" is one of 0, 1, 2 or 3; note that the speakers are not the same as those in the `TelNum_TRN` corpus.

The fourth task is available only for PSTN data and it was collected from an actual ASR system in field experiments; the vocabulary consists of the variations of all city (*communes*) names in France. This task, called `Cities` is the most difficult due to the large size of the vocabulary; there are about 45000 compound words (including denomination variants) in the vocabulary with many homophone words (words with the same phonetic pronunciation) like "*Nantes*" and "*Nante*"; there are also many word with similar pronunciation like "*Rennes*", "*Arrènnes*", "*Airaines*", and "*Parennes*". Note that city names like "*Saint Laurent les Bains"* are modeled as single (compound) word; in their pronunciation there is an optional pause between words. Table 1.2 presents the number of word occurences in each task and channel.

Task	PSTN	GSM
`Digits`	4375	5597
`Numbers`	7244	7005
`TelNum`	28320	30380
`Cities`	6584	n/a

Table 1.2: Total number of word occurrences in each task and channel.

The figure of merit used to evaluate the models is the word error rate (WER). In continuous speech tasks such as `TelNum` there are three types of errors: substitutions, insertions and deletions. These errors are counted by aligning the recognized word sequence (hypothesis) with the reference (correct) sequence [Hunt 1988]; a substitution occurs when one word is mistakenly recognized as another word, a deletion is the removal of one of the reference words and an insertion is when a extra word is added to the hypothesis. The alignment is made with dynamic

programming and different weights are assigned to substitution and deletion/insertion. The word error rate (WER) is then given by

$$\text{WER} = \frac{S + I + D}{N} 100\%$$

Where N is the number of words and S, I, and D the number of substitutions, insertions, and deletions, respectively.

The next section presents the model training path that is followed in this work.

1.5 Training Path

As shown in Section 1.3.3 the parameter estimation procedure begins with assigning initial values to the parameters and then doing some iterations on the training data. It remains an open issue how to derive initial values for the parameters. One possibility is to start with context-independent phone models, thus there are very few parameters, and use isolated words data. The data is roughly segmented with a speech detection algorithm [Stadermann 2001] and then the feature vectors in each segment are linearly assigned (flat start) to each transition (of the corresponding model) and parameters for single Gaussian densities are estimated. Then some iterations are performed (using either EM or Viterbi-training); in these iterations the segmentation of the data is said to be *free* as the boundaries between speech units (CI phones) and silence may change from one iteration to the other. After these iterations on isolated speech, other utterances with continuous speech are added to the training database for further tuning the models. Then the parameters of CI models are used to initialize parameters for CD models and some iterations are performed on the full training corpus. Estimation with Gaussian mixtures is described in Section 1.5.1.

Viterbi-training is used in this work in all estimation iterations. The "Decoder" receives a training utterance (series of feature vectors) and its word transcription and only the paths in the compiled FSN/transducer corresponding to the same word sequence are explored. Then the "Decoder" outputs the alignment of the feature vectors with each transition, i.e. to each vector is assigned a transition identity (or the identity of the probability density function (pdf) attached to it, this will be clearer when parameter-tying is discussed). Then the parameters are estimated from all feature vector/pdf alignments using the formulae given in Section 1.3.3. Using the new estimates for the parameters, the whole decode-estimate process is repeated and it stops when

the likelihood increase between iterations is below a threshold.

1.5.1 Gaussian Mixture Training

After the training of single Gaussian models has converged, initial parameters for Gaussian mixtures are obtained simply by *splitting* [Sankar 1998] the mono Gaussian densities. A Gaussian mixture is "split" by creating two new Gaussian components for each original component; the mean vectors for the two new components are given by adding (subtracting) a random perturbation vector to the original mean vector, the variances are kept to be the same and the mixture weight is equally divided between the new components. After splitting all suitable Gaussian components, the same decode-estimate procedure is repeated until convergence, as presented in Section 1.3.3.

The term "suitable" Gaussian was employed because it may happen that not every Gaussian component is worth to be split. A criterion can be applied to select which components are going to be split, thus reducing the overall number of parameters and improving modeling. The most common criterion is the "mass" (cf. Section 1.3.3) of the Gaussian; if it is below a minimum (ranging from 3-50 vectors) the Gaussian is not split. This criterion is used to improve the confidence on the variance estimation, but it is not unusual that some systems set a limit as low as 3, because there is another mechanism that limits the minimum value of the variances. This mechanism prevents the occurrence of Gaussian densities that are too "sharp" or "peaky".

Other criteria are the well known: Bayesian Information Criterion (BIC) [Schwarz 1978] and Minimum Description Length (MDL) [Rissanen 1978, Hansen 2001]; both are formally equivalent in the case of Gaussian densities. Those criteria aim at making a compromise between the amount of training data (represented by the Gaussian's mass) and the complexity of the model (number of parameters). Those criteria are individually applied to each component to validate (or to reject) the split; the drawback (from the point of view of computational cost) is that the parameters have to be re-estimated for the application of those criteria. A split is validated if the difference between the value of the criterion (BIC or MDL) after the split and before is positive (or larger than a threshold). The MDL is given by:

$$\mathrm{MDL}(\mathbf{X}, \Theta) = -\mathcal{L}(\mathbf{X}, \Theta) - \frac{\mathrm{N}(\Theta)\log(T)}{2}$$

Where $\mathbf{X}$ are the T data points (feature vectors), Θ are the parameters of the mixtures, $\mathcal{L}(\mathbf{X}, \Theta)$

is the log-likelihood of the data and the model and $N(\Theta)$ is a function which gives the number of parameters Θ.

There is a variation of the BIC that is called *penalized* BIC, where a penalty factor η is included in the second term of the rhs $\eta\frac{N(\Theta)log(T)}{2}$ of the previous equation; this penalty factor is empirically tuned (although it could be formally done) to optimize the model selection.

Some informal experiments were performed with a simple approximation: the likelihood increase after the split should be higher than a threshold to validate the split. When mixtures up to 16 Gaussian components are trained, approximately 20% of the new Gaussian densities could be ignored without performance degradation. Although interesting from the point of view of reduction of the number of parameters, split validation is not further pursued in this work. Instead, an alternative Gaussian mixture initialization scheme [Messina 2004b] that avoids the repeating split/estimate iterations is described in the next section.

1.5.2 Gaussian Mixture Initialization

It is assumed that the training data are already aligned to some models [1]. It is sufficient to align the data to single Gaussian models; single Gaussian models do not deliver optimal performance in recognition but with forced-alignment they provide good enough segmentation of the data which allows to derive proper initial models as will be shown by the results.

The idea driving the algorithm is to "increase" a mixture (i.e. add a Gaussian component) when the distance between a feature vector and the mixture is above a given threshold; the distance measure that is used here is simply the minimum Euclidean distance between the feature vector and the mean vectors of all Gaussian components belonging to the mixture. The variances are not used at all because during mixture growing, their values may not be properly estimated. Threshold values can be set for the maximum number of Gaussian components in a mixture and the "membership" distance, which controls how the mixtures are increased. Proceeding in this way avoids the split/train iterations and also avoids reading large amounts of data in memory.

Each "proto-mixture" (one for each transition or density in the model) stores for each possible Gaussian component: a mean vector, an accumulator (for the variances), and the number of vectors k that were aligned (so far) with the mixture. When a new component is added, its

[1] The target models' topology could be different, but the labels identifying the densities have to be transformed accordingly.

mean vector is set to be equal to the current feature vector and the variance accumulator is set to zero. For each mixture it is given the maximum number of elements $maxElem$ and the maximum membership distance $dMax$ to "accept" the vector as belonging to the mixture in its current state. Let $d_j^t = \text{dist}(x, \mu_j^t)$ denote the distance to the j-th component of the t-th mixture, and let i denote the nearest element and numElem(t) the current (actual) number of components in the mixture.

To avoid numerical problems, the mean and the variance accumulators are calculated as described in [Knuth 1997]. After reading t vectors (let x denote the current (t-th) observation vector and let μ_i^t denote the mean vector of component i when t vectors are read) for a given mixture, the mean vector (for all dimensions d) of component i is update by equation:

$$\mu_{i_d}^t = \mu_{i_d}^{t-1} + \frac{x_d - \mu_{i_d}^{t-1}}{t}$$

And the value of the variance accumulator increases as in:

$$S_{i_d}^t = S_{i_d}^{t-1} + \left(x_d - \mu_{i_d}^t\right) \cdot \left(x_d - \mu_{i_d}^{t-1}\right)$$

The variance is given by $\sigma_{i\,d}^{2\,t} = \frac{S_{i_d}^t}{t-1}$.

When there are no components with distance smaller than $dMax$, a new component is added to the mixture; if the maximum number of components is reached, the nearest component is updated. The algorithm is summarized below; it was later found that this clustering algorithm is known as *Leader-follower* [Duda 2000], but in this work it is called Sequential Clustering Algorithm (SCA), because data are processed and clustered sequentially.

---

for all pairs vector/density: $(x; t)$ **do**
  $d_j^t = \text{dist}(x, \mu_j^t)$
  $i = \text{argmin}_j\, d_j^t$
  if $d_i^t < dMax$ **then**
    update element i
  else
    if numElem(t) $< maxElem$ **then**
      increase mixture
    else
      update element i
    end if
  end if
end for

SCA has lower complexity than standard K-means because each data point in training is only visited once, and not as many as K times as in K-means. It is very costly to run a K-means when there is a large number of vectors associated with a mixture; the usual solution is to employ a subset of the training data for K-means initialization of Gaussian mixtures, but this may leave out some regions of the feature space, slowing down the convergence of training and maybe reducing the modeling power of the mixture with regard to the whole data set. Moreover, with the SCA all training data available for each mixture is used thus the feature space as spanned by the training data is visited, avoiding regions without any mean vector in their neighborhood.

After processing all training data (`PSTN` and `GSM` corpus), the components that "saw" (were aligned) less than a minimum number of vectors $minM$ could be removed, or the nearest element can be found, still in the sense of Euclidean distance between the means, and these elements are merged. The latter approach was chosen and those components are merged until complete removal of low-mass components. For densities that are associated with a number of vectors that is less than $minM$, the maximum number of elements is set to one beforehand; this avoids growing a mixture and then merging the components back into a single one. Initial values for the variances are calculated with the accumulators and values for the mixture weights are calculated with the number of vectors associated with each component.

Then some training iterations are performed to refine the estimations of the parameters. In figures 1.5 and 1.6 the WER for all PSTN tasks are illustrated for the initial models (just after clustering) and after 7 Viterbi iterations for some values of $dMax$ (8 and 16 Gaussian components, respectively, with $minM = 20$); in Figure 1.7 we present the results for GSM data and with 8 and 16 Gaussian components models. The results are comparable to those obtained with usual split/train, represented by the continuous line (the dotted lines are the 95% confidence interval), but with much less iterations and a shorter training time. It is also worth to note that there is no correlation (not everywhere) between the performance of initial models (marked with upside down triangles) and of the models after training (marked with stars); this may be due the fact that the initial models are near a local maximum and that the subsequent iterations just move the parameters closer to that maximum.

The performance of the trained models is not strongly dependent on the value of $dMax$ (below 17), which is a good property of this algorithm; the "optimal" value of $dMax$ lies between 5 and 10 and can be experimentally found.

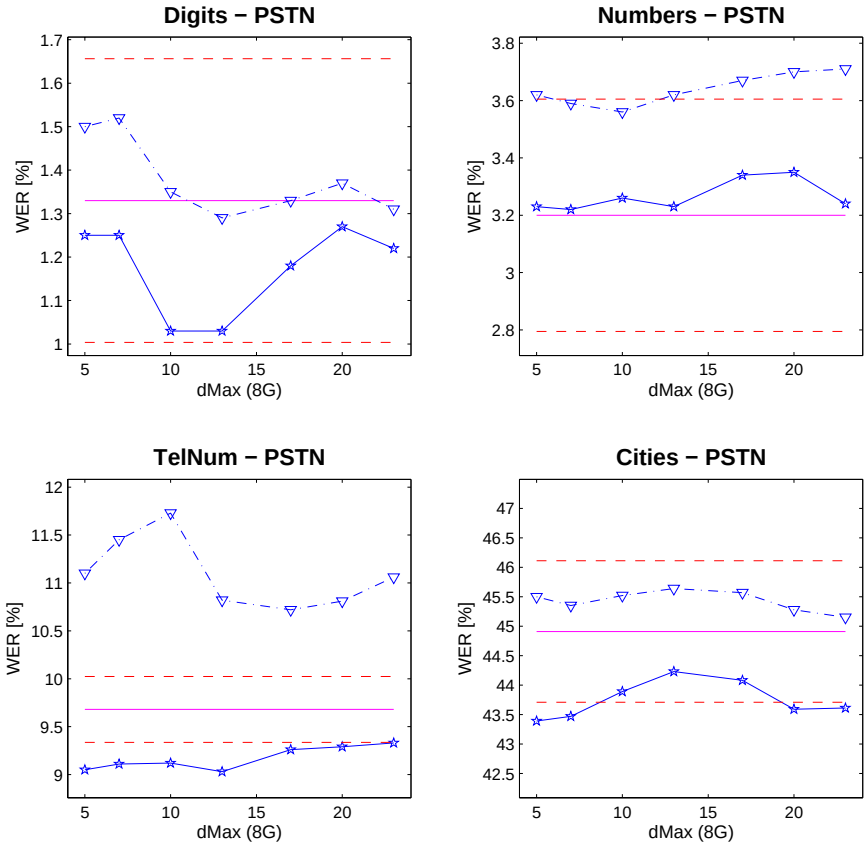


Figure 1.5: WER for Gaussian mixture models (8 components) for split/train and SCA models on PSTN data and some values of dMax.

1.5.3 Training with Fixed Alignments

As mentioned before, a further simplification of Viterbi training is adopted in this work; instead of re-aligning the training data with the "Decoder" after each parameter estimation iteration, the same alignment (obtained with a single Gaussian) is used throughout all Gaussian mixture training iterations; this scheme is called *Fixed alignment* training. Another variation is to re-align the data from time to time, e.g. at the end of the parameter re-estimation iterations, i.e. before every increase in the number of components (or split).

To assess the losses in performance, some preliminary experiments are performed comparing fixed alignment training against standard Viterbi training; the training and evaluation databases have only telephone numbers (`TelNum_TRN` and `TelNum`) and the models are CD phone models. In Figure 1.8-a the WER for the different training schemes is shown as a function of the number of Gaussian components. This figure illustrates the WER for Viterbi, fixed alignment from mono Gaussian models, and training with re-alignment before each split (denoted by "Fix_Re-align" in the figure). The different WER for the single Gaussian models is due to the extra iteration

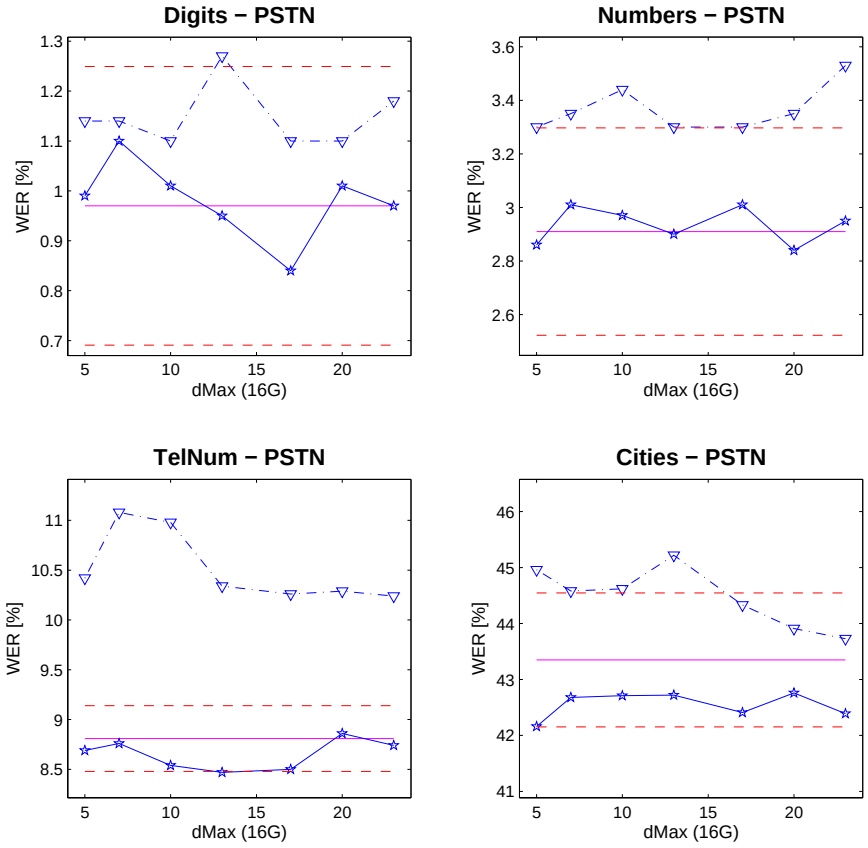


Figure 1.6: WER for Gaussian mixture models (16 components) for split/train and SCA models on PSTN data and some values of dMax.

that was performed to obtain the alignment, this is avoided in the other training experiments. There is a slight degradation but it is not significant.

Figure 1.8-b illustrates the evaluation on telephone numbers, but training is performed on the (larger) `PSTN` training corpus; note that this corpus does not have any telephone number samples; this explains the lower performance compared to the previously presented results. The recognition results with the Viterbi scheme are with a set of existing models that were trained on a different training corpus so the results are not directly comparable, but they give a good idea of the expected performance, This figure does not present the "Fix_Re-align" scheme because it is not used in this experiment.

Again, the differences are not significant and the iteration time, i.e. the time needed for one iteration, is reduced by a factor of *forty* (40) on average (it decreases as the number of Gaussian components increase, as the only computation that is needed is the identification of the best scoring component). The total training time can be further reduced by training the mono Gaussian models on a subset of the training data, then performing a couple of iterations

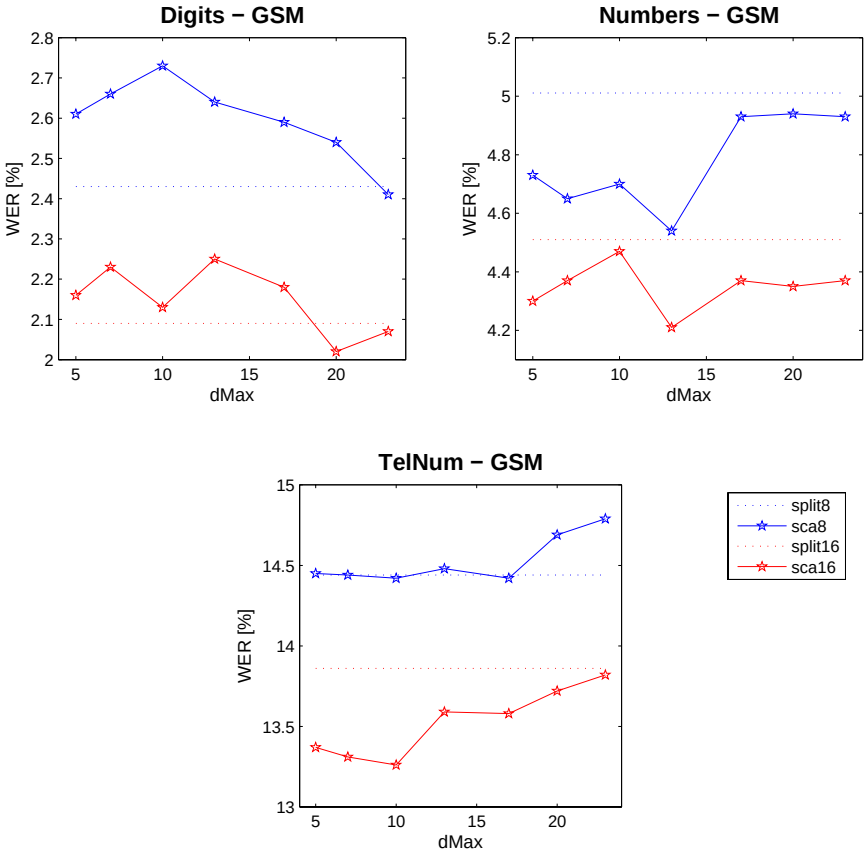


Figure 1.7: WER for Gaussian mixture models (16 components) for split/train and SCA models, GSM data and some values of dMax.

on all data and then store the alignments. The iterations on all data are optional and acceptable performance is obtained without the latter iterations.

It is interesting to note that training and testing on the same type of data results in very high performance; single Gaussian models trained on the `TelNum_TRN` corpus performs as well as models with 8 Gaussian components trained on the much larger `PSTN` database.

1.5.4 Model Adaptation

Adapting models is quite useful in speech recognition; it allows to recover from differences between training and test conditions and also to "tune" the recognition system to a particular speaker's voice. It is also possible to adapt the models either to a new vocabulary or to a different acoustical environment.

We use the well-known Maximum Likelihood Linear Regression (MLLR) method [Gales 1996, Delphin-Poulat 2001]. The idea behind MLLR adaptation is to find a linear transformation of the parameters that maximizes the likelihood to the adaptation data. The parameters (Gaussian

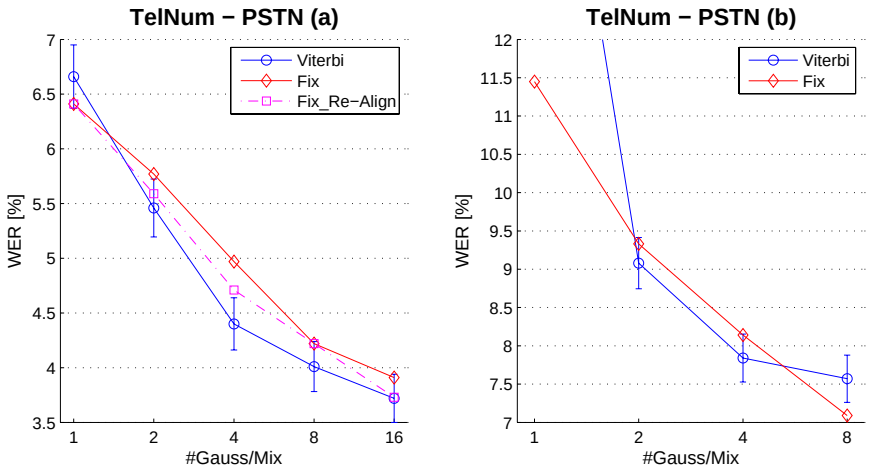


Figure 1.8: Comparison between training schemes (Viterbi, Fixed alignment, and Re-align before split).

means and variances) are adapted as:

$$\begin{aligned} \hat{\mu} &= \mathbf{A}\mu + b \\ \hat{\Sigma} &= \mathbf{A}\Sigma\mathbf{A}^t \end{aligned}$$

Where μ, Σ are the original means and variances and the same symbols with a hat are the adapted parameters. The transformation parameters are a matrix $\mathbf{A}$ and a bias vector b; the transformation matrix $\mathbf{A}$ can be either full or diagonal (to reduce the number of parameters).

In class-based adaptation, regression classes are defined (phonetically or acoustically) and all Gaussian components belonging to the same class are adapted with the same transformation. The classes define Gaussian components that are similar (according to the chosen criterion) and thus will be adapted in the same fashion. This classification allows the adaptation of all Gaussian elements with limited training data. The re-estimation equations are derived and explained in detail in [Gales 1996].

First all the Gaussian functions in the models are grouped (bottom-up) in a tree structure; the nearest pair of Gaussian densities are merged into a node (to each node is associated a Gaussian) and the process is repeated until a single Gaussian node remains. Then, some iterations on the adaptation data are performed and the total mass associated to each node (node plus its offsprings, down to the leaves) is checked against a minimum threshold rlm and if the mass is larger than rlm, a regression class is associated to that node and the parameters of the transformation are calculated. All Gaussian functions (means and variances) of each class are

then adapted with the class-specific transformation; we use a diagonal transformation matrix. The mixture weights are not adapted as this adaptation does not improve performance (based on in-house experiments). The next section presents context-dependent modeling.

1.6 Context-dependent Units

Most of today speech recognition systems employ phoneme-based modeling units. Phonemes are attractive as units because they exist in a small number (36 for French, 44 for English), thus it is relatively easy to estimate robust models. There is a large body of phonetic knowledge that can be used to obtain phonetic pronunciations for any word in a language, making the vocabulary quite extensible. Due to limitations that are described next, the basic units which are *de facto* employed are context-dependent (CD) models of phones (allophones).

One problem with phonemes is that they are an artificial representation of speech; people when asked to repeat a word that was not understood will most likely syllabify it or spell it letter by letter, and not use any phonemes at all. Phonemes are good for theoretical studies but are not a good representation for speech communication.

Another problem is the variability caused by *co-articulation*. When people speak, tongue, lips, and jaw are moved rapidly: so the motions needed for adjacent vowels and consonants have to be produced simultaneously to generate smooth speech. Co-articulation is the name of the mechanism used by the brain to coordinate those motions. Moreover, acoustic information about a vowel or consonant is spread out by co-articulation, helping a listener understand what is said. Consequently, phonemes (in the canonical sense) do not really exist in speech, but the variations of each phoneme (called allophones) do exist: they are the result of the interaction between the phoneme with the surrounding phonemes, or its *context*. Phonemes affect each other at different levels; one phoneme can be not at all affected by a preceding (or succeeding) phoneme or it can be heavily transformed by the context. Contextual variations can affect not only the immediate neighboring phoneme, but can affect many others (e.g. in the case of lip rounding). Some systems use a large contextual window (with length up to 5 preceding and 5 succeeding phonemes) to take into account the most of contextual variation [Ljolje 2000, Finke 1997].

The drawback of CD phoneme-like units is that the number of possible units increases exponentially with the length of the context window; even if this number is limited by the phonotactic constraints of the language, it remains quite a large number. A large number of units results

into a large number of parameters to estimate; reliable estimation of such a huge number of parameters as encountered in speech recognition systems is very difficult. This problem is difficult because an excessively large amount of training data would be needed to provide examples of each possible context. As the number of units increases, the number of samples available in the training data for each unit decreases, making the estimation less reliable and less robust. Due to the large number of units, it is common to share data between transitions (or states) that are similar; this technique is a particular case of the more general parameter tying [Young 1992, Takahashi 1995]. Parameter tying does not affect the re-estimation formulae, it just changes how the data are gathered for parameter estimation. The most common form of tying is via a decision trees; the tree is build by asking (phonetically meaningful) questions about the neighboring phonemes. Starting from the root node where all data are gathered together, the question that most reduce the entropy is selected and the data are split into two child nodes; the process is repeated until a stopping criterion is satisfied. Details on this technique can be found in [Bahl 1991, Nock 1997, Young 1994]. Tree-based state tying requires models for all units; so there is a preliminary training step to obtain those (untied) models and then build up the tree.

In this work, instead, states/transitions are tied beforehand; an expert phonetician has decided which contextual states are similar and thus can be treated as a single state, as shown in [Bartkova 1991, Jouvet 1994]. The topology illustrated in Figure 1.9 is used for all context-dependent units; just the number of entry (exit) contextual states has to be adjusted to match the number of left (right) contexts defined for each phoneme. The transition "fcx" provides a path through the model that can be traversed with just 3 observations, giving the minimum duration of a phone. There is a list of left (right) contexts that are available and each contextual state is identified to one of the contexts, so the proper states can be connected with the corresponding transitions when building up composite models for words and phrases. This structure specifies the tying of the parameters. All paths from an entry to an exit state corresponds to a context-dependent model of the phoneme.

The basic units that are modeled in this work are context-dependent phones (allophones) which depend only on the immediate preceding and following phonemes. These units are similar to the usual *triphones* [Schwartz 1985]. The parameters of the contextual states are bootstrapped from context-independent (single Gaussian) models which have just a single entry (exit) "contextual" state, then a small and random perturbation is added to the Gaussian mean vectors of

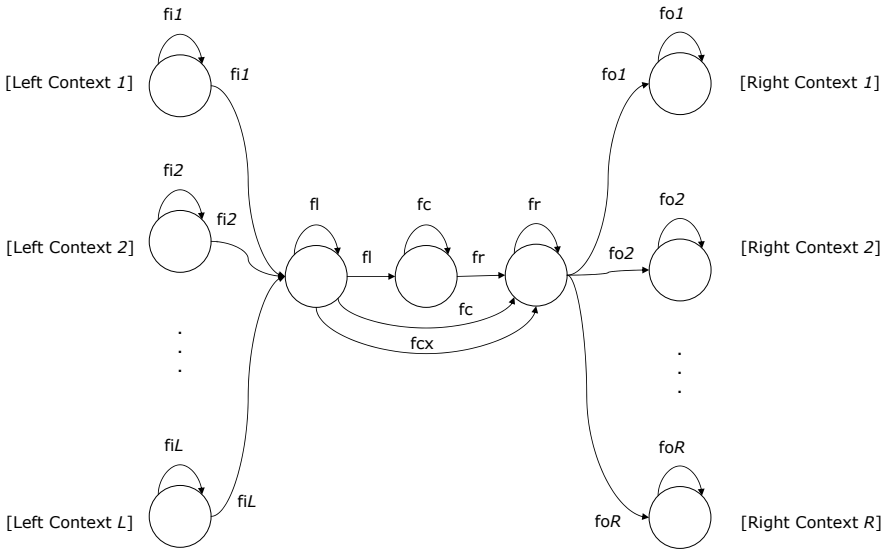


Figure 1.9: Topology used for context-dependent phones.

the context-dependent states and the parameters are re-estimated.

1.7 Baseline Models and Results

The reference results that are going to be used to assess performance are obtained with models trained respectively on the `PSTN` and `GSM` corpora with the usual split/train procedure; models have up to 16 Gaussian components per mixture.

When multi-modeling is employed, a different baseline has to be adopted. The training corpus of both channels are combined and then used to train models up to 32 Gaussian components; this is the usual way to deal with the variations and is called *mixed training*.

The results on all tasks and channels are presented in figures 1.10 and 1.11 for channel specific ("base") models and mixed training ("mix") on PSTN and GSM data respectively. All results are for allophone models as described in Section 1.6.

Regarding the base models, it can be seen that the errors on GSM data are higher than those on PSTN data; this is expected as the GSM data is noisier (on average) than PSTN data (the SNR in fixed telephone lines should be at least 25dB according to the recommendation) and also because speech is encoded and decoded, thus being modified by the transmission process. The WER for `Digits` and `Numbers` are already quite low and it is expected that improvements will be better observed for the `TelNum` and `Cities` tasks that presents the highest error rates. With a N-best decoding [Schwartz 1990, Jiménez 1995] (the system outputs the N top scoring

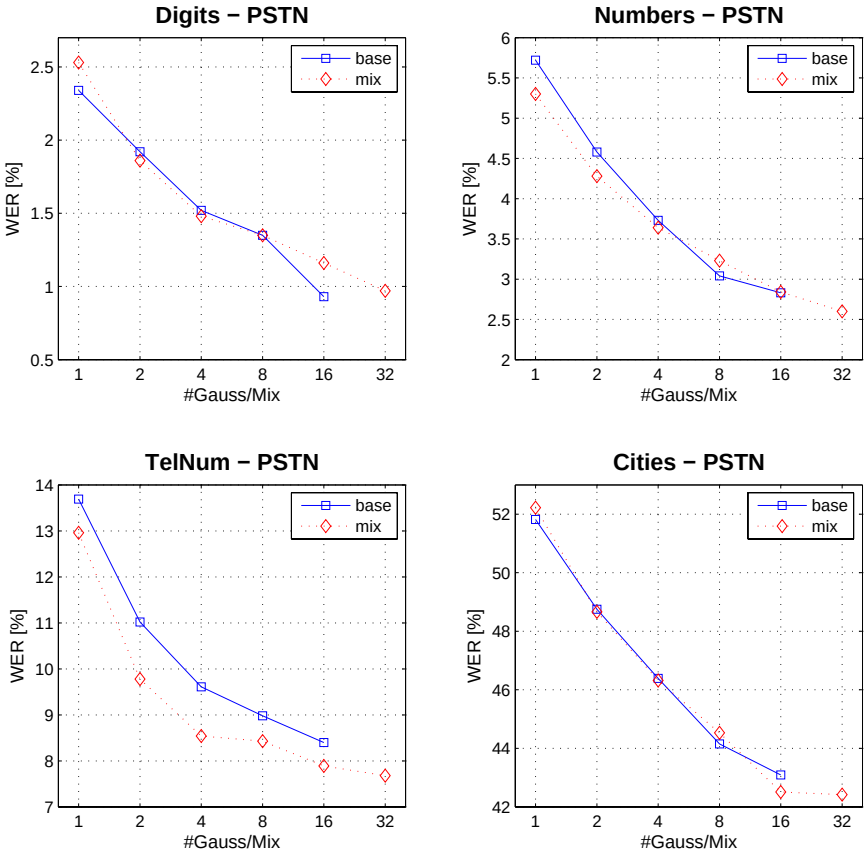


Figure 1.10: Baseline and mixed training model results on PSTN data.

hypothesis for the utterance being recognized) of the `Cities` task, the WER drops from about 43% to 21% with $N = 10$ for 16 Gaussian models; this large drop is due the many city names that are phonetically similar (there are mane homophonic pairs, i.e. words that have the same representation in terms of phonemes) which cause confusion errors.

For the mixed training models, note that the results improve slightly on the PSTN tasks and degrade (again slightly) on the GSM tasks. This can be explained by a bias towards PSTN corpus which is about three times larger than the GSM corpus, but still benefits from the information brought by the extra data.

The following training scheme is adopted for all models: first, single Gaussian context-independent units are trained on the small training data (`Allophones`), then context-dependent models are created and the parameters are initialized from the CI models. A small perturbation is added to the mean vectors and some Viterbi training iterations are performed on the larger corpus (`PSTN` or `GSM`). The trained CD mono Gaussian models are used to align the training corpus and Gaussian mixture models are estimated with the split/train procedure, but using

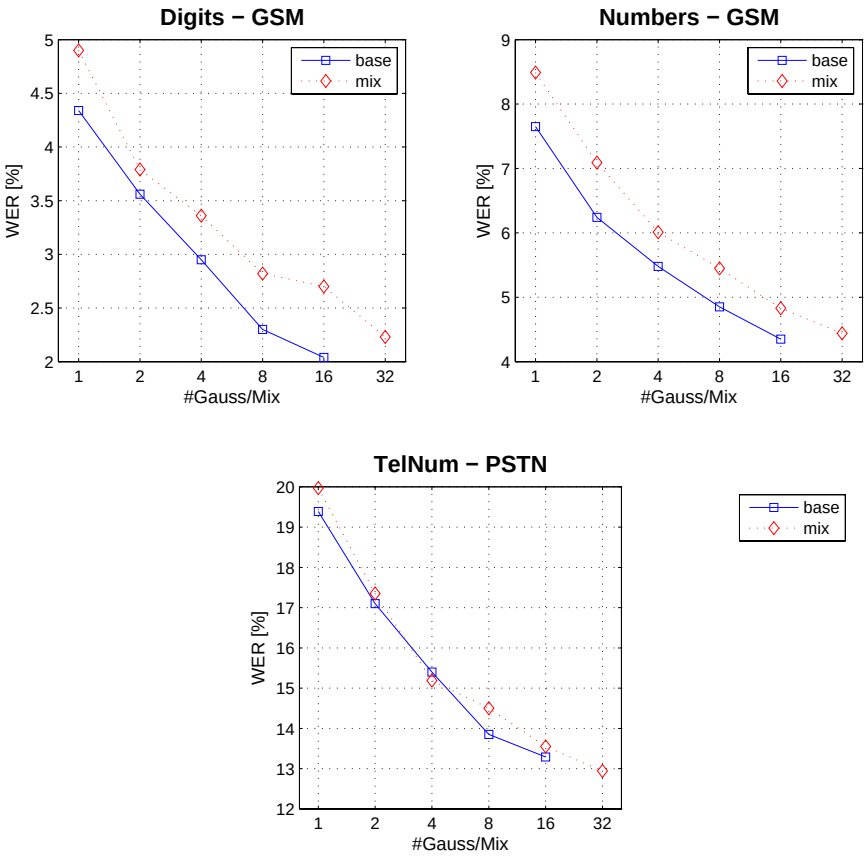


Figure 1.11: Baseline and mixed training model results on GSM data.

the fixed alignments.

Chapter 2

Background on Long Units in Speech Recognition

Here we review work present in the literature about speech recognition with long units. In these systems, the basic representation unit is not the phoneme, but syllables or other longer representation of the sounds in the language; phonemes can be still a part of the units, as some units correspond to a single phoneme.

There is a large body of work done on syllable based speech recognition for Japanese and Chinese, which is not explored here mainly because those languages are naturally syllabic and the focus of the present work is on the use of syllables with languages that are treated mainly with phoneme-like units in speech recognition.

In the following sections, the literature survey is divided by site (university, group, company); the works are ordered chronologically and finally we conclude with a synthetical discussion of the results.

2.1 ISADORA, Universität Erlangen-Nürnberg

ISADORA is a speech recognition system [Schukat-Talamazzini 1993] that allows the construction of complex speech modeling networks. Speech units are represented through a hierarchy of units, so every unit in a given layer can be decomposed into strings of the units in the subsequent layer. There are six layers: compound words (target language is German), words, syllables, demi-syllables, cluster (initial/final consonant cluster as well as vocalic nucleus) and phones. An HMM is assigned to a unit if its occurrence count is larger than a given threshold.

In [Schukat-Talamazzini 1992], the so-called *Context-Freezing Units* (CFUs) were introduced (same hierarchy of units as mentioned before) as a large sub-word unit for speech recognition. The idea behind CFUs is to approximate the use of whole word models given limited training data, and then capture the contextual variation by training models for the resulting suprasegmental units. These units are exclusively context-independent.

The system uses discrete HMMs (soft VQ with 3 labels) with 2-4 states for phone-like units; state-transition and output distribution probabilities are trained with the Viterbi algorithm and a state replication technique is adopted as duration model with a replication factor adapted after each iteration. Training data consists in 90 utterances from 4 speakers (two male and two female speakers) and test data consists in 10 utterances from the same 4 speakers, but with a different vocabulary; the perplexity of test grammar is 44. For the same number of PDFs, CFUs perform better than triphones, but are outperformed by triphones when the number of PDFs is over 2000. It was also shown that with adequate back-off strategies, triphones perform comparably to the CFUs.

There are some points that are not very clear: there is no indication about which CFUs were retained after training nor their average length (in phonemes); moreover, it seems that the test data is not large enough to provide statistically significant differences in the results. The most interesting feature is the hierarchical organization of the units, that enables robust parameter estimation and a vertically more complex modeling, provided that increasing amounts of training data are available.

2.2 ATR labs

At ATR labs, in Japan, non-uniform unit HMMs [Matsumura 1994, Matsunaga 1995, Matsumura 1995b, Matsumura 1995a] were investigated in Japanese speech recognition. The non-uniform units consist in sequences of phonemes with different lengths. The motivation is to provide more precise acoustic models than CD units, as these units should be more efficient in capturing long-distance contextual influences. Another point is to create units that cover the allophonic variations in target speech (for a particular evaluation task), because it is futile (sic) to generate non-uniform units that are not entirely contained in target speech [Matsumura 1995a].

Preliminary experiments [Matsumura 1994] showed that longer units (syllables) provide good performance in speech recognition; this fact motivated the use of the (long) non-uniform units.

Since a high occurrence of insertions (syllable /ra/ in place of /a/) was observed and since those insertions severely degraded the performance, the alignment of the training data was "restricted" [Ariki 1994], so that the data used to calculate the forward-backward variables were constrained to a marginal section around manually-defined phoneme boundaries.[1] Training data consists of 2620 isolated words, adaptation and test data has 1103 and 274 phrases respectively and text data comprises about 9.3×10^4 Japanese phrases represented with phonetic symbols; the need of adaptation data will be clear when the unit selection process is described.

Phoneme models (26 models) have 3 states with 5 mixture components and non-uniform models (96 models) have $3 \times s$ states and 5 Gaussian components per mixture, where s is the number of phonemes in the non-uniform unit. Non-uniform units provided the best performance in isolated word recognition as Table 2.1 shows (figures are correct word and syllable recognition). In Method I, syllable models are created if there are enough occurrences in training data, otherwise concatenation of phoneme models is used; in Method II, both phoneme and syllable models are simultaneously used and the model with highest probability is selected. Syllables are seen as non-uniform units in this context.

Method	syllable	word
Phoneme model only	90.42%	88.8%
Syllable model only	94.63%	82.6%
Non-uniform (Method I)	92.06%	87.0%
Non-uniform (Method II)	94.12%	89.5%

Table 2.1: Syllable and word recognition results.

It is not clear why the word recognition rate is lower for syllable models (in isolation from the other models) compared with the other models; it would be expected that as the syllables are correctly recognized, so would be the words.

The procedure used to select the non-uniform units is illustrated in Figure 2.1. All data sources (text, training, adaptation and development data) are from the same application domain, i.e. conference registration. First, phoneme models are learned from training data. Then units are selected from text data on the base of occurrence frequency. Initially the frequency of all two unit (starting with phonemes) sequences is calculated and the sequence with highest frequency is selected. [2] If this sequence satisfies the acoustical constraint (described below), it is accepted

[1] This procedure has the nice side effect of speeding up the training procedure.

[2] This process heuristically minimizes the entropy of the training data; it was found that there is no difference in performance in comparison to a formal process.

as a new unit and the process is repeated. If the sequence is rejected, the sequence with next rank in frequency is examined and so on. The acoustical constraint is that the candidate unit (after training) should achieve higher acoustic likelihood on the development data. Initial models are created by concatenation of phoneme models. The candidate unit model is trained on the adaptation data and the phoneme models are also retrained using the EM algorithm. Repeating the algorithm 100 times resulted in 37 non-uniform units with an average length of 2.6 phonemes.

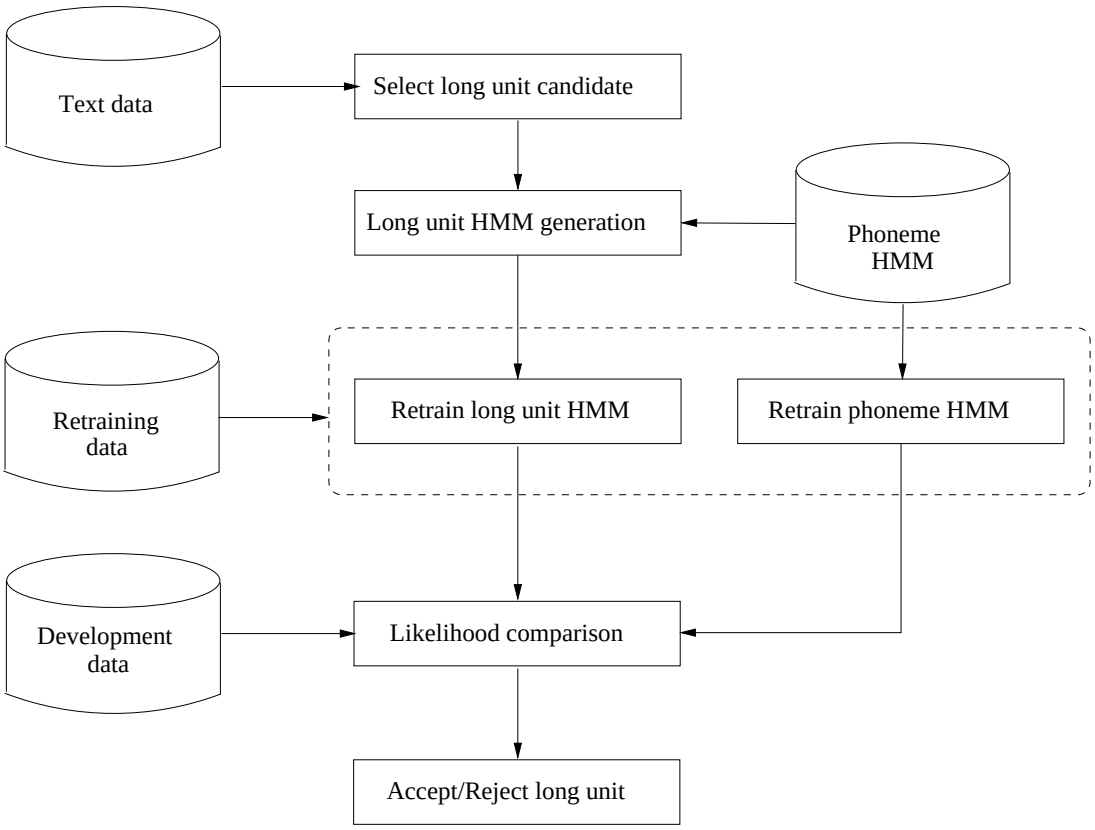


Figure 2.1: ATR Labs non-uniform unit generation.

Tests are performed on sections where the long units are applicable. Results showed that 97.5% of these sections are correctly recognized and long units have higher likelihood than the concatenation of phoneme models on 80% of all sections correctly recognized. Non-uniform models were trained on fewer data than the phone models.

In [Matsunaga 1995] and [Matsumura 1995a] the experiments were expanded to include continuous speech. The results presented here refer only to [Matsumura 1995a], because it is a more developed work than the former. Data sources are the same as previously described (for each speaker), but the test set now consists of 267 phrases for speaker MHT and 277 for speaker MAU; the phoneme perplexity of the LR-phrase grammar is 5.9. The baseline models consists

in 26 context-independent (CI) phoneme models with 5 Gaussian mixtures and 3 states trained with the EM algorithm and CD models generated using the Successive State Splitting (SSS) algorithm [Ostendorf 1997]. The HMnet for the CD models was composed of 600 (single Gaussian) states. The results for the baseline CD models and for these models after vector-field smoothing (VFS) adaptation are given in Table 2.2. Due to the context-free grammar restrictions, crossword units could not be evaluated.

Retraining	Baum-Welch		VFS
Speaker	CI models	CD models	CD models
MHT	9.4%	8.7%	8.3%
MAU	8.7%	8.7%	9.4%
Average	9.0%	8.7%	8.9%

Table 2.2: Baseline sentence error rates.

Using the adaptation data, CI non-uniform units are trained (using restricted training) and 48 units are generated for speaker MHT and 61 for MAU, by iterating the algorithm 100 times. The average length of the units are 3.0 (MHT) and 2.8 (MAU). It was noted that when long units are accepted, the occurrence counter was on average 55.4 (MHT) and 74.5 (MAU) and when they were rejected, the counts were 20.8 and 20.3 respectively. This shows that when there is more training data, long units tend to be accepted. In Table 2.3, sentence error rates for phrase recognition are presented; there are 24 (MHT) and 27 (MAU) long units (this was a preliminary experiment, that is why there are fewer units). The use of CI non-uniform units resulted in an improvement of 18% and 17% over CI and CD baseline results (on average).

Speaker	Phrase error-rate
MHT	7.6%
MAU	7.2%
Average	7.4%

Table 2.3: CI non-uniform units, Baum-Welch retraining.

Training of CD long units was also evaluated. The initial models are created from the CD phoneme models in the HMnet and after retraining, the states of the resulting model are not part of the HMnet anymore. The phoneme sequence candidates included the preceding/succeeding phonemes sets. Due to data sparseness, the models are retrained (adapted) using VFS. Repeating the selection algorithm 100 times resulted in 67 (MHT) and 72 (MAU) long units. The average length of the units is 2.6 (MHT) and 2.7 (MAU), not including the preceding/succeeding

phonemes. The occurrence counters for acceptance are now much lower (as expected), on average 17.2 (MHT) and 25.3 (MAU) and 6.7 (MHT) and 6.5 (MAU) when rejected. With respect to the number of parameters, CD models resulted in a slight increase of 10%. Phrase recognition results are presented in Table 2.4, where 37 (MHT) and 32 (MAU) long units are retained (due to the threshold on the amount of training data). The improvement brought by the CD long units is of 20% and 19% over the CI and CD baseline results and of 3% over the CI long units (average).

Speaker	Phrase error-rate
MHT	7.2%
MAU	7.2%
Average	7.2%

Table 2.4: CD non-uniform units, VFS retraining.

The results using CI and CD non-uniform units did not present a significative difference. The reasons given were: the total number of parameters is not that different; the same beam-width was used in recognition, but the LR-parsing space for CD units is much larger than for CI units and finally, the retraining algorithm is different, it was used EM and restricted training for the CI units while VFS was used for the CD units.

Using isolated speech may have an effect on the models, since it may limit contextual modeling for the word boundaries units (not of concern for the presented evaluation, as models are non crossword). The proposed procedure for selecting long units is interesting because it is based on data-driven approaches and there is an acoustical validation of the candidate units before acceptance. With the availability of larger training databases, more contextual information may be added in the selection of the units, such as information about syllable or word boundaries, because prosody might influence the realization of the units in these contexts.

2.3 KAIST

In the Korea Advanced Institute of Science and Technology (KAIST), a neural-network (NN) system, called U-net, using non-uniform units was developed [Yu 1995] [Yu 1996] [Yu 1997]. Non-uniform units are defined on segments in between stable regions of the cepstral coefficients and segmentation of the speech signal is performed before recognition. The network structure used to model these units is based on the fact that the units are composed of stable parts (the

edges) separated by a region where the feature vectors present a transient behavior. Different features are used for the stable and transition parts; stable parts use the static cepstral coefficients while the transition parts use the Δ coefficients. As the units span several phonemes (up to four), they capture the contextual effects among the phonemes. This system was targeted at word-spotting; recognition errors are caused either when a keyword is not detected in the speech signal or when a word not present in the signal is detected (false alarm).

The recognition system consists of three modules: the first module segments the speech signal, the second module classifies the segments into units and the last module detects the words. The block representation of the system is depicted in Figure 2.2.

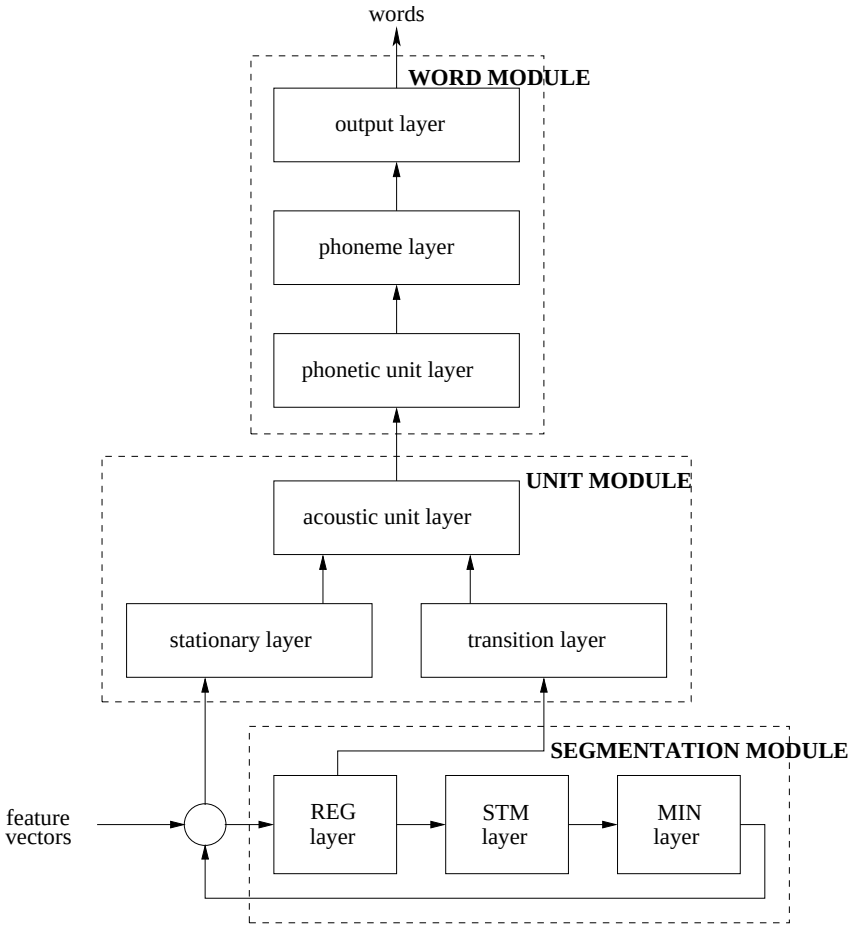


Figure 2.2: Structure of the U-net system.

The units are defined on acoustic data only. The first module scans the feature vectors with a window of seven frames shifted by one frame at a time, and this module segments the signal by "cutting" it in the middle of quasi-stationary points. The method has a low computational load and it segments the stream of feature vectors as follows. The REG layer (cf.

Figure 2.2) calculates regression coefficients that represents the slope of the time function of the parameters [Furui 1986]. At time t, the output vector $\vec{r}(t)$ [3] is:

$$r_m(t) = \sum_{n=-N}^{N} x_m[t+n] \cdot w_n^r \qquad 1 \leq m \leq p \tag{2.1}$$

Where N defines the length of the scanning window ($N = 3$) and p is the dimension of the feature vector, the weights w_n^r are fixed to n and $x_m[t+n]$ is the mth element of the feature vector $\vec{x}$ at time $t+n$. The REG layer has then p computing units (a computing unit is called hereafter as "node").

The next layer, STM, calculates a spectral transition measure [Furui 1986] which is weighted to reduce the segmentation variation. Higher output values indicate a rapid change in the feature vectors. At every time step t it produces an output:

$$s(t) = \sum_{m=1}^{p} |r_m(t)| \cdot w_m^s \tag{2.2}$$

The weights w_m^s connect the mth output of the REG layer and the node in the STM layer and are trained to minimize the number of generated units; the training procedure (unsupervised) is described in [Yu 1995] and it is separated from the training of the other networks belonging to the other modules.

The MIN layer detects points where $s(t)$ is locally minimum. Units are defined in between points where the MIN layer's output is activated. MIN layer searches for minima of the STM layer output at the center of the seven frame window and its output is calculated as:

$$M_t = \mathrm{f_h}\left(\sum_{n=-N}^{N} f_h(s(t+n) - s(t)) - \theta_M\right) \tag{2.3}$$

where the function f_h is a hard-limiting non-linearity and θ_M is a threshold that controls the sensibility of the local minima detector.

Unit module's inputs are the two vectors at each edge of the unit and the outputs of the REG layer for the frames in between these four vectors. The static coefficients are inserted into the stationary layer and the other are inserted into the transition layer. Nodes in the acoustic unit layer are connected to those in the stationary and transition layers.

The word module plays the role of lexicon in the system. Nodes in the phonetic unit layer

[3] The output of the REG layer are related to the so-called Δ coefficients.

are connected to those in the acoustic unit layer and are active when the linked acoustic unit nodes are active. The connections between the nodes in the phonetic and acoustic unit layer are many-to-many because the different segments may contain the same phoneme. Let word w be composed (in a pronunciation dictionary) by n phonemes $p_1, p_2, \cdots, p_n$ then in the phoneme layer there are n nodes corresponding to the phonemes and they are unique to word w. The nodes in the phoneme layer are activated only when they are activated in order, otherwise their output is reduced by a penalty d. Nodes in the phoneme layer are connected to many nodes in the phonetic unit layer. Figure 2.3 illustrates the connections between the nodes in acoustic, phonetic, phoneme and word layers for a given word w. The mapping provided from the acoustic unit to phonetic and phoneme layers is needed because it makes it simpler to segment speech into the acoustic units and to describe words in terms of phonemes.

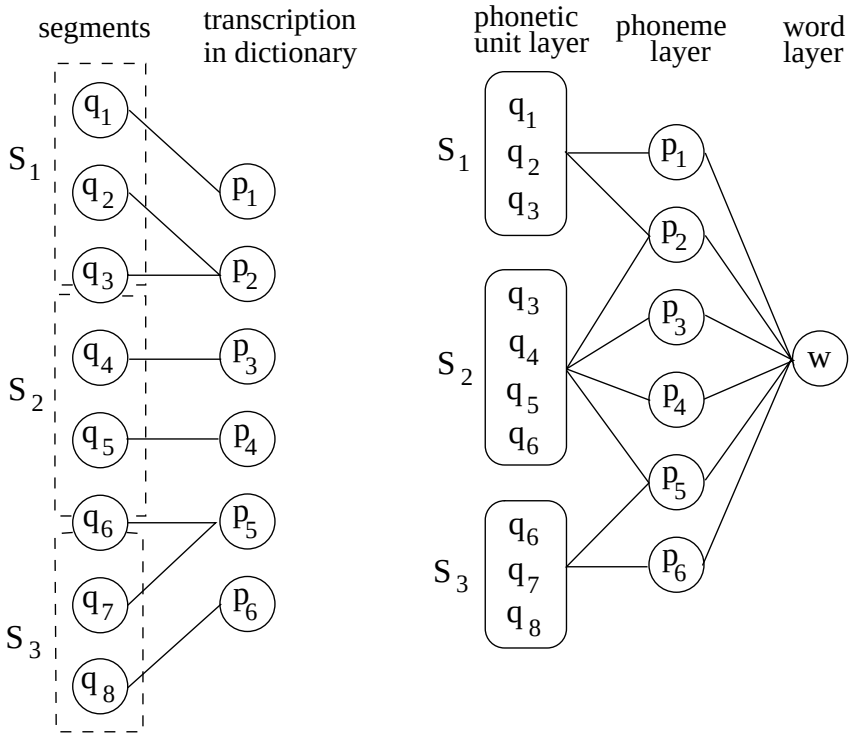


Figure 2.3: Links between the phonetic unit, phoneme and word layers.

The weights in the last two modules are trained using supervised Hebbian learning [Yu 1996] and phoneme-segmented speech. The phonemes in a segment define the unit; if the unit corresponding to a segment has not been created, a node is added to the acoustic unit layer and corresponding nodes and links are created in the stationary and transition layers. Nodes and links are also created in the phonetic unit layer. When a new word is added, the necessary nodes

(one per phoneme) are added to the phoneme layer and the links to the phonetic unit layer are also created. Multiple utterances with a given unit cause the update of the weights associated to the unit (in all layers in the last two modules). Training is performed by running a recognition pass and the segments causing an error (not detecting a word) are used to train the units.

Evaluation of the system is performed using a subset of the `TIMIT` database. Feature extraction generates 14-dimensional LPC cepstral coefficients using a window of 16msec, shifted by 8msec. Table 2.5 shows the test conditions and results.

vocabulary		520 words		1000 words
segmentation		transition	stationary	stationary
Number of sentences	training	900	900	1782
	test	150	150	381
test keywords		594	594	1213
Number of speakers	training	368	368 (72+296)	623 (190+433)
	test	103	103 (9+94)	215 (53+162)
Number of units	defined	5339	5068	7902
	test	5710	4077	7886
% unit correct		48.3	59.3	57.5
% word correct		51.2	77.8	65.5

Table 2.5: Test conditions and results.

In Table 2.5, the data presented in column "transition" are related to setting the segment boundaries at peaks of the MIN output instead of valleys. This shows that defining segment boundaries in transition regions results in degradation because the segmentation is less stable. It should be noted that 411 (10.1%) units in test data are unseen units and 80 (13.5%) words in test data are missed because of the unseen units; when the word detection rate is 77.2% at a false alarm/keyword of 23.6%, about 59% of word errors are due to unseen units. The system is evaluated against dynamic time warping based on whole word models (1000 words task) and provides much higher detection rates, roughly 30% on absolute values. The simulation time was reduced by pre-selecting units using only the two vectors at the ends of the segments. This decision was taken to reduce the time needed to evaluate all the activation values in the acoustic unit layer. The number of nodes that needed full evaluation was reduced to 10% of the total and that resulted in no degradation in the detection rate. Real-time (RT) factors were also presented; during training a factor of 0.7RT is attained and during recognition, factors of 1.0RT and 1.6RT are measured for vocabularies of 500 and 1000 words respectively.

2.4 Center for Spoken Language Understanding

In [Hu 1996] syllable-like units were employed in speech recognition (American English). The system uses trajectory models [Goldenthal 1994] to model speech; this type of system needs accurate segmentation into units. It is very hard to get accurate phoneme segmentation as the number of possible boundaries is huge, making an exhaustive search impracticable. Moreover, some transitions are difficult to detect such as vowel-vowel or liquid-vowel. To overcome those limitations, the transitions between phonemes are captured by grouping those phonemes together and treating them as a single unit. Modeling with syllables makes it possible to capture those difficult to detect transitions, not only within phoneme boundaries but also in other hard to segment regions.

Sequences of phonemes in which it is difficult to find boundaries are grouped together using pre-defined rules, like merging a stop and a vowel, or a vowel and the adjacent semi-vowel; word pronunciations are defined using the resulting units. An acoustic segmentation algorithm is used to pre-segment the speech signal before decoding. The parameters are tuned to detect rapid spectrum changes and it is common to over-generate segments; so the detection of boundaries between units is easier (from the definition of units) and the phoneme boundaries that are not detected make no problem because they are ignored in the design of the units. The trajectory models used in the work are described in [Goldenthal 1994] and 10 states are used for each unit model; artificial neural networks are trained to estimate probabilities for the units (Gaussian mixtures were not successfully integrated due to the lack of training data.) Search is implemented with a time-synchronous Viterbi algorithm; only boundaries detected by the segmentation algorithm are considered as unit boundaries and just a limited number of look-ahead segments are considered for each state in the search, significantly reducing the computational effort.

The evaluation task is the recognition of the twelve months of the year; there are 814 files for training and 796 for evaluation. Perceptual Linear Prediction (PLP) coefficients (8 coefficients) plus log-duration of the units are used as features. Trajectory models are estimated from forced-alignment boundaries (using existing phoneme-based models) and evaluated with forced-alignment and automatically detected (with the segmentation algorithm) boundaries; the evaluation results, in terms of word error rate are given in Table 2.6

Training was performed using boundaries detected with a different recognizer, thus perfor-

Forced	Automatic
3.5%	5.2%

Table 2.6: Word error rates for boundaries detected with forced-alignment and with the automatic algorithm.

mance may be sub-optimal as it is not sure that the segmentation algorithm would produce similar segmentations as the forced decoding. The initial syllable-like models were used to re-segment the training data and new models were estimated; the error rate dropped form 5.2% to 4.5%. Note that the training and test vocabularies are the same; remember that our own results presented in Figure 1.8 show that there is a strong bias when the training and test vocabularies are the same. Thus it is hard to draw conclusions in those conditions.

2.5 Technical University of Munich

To handle co-articulation effects, macro demi-syllables (MDS) were introduced in [Pfau 1997]. MDS are constructed to explicitly include as many phonemes as possible. Using demi-syllables (phonemes are included) as base units has the advantage that most of the phonotactic constraints are implicitly incorporated [Plannerer 1993]. In German there are 54 initial consonant clusters, 160 final consonant clusters and about 130 vowel clusters including diphthongs and "syllabic consonants". The size of MDS is in the range from a single demi-syllable up to a whole word.

The vocabulary is decomposed into a set of MDS that maximizes a quality measure Q. This quality measure takes into account both the frequency of occurrence of all the MDS used to represent a word and the total number of MDS. The following value defines the rating of the frequency ($freq$) for a given MDS_k that appears in training data:

$$q\left(MDS_k\right) = f\left(F_{min}, F_{max}, freq\left(MDS_k\right)\right)$$

where f is a sigmoidal function; this function has a strong rise with the value of $freq()$. The values F_{min} and F_{max} fix the turning points of the sigmoidal function: the MDS enters the inventory if its frequency of occurrence is greater than F_{min} and F_{max} sets a limit on q, so all the MDS that appear with high frequency contributes in the same way to the quality measure (2.4).

The quality of the decomposition of a word W_i into k MDS is defined by the product:

$$Q(W_i) = \prod_{k \in W_i} q(MDS_k) \tag{2.4}$$

The average quality of the whole vocabulary cannot be exactly maximized, so an iterative procedure was developed. The quality Q of word W_i is maximized independently of the other words. The procedure goes as follows:

1. Build all possible MDS with sizes from the base unit (demi-syllables) up to whole words;
2. Calculate the frequency rating for every MDS;
3. Eliminate the MDS with frequency less than F_{min};
4. Determine the best decomposition of every word W_i by maximizing $Q(W_i)$. The quality measure $q(MDS_k)$ for every MDS is kept constant, regardless of the use of the MDS in other words. The maximization is carried out using the Viterbi algorithm;
5. Count only the MDS actually used after decomposing the vocabulary and calculate a new $q(MDS_k)$;
6. Proceed with step 3.

Initial models for the MDS can be created by concatenating the models of the base units; the problem with this approach is the increase in the number of parameters to be estimated. The solution proposed to control the increase of the number of parameters was that the states of the resulting MDS could select the components of their mixtures in a general codebook[4] of the base unit; they can, however, have their own set of mixture weights. To allow more flexibility, the states of the MDS can share the codebooks with the neighboring states up to a given radius r; this implies an enlargement of the mixture sets; this sharing with the neighboring states is believed to help in dealing with the co-articulation effects. Starting from phoneme models, the number of mixtures is raised from 20k to 1.14M ($r = 1$). Parameter estimation is performed with Viterbi training and components with very small mixture coefficients are removed.

Experiments are performed using the `Vermobil` database (CDs 1-5, CD7 and CD12); a total of 40000 possible MDS could be found with size ranging from 1 up to 15 phonemes. Training

[4] A Gaussian codebook is simply a set of Gaussian densities.

vocabulary consists of 7000 words that could be decomposed with few iterations as shown in Table 2.7.

F_{min}	F_{max}	# of iterations	# of units
20	50	8	738
20	100	6	607
50	100	7	595
100	500	4	264

Table 2.7: Number of iteration until a stable set of MDS is found.

Training data amounts to 27 hours and test data to 41 minutes of spontaneous speech dialogues. The dimension of the feature vectors is 66 (no more information about the front-end) and the following models are Viterbi trained:

- 54 context-independent phonemes (3 to 4 states), with a total of 176 states and 161 diagonal Gaussian codebooks; the state transition probabilities are trained individually for each model-state;

- 595 MDS models (from 3 up to 39 states) with a total of 7273 states using the same number of Gaussian densities; and

- 2100 triphone models (crossword).

The test results with these models are presented in Table 2.8. MDS perform almost as good as triphones with a slightly larger number of parameters. We should note that the dimensionality of the feature vectors (66) is quite high and might cause some estimation problems and may also influence on the choice of the radius r of states that can share Gaussian codebooks. The quality measure that is proposed inspired our own measure to help in the back-off for unseen units (cf. Section 3.2.2).

Models	word rec. rate	word error rate	# of parameters
phonemes	69.2%	35.4%	2.5M
MDS ($r = 1$)	78.2%	25.8%	3.6M
triphones	78.5%	25.2%	3.5M

Table 2.8: Test results and total number of parameters.

2.6 Télécom Paris / ENST

Multigrams [Deligne 1997a, Deligne 1997b] were designed to model sequences of text symbols with variable length. Multigrams can be seen as a production model that produces a string $\mathbf{Z}$ composed of a sequence of units drawn from $\{z_i\}$. The actual sequence of multigrams is hidden and it needs to be determined from an observable sequence of strings $\mathbf{O}$ (via a segmentation $\mathbf{S}$), which is shown in Figure 2.9.

$$\begin{array}{lccccccc} \mathbf{Z}: & z_{i1} & & z_{i2} & & z_{i3} & \cdots \\ & \Uparrow & & \Uparrow & & \Uparrow & \\ \mathbf{S}: & [o_1\ o_2] & \oplus & [o_3] & \oplus & [o_4\ o_5\ o_6] & \cdots \\ \mathbf{O}: & o_1\ o_2 & o_3 & o_4\ o_5 & o_6 & \cdots & \end{array}$$

Figure 2.9: Segmentation of an observation with multigrams.

This model is extended by *joint multigram* models. When a multigram unit is produced, a set of parallel observation strings is observed instead of a single one. For speech recognition, a pair of observations is sufficient (but could be expanded), e.g. the feature vector and the corresponding unit label. The segmentation $\mathbf{S} = (\mathbf{S_O}, \mathbf{S_\Omega})$ now refers to a segmentation of the two observed strings $\mathbf{O} = (o_1; o_2; \cdots)$ and $\mathbf{\Omega} = (w_1; w_2; \cdots)$ as illustrated in Figure 2.10; note that the two strings do not need to have the same length. Finding a joint segmentation of the two strings is the same as creating a many-to-many mapping between them; each string in $\{z_i\}$ is a mapping between the strings in the alphabet of $\mathbf{O}$ and $\mathbf{\Omega}$.

$$\begin{array}{lccccc} \mathbf{Z}: & & z_{i1} & & z_{i2} & \cdots \\ & & \Uparrow & & \Uparrow & \\ \mathbf{S} = (\mathbf{S_O}, \mathbf{S_\Omega}): & \begin{bmatrix} o_1 \\ w_1\ w_2 \end{bmatrix} & \oplus & \begin{bmatrix} o_2\ o_3 \\ w_3\ w_4\ w_5 \end{bmatrix} & & \cdots \\ (\mathbf{O}, \mathbf{\Omega}): & o_1\ o_2\ o_3\ \cdots & & & & \\ & w_1\ w_2\ w_3\ w_4\ w_5\ \cdots & & & & \end{array}$$

Figure 2.10: Segmentation of an observation with joint multigrams.

For any given set $\{z_i\}$ of multigrams, the optimum string $\mathbf{Z}^*$ and segmentation $\mathbf{S}^*$ in the ML sense are given by:

$$(\mathbf{S}^*, \mathbf{Z}^*) = \arg\max_{(\mathbf{S},\mathbf{Z})} \mathcal{L}\left(\mathbf{O}, \mathbf{S}, \mathbf{Z} | \{z_i\}\right)$$

Solving for the optimal set $\{z_i\}^*$ is not directly possible. An iterative approach with the EM algorithm is adopted; the details of the formulation are given in [Deligne 1997b] and will be omitted here. At each iteration the data likelihood and model complexity are optimized.

From an initial set of multigrams $\{z_i\}^{(0)}$ a 2-step iteration process is repeated (also at the kth iteration): reduce complexity of $\{z_i\}^{(k)}$ by excluding the multigrams with very low probability and re-estimate the parameters (probabilities of the multigrams) to maximize the data log-likelihood $\log\left[\mathcal{L}\left(\mathbf{O},\mathbf{S},\mathbf{Z};|\,\{z_i\}^{(k+1)}\right)\right]$; the procedure in case of joint multigrams is similar.

Multigrams can be derived from text or from speech data. In the case of speech data, a sequence of feature vectors is decomposed in a sequence of Temporal Decomposition [Atal 1983] (TD) target vectors. TD describes a speech segment as a linear combination of vectors (target vectors) which are expected to have less variability than the original vectors; the use of TD reduces the number of frames by a factor of 4.7 and the average number of variable-length sequences of TD are modeled by multigrams and a HMM is associated to each multigram. It is reported in [Deligne 1997b] that the iterative procedure does not allow to create a new HMM even if acoustic modeling could be improved. LPCC coefficients (order 16) are calculated every 10 msec and the TD targets are calculated. Discrete HMM are used and TD vectors are vector quantized with (32, 64 and 128) codewords. The number of acoustic symbols in a sequence is limited to 5 and at every iteration, multigrams with less than 10 occurrences are discarded. That means that when all multigrams appear at least 10 times, it is accepted that the set is stable.

Using acoustically derived units provides a criterion that is more consistent (and possibly more reliable) for speech recognition. Joint multigrams are used to derive a probabilistic mapping between sequences of acoustic units and sequences of phonemes. The set of multigrams $\{z_i\}$ defines an intermediate representation between acoustic and phonetic levels; the recognition process is divided into 2-steps: first, the acoustic observation $\mathbf{O}$ is decoded into the most probable structure $(\mathbf{S}^*,\mathbf{Z}^*)$ and second, the phonetic string with maximum likelihood is retrieved from $\mathbf{Z}^*$. Note that the resulting concatenation of sequences may result in non-lexical sequences. During actual decoding the candidates are restricted to only those that are lexically valid and then a n-gram language model can be used.

Training data consists in 1 hour of speech from a single speaker and test data consists in 30 min of speech from the same speaker. Both sets are from the same task, sailing weather forecast in French. Three reference systems are trained:

- 35 phoneme HMMs
- 1599 triphone HMMs plus the 35 phoneme HMMs

- 401 HMMs modeling the words in training vocabulary

As TD vectors are used, the phoneme HMMs have only 1 state. Better results are obtained with a codebook of size 32 and the corresponding results are reported in Table 2.11. The authors's

	Acoustic			
	phonemes	triphones	words	multigrams
Number of HMMs after convergence	37	1636	403	613
Average frequency of model in training set	1280.0	29.9	34.6	30.4
Average number of states per model	1	1	4.7	3.5
Phonetic accuracy with no language model	47.0%	55.5%	81.2%	70.6%
Word accuracy with bigram language model	83.9%	86.5%	82.3%	76.1%

Table 2.11: System comparison with phonetic units and multigrams.

explanation about the better performance of multigrams (phonetic accuracy) in comparison to phonemes and triphones with no language model is that some linguistic information is already encoded in the multigrams which is not the case for phonemes or triphones and the interface between the linguistic and multigram level is not optimal as both are optimized separately. It can be said that this interface is even preventing the search to find the correct path, otherwise multigrams would still have better performance using a bigram language model. The average occurrence frequency of triphones, words and multigrams is very similar, that may be linked to a high number of short words in the speech data. As words and multigrams have more states than triphones, it is possible that the amount of training data also plays a role in the lower performance of the larger models in terms of word recognition; word models still have good performance as they represent quite well the test data.

2.7 The Queen's University of Belfast

To reduce the contextual effects (co-articulation, transitions) between phones, multi-phone strings are used as alternative to context-dependent phones for speech recognition. Another advantage is the multitude of transcriptions for monophone sequences that may increase recognition accuracy. The use of phone-pairs as multi-phone string was studied in [O'Neill 1998]. The following advantages of multi-phones are listed (from [O'Neill 1998]):

- to reduce in the loss of contextual information;
- to reduce in the loss of inter-phonetic correlation and transitions;
- to alleviate the problem of too short units; and

- many syllables are contained in two to three phonemes-long sequences.

The main difference between phone-pairs and diphones is that the influence of both left and right phonemes is simultaneously included. In the study [O'Neill 1998] phone pairs are context-independent and two transitions are valid for a given word w surrounded by silence; in terms of "phonemes"

$$w = \text{SIL a b c d e f g h SIL}$$

And in terms of phone-pairs:

$$w = \text{SIL a\_b c\_d e\_f g\_h SIL}$$

or

$$w = \text{SIL\_a b\_c d\_e f\_g h\_SIL}$$

The existence of two valid transcriptions resulted in a increase in performance. As expected, the distribution of the phone-pairs is not uniform; in the `WSJCAM0` database there are about 1500 phone-pairs but there are 12 pairs that appear only once and 200 having less than 5 examples. To cope with unseen pairs, a simple concatenation of phone models is used.

Evaluation of the phone-pairs is performed on both `TIMIT` and `WSJCAM0` databases. Phone-pairs are modeled with 6 state HMMs with Gaussian mixture PDFs; best results are obtained with 15 Gaussian components per mixture. Table 2.12 presents the results for phoneme recognition using a phone bigram matrix (MB) and insertion penalty (IP); IP and MB are used to reduce the gap between recognition and accuracy.

15 Mixtures	Recognition	Accuracy
Original	77.74%	61.18%
$MB + IP = -30$	72.49%	65.08%
$MB + IP = -40$	71.11%	65.27%
$MB + IP = -50$	69.97%	65.12%
$MB + IP = -60$	68.34%	64.13%

Table 2.12: Phoneme recognition and accuracy rates on `TIMIT` using phone-pairs, insertion penalty (IP) and matrix bigram (MB).

Word recognition tests are performed on the `WSJCAM0` database using the 5k bigram model. It is used 15 components for phone-pairs and 12 components for left-context (LC) diphones; the effects of including multiple transcription are also evaluated as Table 2.13 shows.

Model	Word	
	Recognition	Accuracy
1550 LC biphone (12 mixtures)	70.39%	66.97%
phone-pairs (15 mix) SED	70.89%	67.69%
phone-pairs (15 mix) DED	76.85%	73.17%
phone-pairs (15 mix) TED	79.34%	76.67%

Table 2.13: Word recognition and accuracy on `WSJCAM0`.

In Table 2.13 SED, DED and TED stand for: SED = Single Entry Dictionary (one of the phone-pairs transcription), DED = Double Entry Dictionary (both phone-pairs transcriptions), and TED = Triple Entry Dictionary (DED plus the monophone dictionary).

The use of various transcriptions is efficient in reducing the error rate, but no analysis was presented on which kinds of errors are reduced with the different transcriptions. The phone-pair units are more efficient than diphones, but it seems that triphones achieve even better performance than the phone-pairs. The contextual effects that can be modeled by phone-pairs is limited and the use of context-dependent longer units should improve even further the recognition rates.

2.8 MIT Laboratory for Computer Science

In [Bazzi 2001], long units were used to detect out-of-vocabulary (OOV) words. Although the focus is not on speech recognition *per se*, it is interesting to describe the procedure for selecting the variable-length units. The chosen approach is a bottom-up one where units are "glued" to form a larger one by selecting the pair with the highest (weighted) Mutual Information (MI). The MI is weighted by the joint probability $p(u_1, u_2)$ of the units u_1 and u_2 occurring together and it is given by:

$$MI_w(u_1, u_2) = p(u_1, u_2) \log\left(\frac{p(u_1, u_2)}{p(u_1)\, p(u_2)}\right)$$

The MI_w measures how much information about the neighboring unit u_2 is contained in the unit u_1. If the two units are independent then $MI_w(u_1, u_2) = 0$.

Units are merged in an iterative way: it starts from phonemes as initial units and then the MI is calculated for all unit pairs encountered in the training vocabulary; the pair (u_1, u_2) with highest value of MI is promoted as a new unit and then every occurrence of the pair is replaced by the new unit $u = u_1 u_2$. The number of iterations is chosen as a trade-off between

the complexity of the model and the recognition speed. If the process is repeated indefinitely, it will converge to the whole vocabulary.

To reduce computation time, at each iteration the top 10 pairs are promoted. The Linguistic Data Consortium PRONLEX dictionary is used; it contains 90964 words with 99202 unique transcriptions. Running the procedure 200 times results in 1977 units. It is worth to notice that about two thirds of the generated units are valid English syllables and the others are either syllable fragments or multi-syllable phone sequences. This approach to OOV detection was the closest one to an "oracle" OOV model, which shows the ability of the MI to select proper units. This procedure is adopted in the present work as well to find multi-phoneme units as described in Chapter 3.

2.9 Institute for Signal and Information Processing

The work reported in [Ganapathiraju 2001] was the first application of syllables in large vocabulary recognition on telephone bandwidth speech. Syllables were chosen because they can model long-term temporal dependencies and they are in close connection to human speech perception and articulation, and they are closely integrated to some co-articulation phenomena. Syllable deletion rate is much lower than that of phonemes, so syllables might be robust to recognize spontaneous speech.

Syllables are represented in [Ganapathiraju 2001] as a concatenation of lexical phonemes; however it does mean that the corresponding acoustic segment contains all the lexical phonemes, due to pronunciation variations such as phoneme reduction or phoneme deletion, that appear in speech. One inconvenience of syllables (at least in English) is the existence of ambisyllabic consonants: consonants that are at syllable boundaries and belong in part to each syllable. The most plausible boundary phone is tagged to denote ambisyllabicity; some syllables appear multiple times with the additional entries marked by the ambisyllabic tag (at the beginning or at the end of the syllable). Stress is not taken into account to make the system simpler and to avoid underestimated parameters. Context-dependent syllables are not modeled to avoid clustering or state-tying, thus reducing the complexity of the system. All units (syllable or triphones) are word-internal, i.e. no unit cross the boundary between words.

Baseline performance was compared against tied-states triphone models with 8 Gaussian mixtures per state. Training data is the `Switchboard` (SWB) corpus (with 60+ hours of data).

Different systems were trained; the first syllable system has 9023 syllables, with models having a number of states proportional to one-half of the median syllable duration, with no skips. Over 8000 of those syllables have less than 100 training tokens and the performance is poor. Another system was then trained with the 800 most common syllables and the remaining syllables are replaced by their phonemic representation; the phone models used to replace the less common syllables are trained separately and are context-independent models, with 3 states and no skip; each state has 32 Gaussian components. Combining syllables and phone models was a pragmatic solution as there is a possible severe mismatch in the junction between those models which were trained in separation.

A further enhancement of that system was to train a system with the same 800 syllables and CI phones from scratch (instead of bootstrapping). Analysis of the errors shows that a high percentage of words with all phone representation is erroneous, thus the CI phones were replaced by the corresponding CD triphones. The models are combined in the following way: syllable models are taken from the baseline syllable and triphones from the baseline triphone models, then they are combined and re-estimated over the entire training data. Further analysis showed that the top most frequent 200 monosyllabic words covered about 71% of the total number of words in the training set; additionally, 82% of the recognition errors are on monosyllabic words. A new system was trained taking those findings in consideration; homophone words are modeled separately in function of the average duration to make the models with a different number of states. The results for the SWB test set are presented in Table 2.14.

System	Word Error rate [%]
CI monophones	62.3
Word-internal triphones	49.8
800 syllables + 42 monophones	56.4
800 syllables + triphones	51.7
632 syllables + 200 monosyllabic words + triphones	49.3

Table 2.14: Summary of system performance.

As the authors expected, syllable performance level laid between that of triphones and monophones. Analysis of the errors in function of the type of word (monosyllabic or not) showed that it is possible to gain in performance by modeling separately monosyllabic words (word-specific models), outperforming the triphone system. It is worth noticing that the syllable system had lower complexity, even without state-tying.

Another evaluation was performed on the OGI `Alphadigit` corpus [Cole 1997]. Contrarily to the SWB corpus, the vocabulary is quite small (letter and digit strings) and the speech is not spontaneous, but read. This corpus was chosen due to the less significant co-articulation effects (isolated words) and pronunciation variations, a minimal performance difference between triphones and syllables can then be expected. Three systems were trained: cross-word triphones (one part of the context for a given phoneme is in another word), word-internal triphones and CI syllables. The results are summarized in Table 2.15 and in Table 2.16, the results are presented by word category, each category being rather challenging to speech recognition; The E-set is (B, C, D, E, G, T, P, V, Z, THREE) and the A-set is (A, J, K, H, EIGHT).

System	total WER	Alphabet WER	Digit WER
Cross-word triphones	13.3%	16.5%	5.4%
Word-internal triphones	15.2%	19.4%	5.1%
Syllables	10.4%	12.1%	6.3%

Table 2.15: Results for the Alphadigit corpus.

Confusion set	Triphone WER	Syllable WER
E-set	22.2%	18.5%
A-set	16.7%	10.0%
Nasals	11.1%	14.3%
S-F pair	19.4%	17.2%

Table 2.16: Error analysis by confusion category.

The syllables perform better for the alphabet recognition; syllables outperform triphones in the E, A-sets and in S-F pairs. It was expected that triphones would perform much better for those kinds of confusions, given their fine-grain phonetic context. The word “SIX” responds for the most errors in digit recognition, mainly due to confusions in the fricatives caused by the reduced bandwidth of telephone speech.

2.10 Discussion and Conclusion

There was no large gain in the use of long units with respect to the use of triphones in speech recognition. It was expected that longer units would better model the contextual variations and thus provide better performance. Some possible explanations are: the duration model (if even used) was too simplistic to account for the longer models, and the increased number of

states resulted into underestimated models. The methods used to select the units were mainly statistical and some employed an acoustical validation of the units. There was no use of expert phonetic knowledge to select the boundaries of the contextual influence. One possible way to use phonetic knowledge to glue together phonemes into longer units is to rely on the articulatory motions (tongue, jaws, lips, etc) to decide where to split the phoneme sequence into units. Boundaries can be set wherever the position of the articulator is stable or when the articulators are needed to produce a particular sound; the definition of the contexts have to be adapted accordingly, paying attention to the definition at word boundaries. In one of the studies (cf. Section 2.1), the context-freezing units were proposed following a similar idea, but there was nothing done to enforce the division at the regions of lowest contextual influence.

We can identify three ways to define the units based on the type of data that are used:

- *Acoustical*: acoustical data define the "units" which are later mapped into phonemes, being represented by the works reviewed in Section 2.3 (stationary and transition units) and Section 2.4 (syllable like units);

- *Strings*: units are defined based on the strings (graphemes or phonemes) as seen in Section 2.5 (macro demi-syllables), Section 2.8 (weighted mutual information), and Section 2.2 (non-uniform units); and eventually are validated by acoustical evidence (likelihood) as in Section 2.2;

- *a priori*: units are defined beforehand and the instances that have low frequency of occurrence are removed from the ensemble as described in Section 2.1 (hierarchy of units), Section 2.7 (phone pairs), and Section 2.9 (syllables and monosyllabic words).

The work presented in Section 2.6 (joint multigrams) lies mid-way between the approaches that use either the acoustical or the symbolic information because it uses both at the same time to optimize the choice of units to be modeled.

The dependency on the vocabulary is also common in most of the approaches, particularly the one in Section 2.2 which create units that are "tuned" to a specific task. The goal is to maximize performance, but we would like to create models that are the most flexible as possible and not task-specific and non re-usable models (if we change the task of interest). But we are ultimately limited by the content of the training data. We should limit that dependency by not trusting blindly on frequent units because this might be an artifact on the training data and not a valid characteristic of the language.

The weakest aspect of the works that were reviewed in this chapter is that the units are context-independent (exception for the work in Section 2.2, but the number of context-dependent units was too small); CD units were not be used to avoid estimation problems with reduced training data or to circumvent modeling complexities like state tying that would imply in more parameter tuning.

There are some interesting characteristics in the reviewed works that we like to emphasize. First, the acoustical validation (even if likelihood maximization is not linked to error minimization) of the units because it may help eliminate long units that do not provide better modeling that phone-sized units (cf. Section 2.2); second, training on the segments that cause errors (cf. Section 2.3) seems to be a sensible approach, as those are the segments that really provide "new information" and therefore are likely to improve the models. We do not use these ideas, but we think that they are worth to be taken into account in future work.

In this thesis, two procedures to derive long units from the phonetic transcription of words (or phrases) are used; one is a simple syllabification procedure and the other is one using mutual information, as described in Section 2.8; we will try to avoid being too dependent on the training vocabulary by setting thresholds for the selection of units that are modeled, specially for the mutual information approach.

The models of the resulting long units are context-dependent, as we believe that this kind of modeling cannot be discarded for high performance recognition. Moreover, the units are (context-dependent) "cross-word", in the sense that at word boundaries the first (last) phoneme of the following (preceding) word is taken into account in the context. Context-dependent non cross-word units have a generic context that is used at the boundaries, reducing the complexity of the models but losing acoustical resolution. Methods to limit the complexity of the resulting cross-word, context-dependent models are presented and evaluated; the next chapter describes the long units studied in this work.

Chapter 3

Long Units

3.1 Introduction

One way to deal with the contextual variation (cf. Chapter 2) is to adopt units that are longer than phonemes (or long units - LUs) – which range from phonemes to whole words – and are expected to be better than usual phoneme-sized units to cope with the effects of co-articulation. These units should be more adequate to absorb the co-articulation effects because the long term dependencies between the observation vectors are captured directly by the longer models. Phoneme-sized units are just a few observations wide, thus they cannot capture those dependencies which span a longer term [Ganapathiraju 2001]. Another advantage of LUs is that errors at intra-unit level (phoneme substitution or deletion) are naturally absorbed by the model and syllable substitutions are much less likely to occur. In [Jurafsky 2001] an analysis of what kinds of variation are harder for triphones to model is presented and the most difficult variation was found to be syllable deletion; with syllable-based pronunciation this kind of variation is easily dealt with pronunciation variants and it is probable that there are fewer alternatives than with phoneme-based pronunciations.

Two kinds of LUs are studied in this work: the most obvious LU – syllables – and automatically derived multi-phoneme units; these units are the result of a transformation of a sequence of phonemes, which in turn is the output of a grapheme-to-phoneme module. Those units are described in Sections 3.2 and 3.3. It is still desirable that the LUs are context-dependent, to deal with residual co-articulation effects across units. The number of contextual LUs is much larger than for phoneme-based units, and consequently the number of parameters to be estimated is huge. A method to deal with this problem is presented in Section **??**. The topology chosen

for the syllable (and multi-phone units) models is based on experimental results described in Section 3.4. As the number of syllables is very large, it may be the case that some units not seen during training are needed in a given recognition task. One way to perform a back-off to the available (seen) units is presented in Section 3.2.2; the problem of unseen units affects only the syllables; the multi-phone units are obtained by a sequence of merges of pairs of units (starting from phonemes) observed in training data, therefore by repeating that same sequence for the units needed during recognition, we will always end with units that exist in the training data. This chapter ends with the experimental results on the previously presented tasks (cf. Section 1.4), followed by a summary.

3.2 Syllables

Syllables are the most obvious kind of long units (LUs). Their use in speech recognition was proposed about 30 years ago [Fujimura 1975] and recently it has drawn again the attention of the research community. Word boundaries limit syllable boundaries, i.e. there are no cross-word syllables in the lexicon, as this would complicate both decoding and system design.

Some languages like Japanese or Chinese have an intrinsic syllabic nature; moreover the total number of syllables in Japanese is quite low, just a few tenths and in Chinese there are about 1300 syllables plus tones. Other languages such as English have a complicated syllabic pattern and it is not always easy to syllabify a word, due to ambisyllabic consonants: the consonant at syllable boundaries belongs in part to both syllables (e.g. the "b" in *able*). In French when a word ending with a silent (but written) consonant (e.g. /t/, /r/, /s/, /n/) is followed by a word starting with a vowel, the consonant is often pronounced, this is the so called *liaison*; thus there is no clear boundary between adjacent words. Both in English and French, there is a large number of possible syllables (over 8000).

A syllable is generally composed of three parts: the onset, the nucleus and the coda; the nucleus is the only obligatory part of a syllable and it carries peak energy in the syllable. The nucleus is a vowel (or a diphthong), but some languages allow consonants as /l/ or /r/ in the nuclei. The onset is the part that comes before the nucleus and the coda is the part after the nucleus; both onset and coda are consonantal.

An analysis of a manually annotated corpus of spontaneous speech resulted in the following observations about the pronunciation variations of syllables (from [Greenberg 1999], page 163:

"The onset portion of the syllable is generally a "survivor", maintaining its canonical identity regardless of speaking conditions, while the nucleus is a "chameleon", capable of assuming a wide range of vocalic appearances. And the coda often gets no respect, as a consequence of its disposable quality." That study concludes that pronunciation variations at syllable level are more systematic than at phoneme level.

It is expected that even when the canonical pronunciations for the syllables are used, the models will absorb automatically the deviations that appear in current speech (like nucleus substitution and coda deletion); so there is no need to explicitly add the variations to the pronunciation lexicon. This behavior is reflected in the topology chosen for the syllables (cf. Section 3.4).

3.2.1 Conversion from Phonemes to Syllables

There are many methods for finding syllables in the acoustic signal ([Pfitzinger 1996], [Zhong 1999], [Noetzel 1991]); in this work, only the conversion of a sequence of phonemes into a sequence of syllables is of interest; thus automatic methods which are based only on acoustics will not be further evaluated. Moreover, as slight different syllabifications are believed to have minor impact on performance, a simple method based on the theory described in [Grammont 1965] is adopted in this work.

The method goes as follows: a value representing the "opening degree" [1] of the mouth is assigned to each phoneme; the degree is maximal for vowels and minimal for stop consonants. From the observation that a syllable ends with a decreasing degree and starts with an increasing degree, syllable boundaries can be assigned to the "valleys" – $\vee$ ("peaks" are naturally denoted by $\wedge$) in the trace of opening degree as illustrated below for one pronunciation of the French word *quatre* (four).

phonemes	k	a	t	r	ə
opening degree	0	6	0	3	8
trace	/	$\wedge$	$\vee$	/	/
syllables	k|a		t|r|ə		

In Table 3.1 the values of opening degrees are presented; note that they constitute a broad-class partition of the phonemes.

The average number of phonemes per syllable is 2.7 and there are 1600 different syllables

[1] "One opens his mouth to speak and after he has spoken, he then closes his mouth to shut up" [Grammont 1965].

Class	Opening degree
Occlusive: [p, t, k, b, d, g]	0
Nasal: [m, n, ɲ, ŋ]	1
Aspirant: [f, v, s, z, ʃ, ʒ]	2
Liquid: [l, r]	3
Semi-vowel: [j, w, ɥ]	4
Vowel1: [u, o]	5
Vowel2: [a, ɛ, ɔ, ɑ̃, ɛ̃, ɔ̃]	6
Vowel3: [e, i]	7
Vowel4: [y, ə, œ, ø, œ̃]	8

Table 3.1: Opening degree values for the phonemes (in French).

in the training data. Syllables occurring less than 70 times in the training data are removed, resulting in 1200 different syllables to model.

3.2.2 Back-off for Unseen Syllables: Average Occurrence

The number of syllables in a language is very large. It is thus impractical to train models for every syllable given the limited amount of training data; even with increasing amounts of data, it is not likely that total coverage of all syllables will be reached. Therefore it is necessary to deal with the problem of unseen units, i.e. syllables that are needed for a particular recognition task but that were not available (no training data) during model training.

To provide a model for unseen syllables it is possible to *back-off* to a combination of smaller units for which a model is available; the extreme case is to replace the syllable by the concatenation of the models of the phonemes that constitute that unit. Consider a syllable that is composed of M phonemes; there are then 2^{M-1} different ways to decompose this syllable into smaller units (syllables or phonemes). To choose which alternative decomposition is the most adequate, a figure of merit that is called "average occurrence" is proposed in this work. The average occurrence is simply the average number of occurrences of the units in the decomposition, weighted by the number of phonemes that compose each unit. This measure tries to bias the choice towards the alternative decomposition with the highest number of longer (with many phonemes) syllables; it is also reasonable to think that units that receive more data have better estimated parameters.

Suppose an unseen syllable U and a decomposition into K smaller units $U = S_1.S_2.\cdots.S_K$; let the respective number of occurrences for the units in the decomposition be $n_1, n_2, \cdots, n_K$

and let the number of phonemes (length) of each unit be denoted by $l_1, l_2, \cdots, l_K$. The average occurrence $\bar{N}$ of this decomposition is given by equation 3.1. The decomposition with maximum value of $\bar{N}$ is retained and used to provide a model for the unseen syllable.

$$\bar{N} = \frac{\sum_{k=1}^{K} l_k n_k}{\sum_{k=1}^{K} l_k} \tag{3.1}$$

As phonemes have higher occurrence counters than the syllables, the actual value of phoneme counters is replaced by a fixed value. This bias the choice towards an alternative containing more syllables; best experimental results are obtained with a threshold of 200.

To illustrate the choice of an alternative decomposition, all the possible decompositions of the syllable z|w|a|r with the corresponding values of $\bar{N}$ are presented in Table 3.2; the alternatives with $\bar{N} = -1$ have unseen units and are not considered as valid.

Alternative	$\bar{N}$
z . w . a . r	200
z . w . a|r	1529
z . w|a . r	903.5
z . w|a|r, z|w|a . r, z|w . a|r, z|w . a . r	-1

Table 3.2: Alternative decompositions of the syllable z|w|a|r.

3.3 Multi-phone Units

Previous studies on multi-phone units include phone pairs [O'Neill 1998] and macro demi-syllables [Pfau 1997]; the latter are constructed automatically from training data and in both cases the outer states are context-independent. The approach chosen here to build multi-phone units is a variation of that proposed in [Bazzi 2001] for modeling out-of-vocabulary words.

The procedure starts with individual phonemes as units; then at each iteration, the pair of units with highest value of weighted mutual information (MI_w) that satisfies some constraints is merged and promoted to a new unit. The constraints are: the value of MI_w should be above a minimum acceptance value, the number of phonemes in the unit cannot be larger than a maximum threshold and the number of occurrences should be above a minimum number. The procedure iterates until no pair satisfies the constraints or if a maximum number of iterations is reached. Contrarily to [Bazzi 2001], only one pair is promoted per iteration. The value of MI_w between two units u_1 and u_2 is given by equation 3.2.

$$MI_w = p(u_1, u_2) \times log\left(\frac{p(u_1, u_2)}{p(u_1)p(u_2)}\right) \tag{3.2}$$

Where $p(u_1, u_2)$ and $p(u_1)$ are the occurrence frequency of the unit pair and of an individual unit, respectively. When multiple occurrences of the same sentence are present in the training data, there are two alternatives: one is to keep only one occurrence of each different pronunciation; this may avoid biasing the calculations (due to words that are repeated frequently in the training data) and also reduces the computational burden and the other is to keep all occurrences; both alternatives are evaluated.

We explore the possible values for the constraints to evaluate their effect on the generated units; the constraints that are examined are the minimum value of the mutual information MI_{min}, the maximum length of the units (in phonemes) $\#\text{Phones}_{max}$ and the minimum number of occurrences (in the text data) $\#\text{Occ}_{min}$.

3.4 Topology for LU and Context Factorization

In this section we present the experiments performed to chose an appropriate topology for the long units models (emission at the transtions) and a method to deal with the huge number of parameters that are present in context-dependent models for syllables and multi-phone units.

Two different strategies are used to define the topology for the long unit models. One is based on the phonemes that constitute the LU and the other is based on a previous segmentation obtained with existing allophone models to match the "average" duration of the units. To avoid estimation problems with a small training database (`Allophones`) and also to reduce the estimation time, we train only context-independent models. Models are evaluated on the task of recognizing telephone numbers (`TelNum`), which is not part of the training data.

In the first strategy (identified as *LU-CI-01*), models with the topology described for phone-like units (cf. Figure 1.4) are "glued", and there are as many models as the number of phonemes in the (long) unit. When gluing models, the exit and entry states between two phone models are removed and the rightmost internal state is connected to the leftmost internal state of the next phone model via two transitions ("**ft12**" and "**ftx12**"), as depicted in Figure 3.1 in the case of a LU with two phones (for instance "k|a"). We use two transitions to provide alternative paths when the central state is skipped, for example.

Some variations are evaluated:

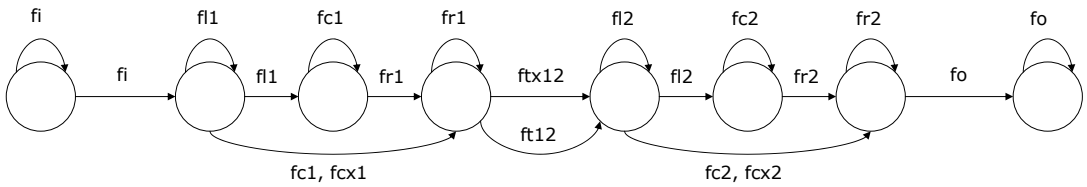


Figure 3.1: First type of topology used for context-independent LUs.

- *LU-CI-02*: remove the extra skip transition (**fcx**) and use only one transition between models;
- *LU-CI-03*: replace the double transition (strategy *LU-CI-01*) by a single one;
- *LU-CI-04*: use only "**fcx**" in the skip; and
- *LU-CI-05*: use only "**fcx**" in the skip and only a single transition (no "**ftx**") between models.

The second alternative topology is based on the phonetic labels obtained using forced Viterbi decoding of the training data with high performance allophonic models (multi-gaussian, context-dependent models). The minimum (F_{min}) and the average number of observation vectors ($\bar{F}$) associated to each long unit (LUs are detected in the phoneme stream) are recorded and the number of states N_S (two states are reserved for context and are not present in the equation) is given by :

$$N_S = \begin{cases} 1 + \frac{\bar{F}}{2} & \text{if } \frac{\bar{F}}{2} > F_{min} \\ 1 + F_{min} & \text{otherwise} \end{cases}$$

Some alternatives are examined; in the first approach, the state skip transitions are removed; the resulting topology for a LU with $N_S = 6$ is depicted in Figure 3.2, (*LU-CI-06*)

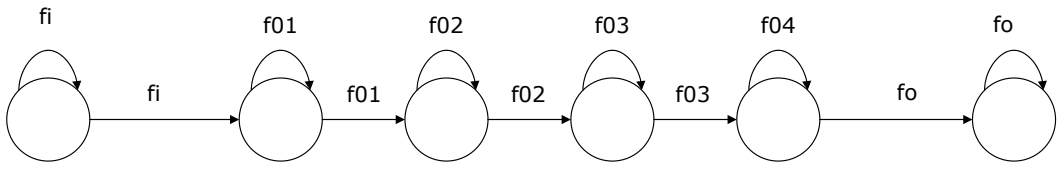


Figure 3.2: Topology used for a context-independent LU with 6 states.

As a second approach (*LU-CI-07*), skips are included to increase flexibility; the topology is as shown in Figure 3.3. The skip transitions ("**fjxx**", where j stands for *jump*) do not share

Gaussian parameters with the other transitions, as in the "model gluing" approach; the goal is to avoid breaking the "trajectory" of the observations.

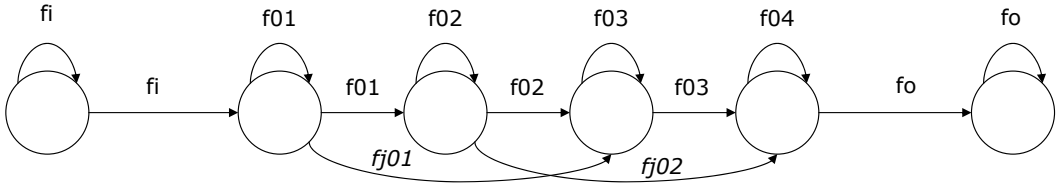


Figure 3.3: Topology used for a context-independent LU with skip transitions.

The next approach removes the skip transitions and adds two transitions (*LU-CI-08*) and finally, both skip and double transitions are retained (*LU-CI-09*).

Instead of dividing the average duration by 2, in the experiment (*LU-CI-10*) we divide it by 3; this reduces the number of states and increases the "looping" factor (probability of remaining in the same state). The topology is as in experiment (*LU-CI-08*) just because it gives better performance than the other experiments in its class.

Model parameters are initialized with the forced-alignment obtained with allophone models, then some Viterbi-training iterations are performed to refine the parameters. Phoneme and LU models are combined in the pronunciation variants in different schemes (due to 28 unseen LUs caused by pronunciation variants); we do not present the results for the preliminary evaluations, but the overall best is scheme 1:

- 1: use only phone models from the allophonic training;
- 2: if there are phone models from the LU training (there are LUs with just one phoneme), use them;
- 3: same as 1, but allow a phonemic transcription in parallel with the LU; and
- 4: similar as 2, but allow a phonemic transcription in parallel with the LU. The phones coming from the LU training are used (when needed) only for the LU transcription.

The results, in terms Word Error Rate (WER), are summarized in Table 3.3 as a function of the resulting number of Gaussian functions in the compiled model for combination scheme 1; results for CI and CD models are also presented for comparison. Results for 4 Gaussian mixture CI phoneme models are also presented because the total number of Gaussian functions is similar and the comparison seems "fair" this way; note that LU models are mono Gaussian.

Experiment	# Gauss	% WER	Observations
CI-phones	240	27.21	Phone baseline
CI-phones 4g	960	18.92	CI phones, larger number of parameters
CD-phones	644	15.17	CD phones, note influence of context
Phone gluing strategy			
LU-CI-01	816	18.19	internal skip: **fc**, **fcx**; transition: **ft** and **ftx**
LU-CI-02	646	20.97	internal skip: **fc**; transition: **ft**
LU-CI-03	752	18.49	internal skip: **fc**, **fcx**; transition: **ft**
LU-CI-04	816	18.62	internal skip: **fcx**; transition: **ft** and **ftx**
LU-CI-05	752	19.00	internal skip: **fcx**; transition: **ft**
Duration-based strategy			
LU-CI-06	534	22.46	internal skip: none
LU-CI-07	720	20.40	internal skip: yes
LU-CI-08	762	19.83	internal skip: none, double transition between states
LU-CI-09	947	19.83	internal skip: yes, double transition between states
LU-CI-10	588	19.90	increased looping factor

Table 3.3: Recognition results for the different topologies.

The reference result for the topologies based on model gluing is given by experiment *LU-CI-01*; comparison of the performance of experiments *LU-CI-02* and *LU-CI-03* shows that the extra transition used in the internal skip is useful. Comparison of the result for *LU-CI-01* against the one for *LU-CI-04* indicates that giving an alternative path through **fc** increases the performance just a bit, and the comparison *LU-CI-03* vs. *LU-CI-04* confirms that it is better to offer two paths for the skip path; from a computational cost point of view, there is no overhead because **fc** is computed in both cases. As it would be expected, the extra transition (**ftx**) increases performance, as shown by a comparison between experiments *LU-CI-04* and *LU-CI-05*. The topology of experiment *LU-CI-03* is adopted for the subsequent experiments with LUs because of its balance between performance and number of parameters and also because it is the one that is closer in terms of performance to the reference models (*LU-CI-01*). Results with CI long units and CD phone models show that it is very important to have detailed contextual models; the results are better for CD phone models even if the number of Gaussian densities in total is smaller compared to the LUs models, indicating that our syllable models should indeed be context-dependent.

Test data is also further decoded with an alternative pronunciation strategy: the pronunciation for each word consisted in a phoneme based transcription in parallel with a LU-based transcription. Analysis of the results showed that when there is an unit (LU) that was frequently seen during training, the decoding hypothesis containing that unit is preferred over the

phoneme-based alternative; as performance increases in the "double" pronunciation scheme, it can be concluded that LUs are useful for modeling and that the errors made by the two models are independent.

In almost all systems that appeared in the literature review (cf. Chapter 2), long units were modeled independently on the context; there was just one study with context-dependent units. This kind of restriction has to be alleviated to have a high performance syllable-based speech recognition system, as already pointed out in [Gauvain 1986].

It is evident that the use of context-dependent syllable models will lead to a huge number of parameters to estimate and an impractical amount of training data would be required for proper estimation of parameters. The most common approaches to increase reliability and robustness of parameters are parameter tying and clustering. In short, parameters that can be shared between different units are identified (at some degree of granularity, e.g. state, phoneme, distribution) and thereafter are estimated with the same data.

To circumvent the problem of the large number of parameters and to keep the spirit of the original system, an approach called "context factorization" is developed in [Messina 2004a]. One assumption is that the contextual influence is maximum for the immediate neighboring phonemes (as for triphones). Consider the topology illustrated in Figure 3.4; the states inside the boxes (dashed contour) are the contextual (connecting) states. Those contextual states are likely to be modeling the most of the transitional variation due to contextual influences and it is reasonable to assume that the immediate following internal states just model a residual part of those influences.

The idea behind this approach is to share the contextual parts of the long unit in function of the context of the first and last phonemes in the long unit; this is equivalent to a classification of the units such that the class identity is given by the first (last) phoneme in the unit when considering the left (right) context. This is conceptually equivalent to dividing the unit into three parts, a left-side part that accounts for the left context of the first phoneme, a central part that is context-independent, and finally a right-side part that accounts for the right context of the last phoneme. For example, the long unit "k|ə" is then factorized as:

$$\text{k|ə} = \text{k\_l} \ . \ \text{k|ə\_c} \ . \ \text{ə\_r}$$

Units that have the same first phoneme share the same left-context dependent model; for example, phoneme "k" and long unit "k|ə" will share the same left-context model "k_l", as well as any other unit that starts with a "k". The same idea applies for the last phoneme; all units

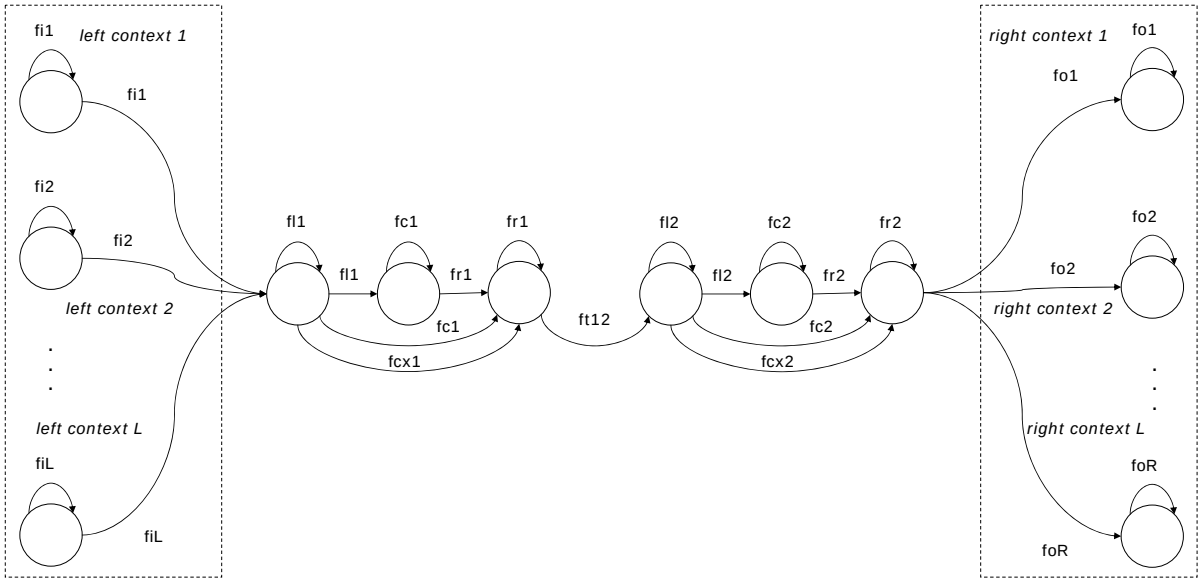


Figure 3.4: Possible topology used for a long unit.

having the same last phoneme (e.g. /ə/) share the same right-context model, e.g. "ə_r". Note that this is different than making the unit dependent on the last (first) phoneme of the preceding (following) unit, as the parameters modeling the contextual influences are, here, shared by many units; otherwise each unit would have its own set of contextual parameters. The parameters of the contextual models are thus trained with large amounts of data, increasing the robustness of the estimation; note that the amount of data is smaller than for CD phone models because some segments are assigned to the internal states of the units and not anymore to the contextual parts, see Section 3.6 for further comments. Only the central parts are specific to each unit, either a phoneme or a long unit; these long unit internal parts become context-independent but receive a fair amount of data, as if we decide to model a given unit, we are certain that there are enough data for training a model for this unit.

Topologies for the contextual parts such as "k_l" and "k_r" are illustrated in figures 3.5 and 3.6 respectively; the number of contextual states depends on the number of contexts for the corresponding phoneme and the shaded state in the figures is non emitting. More complex topologies could be employed, such as one that allows multiple skips into the internal states of the context-independent part or one that have different paths through the "context-independent" part, the multiple paths make the central part "partially dependent" on the context; the topologies for the contextual parts have to be adapted accordingly.

The context length could be longer, but this would lead to another explosion in the num-

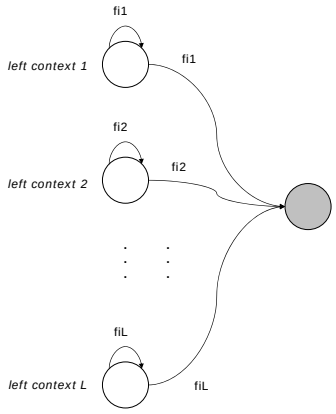


Figure 3.5: Topology used for left context parts.

ber of parameters; this problem can be circumvented with the usual techniques used to train pentaphones or other longer CD phoneme-based units. But as the longer units consume a large number of frames that otherwise would have been assigned to the CD states, we believe that enlarging the context would make parameter estimation less reliable.

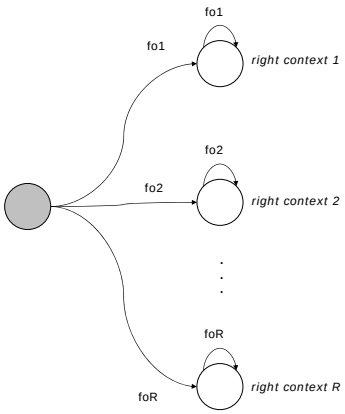


Figure 3.6: Topology used for right context parts.

Training of parameters can be performed with the usual training methods: Baum-Welch or Viterbi (we use the latter). Note that it is also possible to further use some form of tying (i.e. clustering of Gaussian functions and/or densities) after training the parameters. Parameters can be either initialized from the alignments or from trained models, by copying and averaging the parameters of the densities from the corresponding contextual states from the CD phone models.

3.5 Experimental Results

The same topology scheme (*LU-CI-03*) is employed to train models for both syllable and multi-phone units. Initial mono-Gaussian context-independent models are bootstrapped from the forced alignments obtained with multi-Gaussian context-dependent phones; then the parameters are re-estimated with some iterations of Viterbi training. Parameters of mono-Gaussian context-dependent models are then initialized from the CI models. Context-factorization is used for both context-independent and context-dependent models, the difference is just in the number of states in the contextual parts: there is only one state in the CI version of the topology. Mono-Gaussian CD LU models are trained with some iterations of Viterbi training and the alignments obtained at the last iteration are stored and used in the training of Gaussian mixture models, both with the usual Gaussian splitting and with the sequential clustering algorithm (cf. Section 1.5.2).

Results for syllables and multi-phone units are presented below. In Table 3.4 the results for baseline CD phone models are presented with the results for CI and CD syllables (mixtures with a maximum of 4 Gaussian densities per mixture). Units that appear less than 70 times (value obtained experimentally) are removed from the set of valid long units, resulting into 1400 syllables; in all experiments the phoneme counters are set to 200 (cf. Section 3.2.2) when back-off of a unit is necessary.

Models	Task			
	Digits	Numbers	TelNum	Cities
CD-phones	1.58	3.75	10.01	46.93
CI-syllables	1.14	5.17	15.19	53.30
CD-syllables	0.97	3.66	12.02	47.89

Table 3.4: Recognition results (%WER) for CD syllables.

We do not have a clear-cut explanation for the worse performance of syllable models but we can propose some hypothesis; "mixing" the parameters from CD phone models and syllable models may result into state sequences that do not really "glue" well with each other; but as the difference between the forced alignments obtained with syllables and phones is small (on average) this hypothesis looks less likely because the estimated parameters should be similar and the models should therefore match. Another hypothesis that was already proposed [Ganapathiraju 2001] is the different duration of the models (syllables vs. phones) that bothers their mixing. We could also say that the number of parameters is too large, so their estimation is not robust; but this does not seem to be the case as phone models with many components (similar total number of

parameters) are estimated using the same data without much problem.

Different experiments are performed with multi-phone units; we varied the parameters of the unit selection algorithm, and we also consider to retain all or only one occurrence of each different pronunciation (cf. Section 3.3). The values are given in Table 3.5 along with the different experiment identifiers used to index the results in the other tables; we also give the resulting number of units and the percentage of these units that are also in the syllable models (single phone models are excluded from both counts). The results for context-independent models are presented in Table 3.6 for models with 4 Gaussian densities per mixture.

ID	Parameter					
	MI_{min}	$\#\text{Phones}_{max}$	$\#\text{Occ}_{min}$	Use all occs.	#Units	%Syllables
MI02	10^{-4}	6	20	No	689	54.72
MI03	10^{-5}	6	50	No	304	67.11
MI04	10^{-5}	6	50	Yes	2828	21.39
MI05	2×10^{-3}	6	50	Yes	1377	31.81
MI06	10^{-4}	6	20	Yes	3127	19.22
MI07	10^{-2}	6	50	No	59	76.27
MI08	10^{-4}	5	70	No	226	73.89
MI09	2×10^{-3}	6	50	No	306	66.99
MI10	2×10^{-3}	6	70	No	229	71.62
MI11	2×10^{-3}	3	70	No	206	81.55
MI12	2×10^{-3}	4	70	No	217	75.12
MI13	10^{-3}	3	10	No	719	67.45
MI14	10^{-3}	2	10	No	230	75.65
MI15	10^{-5}	2	10	No	337	69.44
MI16	10^{-3}	2	70	No	148	79.05

Table 3.5: Different values for the unit selection algorithm and corresponding experiment identifiers.

Where:

- MI_{min}: is the minimal value of weighted mutual information to accept a candidate pair as a new unit;
- $\#\text{Iter}_{max}$: maximum number of iterations (does not have actual influence with the other experimental values for the parameters);
- $\#\text{Phones}_{max}$: limits the number of phonemes in a long unit; and
- $\#\text{Occ}_{min}$: minimum number of occurrences to accept a unit as valid (text data).

We can see that the more units are used, the smaller the percentage of them are present in the syllable models; the experiments with higher percentage of syllables are those with smaller maximum number of phonemes in the units (the average number of phonemes per syllable is 3). The parameter that has stronger control on the number of produced units is $\#\text{Occ}_{min}$, then comes MI_{min} and $\#\text{Phones}_{max}$.

Models	Task			
	Digits	Numbers	TelNum	Cities
CI-syllables	1.14	5.17	15.19	53.30
MI02	1.90	10.22	15.99	n/a
MI03	1.65	6.36	12.56	55.68
MI04	2.07	11.84	18.28	59.31
MI05	1.88	8.43	14.47	55.76
MI06	2.03	12.13	18.60	60.60
MI07	3.34	7.03	14.11	58.38
MI08	1.75	6.43	13.00	58.26
MI09	1.73	6.38	13.86	57.69
MI10	1.96	6.31	12.68	60.59
MI11	3.00	5.67	13.12	61.41
MI12	1.99	5.40	12.45	60.71
MI13	3.10	5.31	13.90	57.40
MI14	2.91	6.10	14.32	59.52
MI15	2.81	5.83	13.58	58.95
MI16	2.83	5.88	13.76	60.63

Table 3.6: Recognition results (%WER) for CI multi-phone units.

Apparently when all occurrences of each pronunciation are considered to calculate the mutual information (cf. Section 3.3), performance degrades as shown by experiments *MI02* vs. *MI06*, *MI03* compared against *MI04* and finally comparing *MI05* and *MI09*; one explanation is that the units converge to the words in the training corpus as the counters tend to be higher and many merges are elected to be promoted to a new unit.

Experiment `MI04` was not further pursued into context-dependent units because the results did not look promising (only the similar `MI06` was continued to CD models to assess the results). Overall the best setup for CI multi-phone units is the one of experiment *MI03*. Table 3.7 presents the word error rates for CD multi-phone units.

The best overall setup (CD results) is *MI08CD*, mainly because of higher performance on `Cities` and `TelNum` .

Experiments that put into evidence the influence of MI_{min} are the triplet *MI03*, *MI07*, and *MI09* and the pair *MI14* against *MI15*. The performance of the pair *MI14* vs. *MI15* changes abruptly going from CI to CD models, indicating that the threshold value should not be very small; but comparison of the performance of the triplet *03*, *07*, *09* does not lead to coherent results, as experiment *MI03* performs better than the others on `Cities` (which is the preferred task to draw conclusions) for both CI and CD models.

To assess the effect of $\#\mathrm{Occ}_{min}$ on the performance we can compare experiments *MI09* and

Models	Task			
	`Digits`	`Numbers`	`TelNum`	`Cities`
CD-phones	1.58	3.75	10.01	46.93
CD-syllables	0.97	3.66	12.02	47.89
MI02CD	2.18	7.59	14.99	46.67
MI03CD	1.54	5.18	11.93	46.69
MI05CD	1.82	6.32	12.56	46.32
MI06CD	1.92	11.29	16.61	55.24
MI07CD	2.13	3.85	10.40	47.40
MI08CD	1.52	4.82	10.35	45.76
MI09CD	1.63	5.23	12.43	48.59
MI10CD	1.54	5.08	10.44	49.59
MI11CD	2.18	3.77	9.79	51.17
MI12CD	1.65	4.13	9.98	50.00
MI13CD	2.43	4.17	11.09	50.70
MI14CD	2.05	3.71	10.67	49.77
MI15CD	2.11	4.06	10.92	54.55
MI16CD	1.90	3.69	10.64	49.74

Table 3.7: Recognition results (%WER) for CD multi-phone units.

MI10 and also *MI14* against *MI16*. The first pair shows that the choice for *MI09* (better on `Cities`) is a bit better for CI models, but for CD models the better choice is *MI10* (better on `TelNum`). The second comparison pair tends in favor of *MI14* for CI models while for CD models there is no significative difference. As the difference between the parameter setting for the first pair is much smaller, it seems that lowering the threshold is better; note that the counter is on text data i.e. on pronunciation transcriptions. (after removal of identical pronunciations at sentence level after forced alignment).

Finally, to analyze the influence of $\#\text{Phones}_{max}$ we compare the results of experiments *MI13* and *MI14* and also of the triplet *MI10*, *MI11*, and *MI12*. The results indicate that for tasks with the same vocabulary between training and evaluation data, the optimal length of the long units tends to be smaller and closer to the average syllable length, i.e. 3 phonemes. For the task with smaller overlap between the vocabularies (`Cities`), the trend is towards longer units.

Our experiments provided a global overview of the performance in function of the different parameter settings for the unit generation algorithm, but the choice of values for assessing the effect of a single parameter did not provide clear-cut results. It may happen that the optimal choice depends on all parameters at the same time, making the optimization harder; due to time limitations, we could not further investigate this hypothesis.

To illustrate the resulting long units (syllables and multi-phone), Table 3.8 presents the

decomposition of some words with the two kinds of units; where "+" means alternative units, "." means concatenation of units and "|" separates the phonemes that constitute a long unit. We can see in this table that some "obvious" syllables like r|i (in *Pa**ris***) or b|ɔ|r (***Bor****deaux*) are not present in the multi-phone models; this is caused by the control parameters of the algorithm and also by the text data that is used. With the same parameters but on a different data set, it is possible that those syllables would be present in the resulting set of units, if the counters for those particular sequences of phonemes are greater on the new data set.

Word	Syllables	Multi-phone (MI08)
sept	s|ɛ|t + s|ɛ|t|ə	s|ɛ|t + s|ɛ|t|ə
Chantilly	ʃ|ɑ̃ . t|i . j|i	ʃ . ɑ̃ . t|i . j . i
Bordeaux	b|ɔ|r . d|o	b . ɔ|r . d|o
Paris	p|a . r|i	p|a . r . i

Table 3.8: Examples of the resulting syllables and multi-phone units.

3.5.1 Models Initialized with the SCA

The Sequential Clustering Algorithm (SCA) introduced in Section 1.5.2 is used to initialize the parameters for multi-Gaussian models; SCA is used only for syllable models in order to assess if this algorithm is also efficient for longer models. Firstly, models with a maximum of 8 Gaussian components per mixture are initialized and then the Gaussian pool is reduced to different target values for the total number of Gaussian functions (pool size); the initial pool size is 128k Gaussian functions which is smaller than the pool obtained with Gaussian splitting (134k Gaussian functions) ; the results are given in Table 3.9 (the values for SCA parameters are: minimum mass = 10 and membership distance = 5). Only initial models are considered in the pool size reduction, meaning that no refined models are used in the Gaussian merging procedure; initial models are then trained with fixed alignments and a few iterations.

It can be observed that SCA resulted in a smaller pool for the initial model (before merging) with comparable performance; for the telephone numbers task, the performance obtained with SCA models is a bit worse, due to the absence of this kind of utterances in the training data; for the models obtained with Gaussian splitting, as more iterations are performed, the parameters for those densities are better smoothed. As expected, performance degrades gracefully with the reduction of the Gaussian pool.

To further investigate the relation between the initial pool size and the performance of

reduced models, a larger model is initialized with the SCA with a maximum of 32 Gaussian mixtures, membership distance of 5 and minimum mass set to 20. This model should provide a better coverage of the regions where the data points lie in the feature space and thus the reduced models would be better than the reduced models obtained from a smaller initial pool. The comparison may not seem fair, as the mixtures can now select one out of 32 Gaussian functions for the score calculation; the idea behind this comparison is that from a pool of same size, it is still possible to choose a larger number of Gaussian functions for the mixtures and obtain better performances with semi-continuous like models. Table 3.10 brings the results for the models derived from a very large initial pool.

The results show that for the same pool size, slightly better performance can be obtained by selecting more Gaussian functions for the mixtures.

3.5.2 Adapted Models

Adaptation is performed with a class-based Maximum Likelihood Linear Regression (MLLR) [Delphin-Poulat 2001], described in Section 1.5.4. In Table 3.11 results obtained with the mass threshold set to $rlm = 25$ are presented; the models that are adapted are those with parameters configuration `MI02` (cf. Table 3.5). The adaptation experiments in Table 3.11 are identified as follows:

- `32g_m20_d5_red37k`: models obtained with the SCA, the maximum number of mixture components is 32, the minimal mass is 20, and the membership distance is 5; the pool size is reduced to 37k Gaussian functions;
- `8g_m20_d5_red37k`: models obtained with the SCA, the maximum number of mixture components is 8, the minimal mass is 20, and the membership distance is 5; the pool size

Models	Pool size	Task			
		`Digits`	`Numbers`	`TelNum`	`Cities`
`CD-phones 8g`	20276	1.33	3.20	9.68	45.47
`CD-syllables 8g`	134656	0.95	3.18	12.82	47.49
`SCA syllables 8g`	128035	0.95	3.23	13.00	
	60000	0.84	3.22	12.99	
	37544	0.97	3.13	12.86	
	27544	1.12	3.23	12.85	
	10000	1.10	3.49	13.27	

Table 3.9: Recognition results (%WER) for CD syllables obtained with the SCA.

Models	Pool size	Task			
		Digits	Numbers	TelNum	Cities
CD-phones 8g	20276	1.33	3.20	9.68	45.47
CD-syllables 4g	79239	1.03	3.74	12.91	48.21
CD-syllables 8g	134656	0.95	3.18	12.82	47.49
SCA syllables 32g	262262	0.59	2.55	12.49	n/a
	134656	0.65	2.58	12.74	
	79239	0.70	2.98	12.91	
	37544	0.70	2.79	12.90	

Table 3.10: Recognition results for CD syllables obtained with the SCA and different Gaussian pool sizes.

is reduced to 37k Gaussian functions;

- 8g_m10_d5_red10k: models obtained with the SCA, the maximum number of mixture components is 8, the minimal mass is 10, and the membership distance is 5; the pool size is reduced to 10k Gaussian functions;

Identifier	%WER
CD-phones - 4g, 11k	35.89
CD-syllables - 4g, 79k	33.84
Multiphones: MI02 - 4g, 50k	32.35
Syllables: 32g_m20_d5_red37k	34.98
Syllables: 8g_m20_d5_red37k	34.51
Syllables: 8g_m10_d5_red10k	32.26

Table 3.11: Recognition results for adapted CD models for the Cities task.

After task-adaptation, CD long unit models outperform CD phoneme models on the Cities task. For the reduced pool models, having a smaller pool results in better performance; this behavior can be explained by the limited amount of adaptation data (about 10h speech+silence), thus smaller models can be adapted more robustly than models with a larger Gaussian pool. When it is known beforehand that models are going to be adapted to a particular task, the SCA together with Gaussian merging seems to be a promising approach.

3.6 Influence of Data Sparsity on Contextual Factorization

When we factorize the long units into three parts, some segments that were previously aligned to the contextual states of the phone models are now aligned with the internal states of the LU. Consider for example the unit “k|a”, the transition between the phones [k] and [a] are not

assigned to the contextual units “k_r” and “a_l”, but to some of the internal states of the central part of the long unit, ‘k|a_c”.

This reduces the amount of data available to the estimation of the contextual units; to evaluate the effect on the error rate at the beginning of each split/train cycle, we copy the density parameters from CD phone models into the contextual parts and then perform some training iterations, but we keep the parameters of the contextual parts fixed, i.e. they are not re-estimated. But this does not affects the convergence of the EM algorithm.

There was no significant gain (or degradation) of the performance, showing that either there are other factors that have more influence on the resulting error rate or that the amount of data after contextual factorization is enough for proper estimation of the parameters. If a contextual unit has no data we use the parameters from the phone models.

3.7 Summary

In this chapter two methods to convert a phoneme stream into a stream consisting of long units were described; the first method converts phoneme strings into syllables and the second one, converts the phoneme sequence into a sequence of multi-phoneme units.

Different topologies for the long unit models where examined; a topology that presented a good compromise between the performance and number of parameters was adopted for the subsequent modeling. It is worth to note that contrarily to the expectations, the best topology is not based on the average duration of the units, but is based on the number of phones that constitute the unit; the most important variation is to enable the possibility to “jump” the part of the topology associated to one of the constituent phones, permitting then to simulate a phone deletion.

In order to model the resulting units in a context-dependent fashion, the technique of contextual factorization was introduced. This technique can be seen as the sharing of parameters between different units that have the same starting and/or ending phoneme. To circumvent the problem of missing units (here only in the case of syllables) a back-off method was proposed; each unseen unit is decomposed into shorter available units and a figure of merit called average occurrence is used to select the most appropriate decomposition.

Each approach has its advantages and drawbacks: syllables provide an easy decomposition of the words, but we are facing the problem of unseen units; multi-phone units do not present

this problem, but the conversion of phonemes into multi-phone units requires the storage of the sequence of merges produced by the algorithm.

Long units yield similar performance as allophones; in the most difficult task – city names – longer models provided better results than usual allophones. The setup that was used for multi-phone units that perform better than allophones for the `Cities` task (*MI08*) does not result in a large number of longer units; thus with a relatively smaller number of parameters than with the other setups, the estimation of the parameters was more robust resulting in better performance.

In another experiment, we created mixture models with the SCA and then merged similar Gaussian densities down to a desired number; the reduced Gaussian pool models were then adapted to the `Cities` task. The models with the reduced Gaussian pool performed as well as models trained with the usual training path (cf. *MI02CD* in Table 3.11). We believe that adapting to the `Cities` task creates a situation where a larger training database is available; this shows that LU can indeed outperform usual phone-like units, given a larger training corpus which enables better parameter estimation for the more detailed LU models; or if data selection techniques are developed to better use the available data.

Chapter 4

Background on Variability Sources and Multi-modeling

This chapter reviews the different sources of variability present in the speech signal and also presents a literature review on multi-modeling.

4.1 Variability Sources

It is important to deal with the variability of the speech signal because when the characteristics of the signal are different from those of the data used to estimate the model parameters, the recognition performance can be severely impaired.

There are many sources of variability present in the speech signal such as speaker gender, speaking rate (fast, normal, slow), accent (native, non-native), transmission channel, and noise level; this list is by no means exhaustive.

Some of the cited variabilities are not intrinsic to speech, but they are present in the speech signal in any practical recognition system. Transmission channel and noise level are examples and it is well known that they affect the performance of the system. Regarding the transmission channel, the bandwidth can be narrow (telephone) or wide (personal computers); there are errors in remote transmission like packet loss in networks, that could (even if only locally) distort the signal.

Differences in bandwidth change the spectrum of the speech signal, which is the usual base for the representation of the characteristics of speech in current recognition systems; thus one system that performs well for wide band speech does not reach the same quality with narrow

band speech and vice-versa.

Noise is another common source of variability in the speech signal; it can be a stationary background noise (fans, engines, etc.) or localized in time like door slams, telephone rings. Only in very controlled situations it can be eliminated from the signal, but in "real-world" applications it can be reduced, either during signal acquisition with a close-talk microphone or with a microphone array, or with noise reduction techniques, but not completely removed. Noise reduction techniques can degrade the performance by distorting the signal. The effect of (additive) noise on the feature vectors is a shifting of the means and a reduction of the variance of the vectors' distribution [J. P. Openshaw 1994]. Another common effect of noise is to cause the alignment of some speech models with background noise, in particular breath noises at the end of speech, that cause many segmentation errors. A survey on speech recognition in noisy conditions is presented in [Gong 1995] and many robustness aspects of speech processing are in [Junqua 1995].

Dealing with accented speech is quite important in many applications, in particular in language learning aids using speech recognition [Witt 1999]. Accented speech is hard to recognize due to many factors [Tomokiyo 2000]; in non-native speech there are many mispronunciations like phoneme insertion (Japanese speakers insert a /u/ between two consonants) or substitution, e.g. one phoneme is replaced by another phoneme of the target language (Vietnamese often replace /b/ and /p/ because they are not phonemes in their language) or replace then by a phoneme of the native language that sounds similar. Moreover, the replacements are not always systematic (even for a single speaker) and due to the extra cognitive load caused by talking in a non-native language, there may be some problems of articulation. Regional accent also shows "mispronunciations" and sometimes a simple lexicon adaption can help [Schultz 1998], but this is not sufficient to deal with all variations.

The difficulty of recognizing fast (or slow) speech lies in the mismatch between the duration that is modeled by HMMs and the one observed in altered speech and also in the differences in the fine structure of speech caused by the speed of articulation. Moreover, the speed may vary even for the same speaker during the utterance of a single phrase; thus it is likely that this problem should be treated locally. For some speakers, the speech is always fast (or slow), and it is easier to treat the problem. Fast speech is more common in spontaneous speech. The problem can be tackled in many ways [Wrede 2001]: by using rate-specific models, by adapting the topology (or duration models), by tuning the pronunciation probabilities and by manipulating

the observation vectors.

One of the issues in using many models is how to define the rate of speech (ROS) and then being able to classify the utterances as slow, normal or fast speech; besides, the definition should allow the ROS to be calculated/estimated on-the-fly for actual speech recognition applications. In [Mirghafori 1996] there is a discussion on the choices made in calculating the ROS. As recognition performance is lower for fast speech, measures of the ROS based on the recognized words (or phoneme sequences) are likely to be wrong; measures based on the signal itself seems to be more appropriate.

Anatomical differences between speakers are another source of variability in speech; male speakers have longer (in average) vocal tract than female speakers, so the resonance frequencies (also called *formants*) are different from speaker to speaker making the average spectrum for males and females quite different. The most common method to deal with this variation is the Vocal Tract Length Normalization [Lee 1996, Zhan 1997] (VTLN); there are many systems [Wegmann 1999, Neti 1997, Gauvain 2003] adopting different models for female and male speakers to cope with the variability.

4.2 Multi-modeling

This section is a brief review of the literature related to multiple models; the references therein provide a survey on the subject; the goal is to present an overview of multi-modeling in speech recognition. Five main approaches are identified :

- Parallel Hidden Markov Models and Bayesian Networks: in [Brugnara 1992], each unit has one ("slave" HMM) model, and there is a "master" HMM which enables the change of the parameters of the slave model in function of its state. The master "modulates" the probability distributions of the slave models, allowing a multiplicity of representations for each unit. Bayesian Networks (under the more general formalism of Graphical Models) provide a framework to explore the "modulation" of the parameters by measurements made on the signal;

- Mixture of HMMs: in [Rabiner 1989, Korkmazskiy 1997], multiple representations (models) per unit are trained on clustered data. The ways in which the different models are used to decode data are not the same in the two cited references, this will be described in the corresponding section;

- Model selection: multiple models are trained for different conditions and at decoding time, the most appropriate model is selected;
- Parallel decoding: the idea is to train many different acoustic models on data coming from different conditions [Matsuda 2004, Utsuro 2002, Barrault 2005]; during decoding all models are used and their output hypothesis are combined to determine the most likely recognition result; and
- Rate-dependent models: this theme is the one that has received the most of the research efforts [Mirghafori 1996, Pfau 1998, Zheng 2000, Wrede 2001, Nanjo 2002]. The main problem is how to improve recognition for fast speech; models that are trained on rate-dependent speech are used either in combination with other rate-dependent models or the rate of speech is detected and the corresponding models are used.

In the next sections, each one of these approaches is reviewed and the chapter ends with a summary of the previous work and the motivations for the present work.

4.2.1 Parallel Hidden Markov Models and Bayesian Networks

In this approach there are no multiple models per unit, but models are allowed to change parameters to fit the "acoustic context" [Brugnara 1991, Brugnara 1992]. The parameters of each unit model (slave models) are conditioned on the state of a "master" HMM and by proper constraint of the weights on the distributions, it is possible to affect only a subset of the probability distributions describing the observations. Many acoustic situations such as (all from [Brugnara 1992]) voicing, stationarity, fast pronunciation, nasalization, different types of noise, manner and place of articulation can be represented by associating one state of the master model to them.

Both models are assumed to be first order Markov (only dependent on the previous time instant); the parameters of the slave model are dependent also on the state of the master model (primed variables in the equation) as shown by the next equation; Figure 4.1 illustrates those models in the case of an ergodic (i.e. fully connected) master and a slave with three left-to-right states.

$$a_{ij} = P(q_t = j | q_{t-1} = i, q'_t = M_k) \tag{4.1}$$

$$b_j(o_t) = P(o_t | q_t = j, q'_t = M_k) \tag{4.2}$$

where q_t denotes a state of the slave model at time t, q' denotes a state of the master model, $i, j = \{S_1, S_2, S_3\}$, $k = \{1, 2\}$; the rest of the symbols follows the same convention as in Chapter 1.

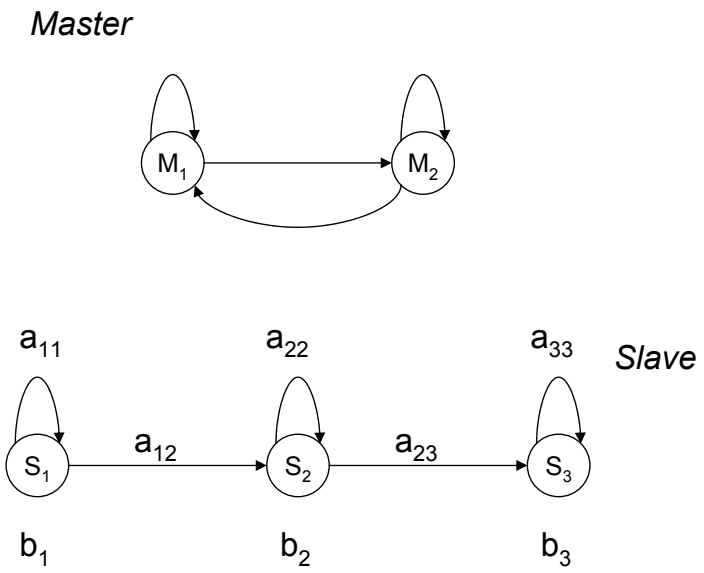


Figure 4.1: Master and slave models.

Now we describe the work on parallel HMMs (PHMMs). Isolated digits [Leonard 1984] were used to evaluate PHMMs; 8 MFCCs, energy and frame duration, from variable frame rate analysis (VFRA) [Zhu 2000, Macias-Guarasa 2003] are used to form the observation vectors. Baseline models are usual HMMs, one for each digit and there are 10 Gaussian components in the mixtures (the duration is also modeled with a Gaussian mixture). There is a single master model (ergodic), with 3 states and its input observations are the frame duration values; there is a slave model per digit with the same topology as baseline models, but there are 9 components.

The percentage of word error rate for all models is presented in Table 4.1. There is a significant increase in performance for PHMMs; the column "KPHMM (VFRA)" is for acoustically tied parameters (details in [Brugnara 1992]). The three states of the master model possibly represent short, medium and long duration; these situations increase (decrease) the probability of staying (skipping) in the states of the slave models. Analysis of the errors showed systematic improvements for diphthongs, supporting the previous hypothesis of duration modeling. The drawback of this approach is the computational burden needed and the multiplication of the number of parameters, as each state in the master model has its set of parameters (in the experiments the mixtures were tied); even if tying is used, it is hard to do it optimally (in some quantitative sense) without first estimating untied models.

baseline	baseline (VFRA)	PHMM (VFRA)	KPHMM (VFRA)
2.86	2.33	1.65	1.37

Table 4.1: Error rates for baseline and parallel HMMs.

In [Hain 1999] a related approach is introduced where the mapping between the phonemes and the models is stochastic. This means that there are possibly many different models for a phoneme (unit) but the actual sequence of models is also hidden, thus the approach is called Hidden Model Sequence (HMS). The most appropriate model is determined locally rather than globally (and *a priori*). Multiple mapping levels can be added between the word sequence and the model sequence but it is assumed that one level only depends on the immediate superior level.

Under this assumption, and considering only a level between the phonemic and model levels, the probability of the observation sequence $\bar{O}$ given the sentence S and the model parameters λ is given by:

$$P(\bar{O}|S,\lambda) = \sum_{M \in \Omega(\bar{R})} P(M|\bar{R},\lambda) \cdot P(\bar{O}|M,\lambda)$$

where $\bar{R}$ is the sequence of phonemes corresponding to the sentence (given by a dictionary), M are the different models associated to the sequence of phonemes $\Omega(\bar{R})$. Two levels can be identified: the first term in the rhs gives the mapping between models and phonological units and the second term is the relationship of the observations and the models. The parameter estimation formulae are given in [Hain 2001].

It is assumed that each phone can produce an *a priori* fixed number of consecutive models, chosen from a subset (predetermined) of all HMMs; thus a N-gram approach can be used for $P(M|\bar{R})$:

$$P(M|\bar{R}) = \prod_{t=1}^{L} P(m_t|m_1 \ldots m_{t-1}, \bar{R})$$

The history was restricted to the immediate left and right neighbors and the dependency on the previous models was also removed, resulting in a HMS counterpart for triphones:

$$P(M|\bar{R}) = \prod_{t=1}^{L} P(m_t|\bar{r}_{t-1}, \bar{r}_t, \bar{r}_{t+1})$$

There are then many models for each phone, and a probability depending on the left and right phone context is assigned to each set of models. For unseen models and contexts, lower order

statistics are used; details of the back-off strategy are given in [Hain 1999].

To reduce the computational cost, results were obtained by re-scoring a lattice of hypothesis given by standard HMMs. There was a significant reduction (relative) in the error rate for the Resource Management [Price 1988] task of 17%, but no improvement on the Switchboard [Godfrey 1992] task. It was proposed that instead of producing a fixed number of models, producing a variable number could better deal with the spontaneous speech effects present in that task; more appropriate back-off (and smoothing) methods should also be investigated in order to better circumvent the problem of unseen and low count units.

Another recent modeling formalism enabling the control of the parameters as a function of the state of the model are Graphical Models (GM) [Jordan 2004, Murphy 2002, Lauritzen 1996]. GM are a powerful formalism that englobe many different classes of statistical models, including HMMs. Speech recognition is modeled under this formalism with the class of directed graphical models, known as *Dynamic Bayesian Networks* [Murphy 2002].

Multi-modeling can be treated under this formalism by making the observation probabilities dependent on various measurement on the speech signal, that would transform the parameters according to the decoding conditions. Proceeding in this way, it is possible to compensate the effects of dividing the signal into frames, e.g. very rapid events like the "explosion" (stop release) of plosive sounds loose their timing information; or we can use a specific (more adequate) feature space for different kinds of sounds (vowels, plosives, etc) to improve the representation instead of relying on the usual MFCCs and their derivatives, which may not be the most appropriate representation for all kinds of sounds. There are many modeling possibilities under the framework of GM; we only evoke here the existence of alternative solutions to acoustic modeling.

4.2.2 Mixture of HMMs

In [Rabiner 1989] multiple models per recognition unit are trained and subsequently used in recognition. The idea is based on the results from template-based system, where performance increases when there are multiple templates per unit (in that case, each word is a unit). The training data for each model (HMM) is clustered with two different methods that are based on the likelihood of the training data. The first method starts with one model per unit and then the model is "split" in two new models; the mean vectors of the mixture components are randomly disturbed and segmental k-means [Rabiner 1986] is used to re-estimate the parameters; then either the model with the largest number of training utterances assigned to it either the

one with greatest total likelihood is split and the process is repeated. In the second method, the training utterances (assigned to each model) having likelihood below a threshold (fixed or relative) are separated and assigned to a new model; the likelihood on the "outliers" samples is increased by this new model.

Models were trained with both methods on the TI connected digits database [Leonard 1984]; the value chosen for the threshold based model clustering was the one that discarded 20% of the lower likelihood training samples. Likelihood clustering (the first method) did not result in any improvements on test data, while threshold clustering yielded an improvement when going from 1 to 2 models per unit; further increase in the number of models did not lead to any improvements. The results are partially reproduced in Table 4.2, where UL stands for unknown digit string length and KL for known length.

Clustering procedure	#models/digit	Training set		Test set	
		UL	KL	UL	KL
Likelihood	1	2.45	1.09	4.02	2.38
	2	1.76	0.71	3.98	2.39
Threshold	1	2.04	1.06	3.59	2.31
	2	1.04	0.37	2.84	1.60
	3	1.07	0.40	2.93	1.61

Table 4.2: Recognition of continuous digits for 2 clustering procedures

As pointed out in [Rabiner 1989], the likelihood clustering method only increases the likelihood of the training data, which is not at all a guarantee to improve performance on test (or independent) data. It was checked that data sparseness was not an issue, so the performance that was obtained is indeed the best that the technique could provide.

We propose another explanation of why increasing from 2 to 3 models per digit for threshold clustering does not increase performance on test data. As outliers are less frequent, there are fewer data to estimate parameters, therefore the estimation is less reliable; moreover, it is less likely that the outliers are in the same region of the feature space in both training and evaluation data. Thus modeling outliers contributes to reduction of the errors in a limited way.

The second paper deals with multiple models per unit based on clustering [Korkmazskiy 1997]; it is hypothesized that HMMs can only characterize the speech signal from an homogeneous source. When the signal becomes heterogeneous, a mixture of HMMs (called generalized mixture of HMMs – GMHMMs) is used to better represent the speech data. Mixing models be obtained by clustering the speech patterns of the same unit

and then training models for each resulting cluster. As speech samples have variable length, traditional clustering techniques cannot be directly employed because there is no distance measure between vectors of different dimensionality. To circumvent that problem, the average likelihood of each HMM state j of unit i is used to create a vector on fixed dimension N_i (the number of states in model i), that represents the location of the speech sample with respect to the model; this measure is directly applicable to mixture models as well, by replacing the likelihood of each state by the likelihood of each component of the mixture associated with the state. This metric is a byproduct of Viterbi alignment of the training data. Then all the speech samples corresponding to each unit are clustered with the k-means algorithm.

Speech data consisted of connected digit strings (English), collected over a wide range of noise conditions (clean to barely audible); strings have from 1 up to 30 digits. Context-dependent sub-word units (274 units) were used and each digit is modeled by a concatenation of three units (head, body and tail); the total number of clusters was 678 (an average of 2.5 models per unit) and 8 mixture components per state were used. Word error rate for traditional HMMs is 2.6% and for GMHMMs is 1.9%. To compensate for the greater number of parameters in GMHMMs, usual HMMs with 20 mixture components were trained with a resulting WER of 2.4%. Discriminative training was also employed, but as this is not of interest to this work, we will omit the details of the training procedure; the WER for HMMs is 1.4% and for GMHMMs it is 1.0% with gradient descent training of the parameters.

But results presented some inconsistency in the discussion about which metric to use: state or mixture; the reported WER for HMMs is 1.9% and not 2.6% as presented in the tabular results (Section 4 of [Korkmazskiy 1997]). If the tabular results are correct, the technique of GMHMMs leads to a relative improvement of 20%.

4.2.3 Model Selection

In [Song 1998] multiple models are trained according to the noise masking level (not the perceptual masking as used in MPEG coding) and during recognition, the noise masking level is estimated and the appropriate model set is chosen. Noisy speech is enhanced with spectral subtraction [Boll 1979] and the portions of the speech spectrum that have energy below the masking level are floored to that level. The choice of the value of the masking level is important because if the level is too low, it is not effective; otherwise if it is too high, the speech signal becomes distorted.

The masking level of the test utterance is estimated from the first 8 frames (it is considered that those frames are non-speech). The energies of the outputs of the MFCC (cf. Section 1.1) filter-bank are used; the mean is regarded as the noise estimate and the standard deviation is used to determine the masking level (after mean removal):

$$\mathrm{ML}_{dB} = 20 \log_{10}(\sigma\gamma)$$

where σ is the standard deviation and γ is a weighting factor; the results showed that performance is significantly dependent on the value of γ. The obtained level is then quantized into one of few levels to reduce memory requirements.

Models were discrete density HMMs (64 codewords) for 50 isolated command and control words for in-car applications; the number of states is three times the number of phonemes in the word. Noisy-conditions were simulated by adding car noise to scaled speech to yield the same SNR as for actual environments. Car noise was recorded at different situations with the given value of average SNR:

- `env1`: car stopped, engine stopped, SNR = 21dB;
- `env2`: car stopped, engine running, SNR = 10dB;
- `env3`: driving in a low traffic road, SNR = 2dB;
- `env4`: driving in a heavy traffic road, SNR = 0dB; and
- `env5`: driving in an express highway, SNR = -5dB.

In Table 4.3 we reproduce some of the recognition results (word accuracy) for clean speech and noise conditions (last column is the average for all noise levels) for baseline models, baseline models with spectral subtraction and noise masking (SS/NM) with the masking values that were used for multiple models (MM), and finally for multiple models (2 levels are used: 69 and 80dB) with the best value of γ.

The results for other values of masking level are worse than those for the selected levels and they did not vary that much. We can see that the recognition rates for the multiple models are a combination between the rates for the individual models for the different noise conditions; as the levels are quite separated, it seems clear that only one of the models is active (the one for $\mathrm{ML}_{dB} = 69$) for noise conditions `env1` and `env2` and the other model, $\mathrm{ML}_{dB} = 69$, for

Models	Noise condition						Avg
	clean	env1	env2	env3	env4	env5	
baseline	96.4	93.0	90.8	76.6	64.6	22.4	74.0
SS/NM $\mathrm{ML}_{dB} = 69$	96.2	95.8	95.6	93.2	90.2	61.8	88.8
SS/NM $\mathrm{ML}_{dB} = 80$	91.8	90.8	90.6	88.6	86.2	74.4	87.1
MM=2 levels $\gamma = 0.3$	96.2	95.8	95.6	92.8	89.6	74.4	90.7

Table 4.3: Recognition results for different noise environments and noise masking.

environment `env5`. For the environments in between, there is a combination of the models leading to mixed results. The largest part of the improvement comes from the noisier environments, numbered 3 and above; the multiple models only improved significantly over the SS/NM setup (with $\mathrm{ML}_{dB} = 69$) in the `env5` condition.

In [Zhang 2004], HMMs for different kinds of noise and values of SNR are hierarchically arranged in a tree structure. Noisy data was produced by adding noise from 28 different sources to clean speech; speech and noise levels are adapted to result into 3 different values of SNR (resulting into 84 conditions). Noisy data is tree-clustered because it was found that data can not be separated by the SNR value only, i.e. samples coming from data with different SNR can be nearer than samples with same nominal value of SNR (samples in the acoustic vector space).

To reduce the computational cost of the tree construction, a Gaussian mixture model (GMM) is created for each condition and trained with the Baum-Welch algorithm. The noise-specific GMMs (64 components) are then clustered based on the likelihood matrix between all pairs of mixture models. There is an HMM for each node in the tree. Models at leaf nodes are trained (actually adapted with MLLR) on a single kind of noise and at a specific value of SNR; models at internal nodes are adapted on a mix of data from the descendant nodes, so they are expected to be robust to noise variations.

During recognition the node (not necessarily a leaf) with maximum likelihood (using the GMMs) to input speech is selected and the corresponding model is used. The results that are most interesting (corresponding to the Test2 setup in [Zhang 2004]) are on "real-world" noisy data; the other test is on artificially added noise, as used during training. The results are in figures, so we are not going to reproduce them here, but there was a relative improvement of 33% over baseline "clean" models with the adapted models; a comparison against models trained with mixed data would be interesting, but it is lacking.

4.2.4 Parallel Decoding

The idea of parallel decoding is to perform decoding with various different models (or different settings) and combine the diverse results. There are many options to combine the recognition hypotheses: in [Fiscus 1997] a voting procedure (ROVER) is used; a machine learning scheme is used in [Utsuro 2004] and in [Markov 2003] a word graph is constructed from the various hypotheses.

In [Matsuda 2004], the word graph combination scheme [Markov 2003] is used to select the output hypotheses of 48 models. The speech is represented with two different front-end streams: the usual MFCC and the differential spectrum based MFCC; it was found that by replacing the speech spectrum by the differential spectrum given by the next equation leads to better performance.

$$D(i,k) = |Y(i,k) - Y(i,k+1)|$$

Where $Y(i,k)$ is the power spectrum for the i-th frame and k is the index of the spectral bin. Then for each stream, there are models for two genders, six values of SNR and for normal and hyper-articulated speech. One hypothesis is chosen for each stream, based on its likelihood, and then they are combined with the word graph method.

The models were trained on (artificial) noisy speech and models for hyper-articulated speech were generated from normal speech models by changing the topology; when a vowel is repeated between two words, i.e. the last sound (vowel) of the first word is the same as the first sound of the second word, an optional silence state is inserted between the words. Models are evaluated for noisy speech (again with combination of clean speech and noise); there are very few utterances for hyper-articulated speech so we will not comment on those results. It was observed that models trained on clean speech perform much better on low-SNR speech than models trained on low-SNR data and evaluated on clean speech. The accuracy of the selected models (per front-end stream) is quite high because of the matched models/data situations (in excess of 80% accuracy). Performance after combining the hypotheses (see [Markov 2003] for the details) from both front-ends is higher or equal than in isolation; this shows that the errors are independent thus the combination increases the accuracy of the system. The drawback of the approach is the excessive computational burden to run 48 decoders in parallel.

The agreement between multiple models is used to improve confidence measures in [Utsuro 2002]. There are many evaluations that are of less concern for the present work,

but points of interest to us are: combination of models of different units (e.g. triphones and syllables) is quite efficient and model combination increases the confidence measure on the recognized words.

Another alternative is presented in [Barrault 2005]; two front-ends are used to train hybrid neural network/HMM [Morgan 1993] models. The hypotheses are combined based on the probability that the hypothesis of one of the front-ends is correct, given the competing hypothesis from the other models; one of the front-ends is known to have better performance thus its hypothesis is used as reference. Tests are performed on Spanish and Italian data from the AURORA3 (portion CH1) corpus [Hirsch 2000]. Please refer to the article for the description of the two used front-ends; we will denote them by m and j.

Three comparison states Q_i are defined to compare the output hypothesis (sequence of words) obtained with each front-end, denoted by W_m and W_j. The states are:

- Q_1: $W_m = W_j$;
- Q_2: the same word or two different words are hypothesized in approximately the same time interval, even if $W_m \neq W_j$;
- Q_3: two segments of W_m and W_j have about the same time bounds but without any word in common.

The word error rate is low in state Q_1; the agreement is 72.66% for Spanish, with a WER of 0.16% and the same hypothesis is generated for 63.16% of sentences for Italian with 2% error rate. WER in state Q_2 is also low, essentially substitutions (not given). It is unlikely that they can be recovered in the case of consensus with scoring methods. The interest is in the state Q_3, where errors can be avoided by a better scoring and decision strategy. The strategy is to maximize the probability of selecting the correct candidate when it is proposed by one of the systems (in this case, the one producing the hypothesis W_m, which is known to provide better recognition).

In the case of a substitution (both systems propose a different word for an interval with overlap larger than 50% of the word time) a decision rule is applied, otherwise the output of system m is accepted. Let $C(w, w_m, w_j)$ represent the fact that w is the correct word, with system hypothesis being w_m and w_j. The rule adopted is:

$$w^* = \underset{w \in (w_m, w_j)}{\operatorname{argmax}} \; P(\sigma | C(w, w_m, w_j)) P(C(w, w_m, w_j))$$

Where σ represents a sequence of labels, one label for each phoneme in the word. The labels are the result of a partition of the posterior probability space of the phonemes given each front-end; this partition represents the zones where the systems agree (or disagree) with different degrees of reliability and it is computed beforehand.

For small vocabularies (as is the case of the CH1 portion of AURORA3), a good approximation for the rule is given by the prior:

$$w^* = \underset{w \in (w_m, w_j)}{\operatorname{argmax}} P(C(w, w_m, w_j))$$

Which is calculated with a different corpus than the one used to estimate the parameters. Using this approximation, the reductions in WER are of 13.67% and 18.25% for Italian and for Spanish, respectively.

In [Meneido 2000] a system using different front-ends is presented; it also employs hybrid models, but the structure of the neural network classifiers is different for each front-end. Instead of running many decoders, the probabilities of each system are combined and a single decoder can be used. Even if there are not many decoders, we think that this approach is more similar to [Barrault 2005] than to the other approaches reviewed here.

The posterior probabilities for each phoneme coming from the different classifiers are combined then passed to the decoder; 38 context-independent phone models are used for the recognition of European Portuguese, with a (large) vocabulary of 27k words. Two and three baseline systems are combined through an average in probability and in log-probability domains; a preliminary analysis of the individual errors showed that they are not strongly correlated.

The posterior probability of phone y given the acoustical observation $\mathbf{x}$ (see [Morgan 1993] for details on hybrid neural networks/HMM systems) as calculated in hybrid systems by an artificial neural network classifier is represented by $P(y|\mathbf{x})$. If there are M classifiers, the average is given by:

$$\hat{P}(y|\mathbf{x}) = \frac{1}{M} \sum_{k=1}^{M} P(y|\mathbf{x}, M_k)$$

In the second combination scheme, the probabilities are multiplied, which correspond to performing an average in the log-probability domain. Now the prior is affected by the averaging

operation (actually, it is raised to the Mth power), so the probabilities have to be scaled:

$$\log\left(\frac{\hat{P}(y|\mathbf{x})}{\hat{P}(y)}\right) = \frac{1}{M}\sum_{k=1}^{M}\log\left(\frac{P_k(y|\mathbf{x})}{P(y)}\right)$$

Combination in the logarithmic domain provided the best results; many systems are combined, so we will omit the details. Comparing the combination of two systems with a larger single system (but the same number of parameters than the combination) resulted in a reduction of the WER of 12%. Combining the larger system with two others reduced the errors by 15%, compared to the large system in isolation. The combination of three simples systems reduced the error, compared to the large system, by 18%.

4.2.5 Rate-dependent Models

Models robust to fast speech are an active area of research [Mirghafori 1996, Pfau 1998, Zheng 2000, Wrede 2001, Nanjo 2002]; not all works that are cited treat this problem with multiple models, but they provide an overview of the subject. We will focus this review on [Zheng 2000] which employs different models for normal (average) and fast speech. In [Wrede 2001] an in-depth analysis of the variabilities is presented on a corpus of read speech, thus it may not be really representative of the effects of fast speech.

In [Zheng 2000] rate-specific parallel models (one for slow the other for fast speech) are used and thus the ROS does not need to be estimated precisely. The rate is allowed to change between words, to reduce the effect of variations during the same sentence. Training data is labeled according to a word-specific ROS measure, that takes into account the distribution of the word duration because words with fewer phones tend to be classified as "fast" even though they are not uttered faster than their normal rate. Using forced alignment of the training data, the following ROS was used:

$$R_W(D) = P_W(d > D) = 1 - \sum_{d=0}^{D} P_W(d)$$

Where W is a given word, D its duration and $P_W(d)$ is the probability of that word having duration d. Thus $R_W(D)$ is the probability of W being longer than D. As $P_W(d)$ is hard to estimate properly, it is assumed that the durations of the phones are independent, therefore the pdf of the duration of a word is the convolution of the pdfs of the durations of its constituent

phones (or other sub-word units). A threshold was chosen to split the training data into two sets, i.e. equal amounts of "slow" and "fast" speech.

In the 1998 NIST Hub-5 [1] evaluation data the baseline system gives 59.8% WER and the rate-dependent models give 57.9%, yielding a relative improvement of 4%; there are other improvements to that system, but they are not reported here as they are very system specific.

4.3 Summary

In all reviewed approaches, there was a mild improvement in the performance of the systems; the approach that yielded the greatest improvement is *Model selection*, but it was not clear from the text how model selection is updated to track varying noise levels. From the results, it seems that we gain the most with models that are adequate to noisy speech; fast speech is an important issue, but it is hard to define how speech is affected and which is the best strategy to circumvent this problem.

This review does not covers other aspects of multiple modeling, such as having many different front-ends (possibly with multiple back-ends) or combination of heterogeneous models (each model being specialized in one aspect of the speech signal). The latter approach seems promising and it will be discussed in Chapter 6.

[1] `http://www.nist.gov/speech/tests/ctr/hub5e_98/current-plan.htm`

Chapter 5

Variability Sources and Multi-Modeling

There are many sources of variability in speech and it is very hard to cope with all the resulting variations. However, in the case where the variation effects can be categorized, one solution is to have multiple models (multi-modeling) that are specialized for each class. A non exhaustive list of examples of classes is: gender, speaking rate (fast, normal, slow), accent (native, non-native), and noise level.

In this work we adopt the multi-model strategy to deal with the variability present in speech, as in *Mixture of HMMs* approach reviewed in Chapter 4. We will concentrate on an *a priori* selection of variability classes, instead of an automatic clustering of training data; we believe that this choice of classes can help to "guide" the estimation of parameters to reduce redundancy and improve the modeling ability. The multiple models have to be "combined" in some way to decode speech from an unknown source of variability. If the source can be reliably detected from the signal, the *Model selection* approach from the previous chapter should be enough, but it may be the case that even if the matched model can be selected, a combined model may still yield higher performance; this can be explained by the hypothesis that a combined model may cover some deficiencies of the specialized model, either because of a more robust parameter estimation or because the combined model can deal with other types of variation that are inherent to the speech signal and cannot be optimally dealt with a (restricted) specialized model.

The following question arises naturally from the approach: what is better, either to have multiple models for each unit, with a few Gaussian functions per mixture or to have a single

model, but with many Gaussian components coming from the various sources? The former seems to involve less computations, as it can be expected that the mismatched models will have low probability and the beam search will rapidly exclude the corresponding hypothesis from the search; putting the models in parallel enables the exploitation of the correlation between the acoustical realizations coming from the same class of variability. The latter approach may result in more computations per active state, but it may happen that less states are active, and as a consequence, the cost is about the same. The issues that have to be solved by the experiments are which classes (conditions) are of interest to model and how models have to be combined.

We evaluate different sources of variability; some of them are well known, such as speaker gender, and they serve as a reference to give an idea of the gains involved; the classes of variability are described in Section 5.1; this section also illustrates the organization of training and test data. Models are combined following three different schemes, each one at one level of representation of speech, as presented in Section 5.3. Section 5.4 gives the experimental results and presents the setup whenever necessary for understanding of a particular experiment.

As data sparsity is a problem with the adopted approach which implies splitting the training data into homogeneous chunks, Section 5.5.1 presents some experiments with model adaptation; the aim is to first train the models on all data and then adapt just on specific data the parameters having a sufficient amount of adaptation data.

5.1 Variability Classes

We choose different classes of variability; their choice reflects a selection of easy to determine values and because we believe that by specific modeling of the variations, recognition performance could be improved.

The training data is split into many chunks, depending on the type of variation that is studied. For instance if gender is used, the data are split into two chunks, labeled “female” and “male”; data could be further split into a third class, namely “child” if enough data are available. Evaluation (test) data is split in the same fashion; this enables us to perform recognition tests on matched (we also denote this as *Oracle* experiment) and mismatched data.

5.1.1 Speaker Gender

Splitting the data between genders is quite common nowadays [Wegmann 1999, Neti 1997, Gauvain 2003]; this class is used not only to measure its impact on recognition, but also to serve as benchmark, due to its widespread adoption by the community. Table 5.1 shows how the training and test data are divided between the genders; PSTN data accounts for about 80% of all data, the remaining 20% are GSM data.

Corpus	PSTN			GSM		
	Female	Male	no tag	Female	Male	no tag
Training	51.55%	48.18%	0.27%	41.22%	46.26%	12.51%
`Digits`	38.88%	61.12%	—	42.34%	57.66%	
`Numbers`	43.73%	56.27%	—	52.79%	47.21%	
`TelNum`	59.16%	40.84%	—	47.74%	52.26%	
`Cities`	50.67%	49.33%	—	—	—	—

Table 5.1: Division of training and test data per gender for all channels.

Tables 5.2 and 5.3 present the recognition results (WER) on telephone numbers (`TelNum`) when we force the recognition with models trained on data of the same and also of a *different* gender than test data; this table illustrates the effects of the differences in both data distributions on recognition results; models have 4 Gaussian components per mixture.

		Models	
		Female	Male
Data	Female	8.05	*16.29*
	Male	*26.93*	10.66

Table 5.2: `TelNum` results with matched and *mismatched* models and data, gender dependent (GD) models and PSTN test data.

		Models	
		Female	Male
Data	Female	13.91	*33.61*
	Male	*21.29*	14.52

Table 5.3: `TelNum` results with matched and *mismatched* models and data, gender dependent (GD) models and GSM test data.

5.1.2 Transmission Channel

Here we simply keep the training corpora separate, i.e. PSTN on one side, GSM on the other; the sizes of training and evaluation corpora are in Section 1.4.1 and Section 1.4.2 respectively. Using the channel as variability class aims at exploiting the known differences in the data from land-line and wireless channels and try to benefit from that knowledge to create better (specific) models.

Table 5.4 presents the WER for the `TelNum` task when models are trained on the same channel as the evaluation data and also when there is a *mismatch*; models have 4 Gaussian components.

		Models	
		PSTN	GSM
Data	PSTN	9.61	*19.19*
	GSM	*15.43*	14.20

Table 5.4: `TelNum` results with matched and *mismatched* models and data, channel dependent (CHD) models and PSTN/GSM test data.

5.1.3 Signal to Noise Ratio – SNR

From the literature review (cf. Chapter 4), the SNR plays an important role in determining the performance of a speech recognition system. Performance is high when the SNR is high, while it degrades with decreasing values of the SNR; the lower performance is due to distortions caused by the higher levels of noise energy, "shifting" the parameters from their location in the feature space. The noise level itself is not a speech-specific source of variability, but any practical recognition system has to live with the adverse effects of noise on performance. Hence, we adopted the estimated value of the SNR as a class of variability.

There are many ways to estimate the SNR from the speech signal [Hirsch 1995, Tchorz 2003, Ephraïm 1984]. In this work we adopt an off-line estimation, based on forced alignments; we proceed in this way for two reasons: one is to try to have a precise estimate of the SNR, the other is to use the actual decoder that is employed during recognition time, in order to make the measurements as in-line as possible with the true usage conditions. The estimate of the SNR is given by:

$$SNR = 10 \log_{10} \left(\frac{\widehat{S+N} - \hat{N}}{\hat{N}} \right)$$

Where $\widehat{S+N}$ is the average energy for the frames marked as speech (actually noisy speech) and $\hat{N}$ is the average energy for the frames marked as silence (leading and trailing) or inter-word pauses.

Tables 5.5 and 5.6 present the WER for the `TelNum` (PSTN and GSM, respectively) task when models (4 Gaussian components) are evaluated in matched conditions, and to illustrate the effects of the mismatch, the table also brings the results when the models are used on data belonging to a different range of SNR.

		Models		
		SNR $< 20dB$	$20dB \leq$ SNR $< 28dB$	SNR $\geq 28dB$
Data	SNR $< 20dB$	11.66	*11.80*	*14.97*
	$20dB \leq$ SNR $< 28dB$	*9.04*	8.04	*9.39*
	SNR $\geq 28dB$	*7.14*	*6.06*	6.40

Table 5.5: `TelNum` results with matched and *mismatched* models and data, SNR dependent (SNRD) models and PSTN test data.

		Models		
		SNR $< 20dB$	$20dB \leq$ SNR $< 28dB$	SNR $\geq 28dB$
Data	SNR $< 20dB$	16.30	*16.77*	*23.91*
	$20dB \leq$ SNR $< 28dB$	*10.28*	8.97	*12.05*
	SNR $\geq 28dB$	*6.40*	*5.78*	5.88

Table 5.6: `TelNum` results with matched and *mismatched* models and data, SNR dependent (SNRD) models and GSM test data.

The results on PSTN data show that in general, matched models give the highest performance; the data with high SNR is "easier", as even severely mismatched models (low SNR) give quite reasonable performance on these data. The models trained on mid-range SNR perform well in all data sets; for this source there seems to be little margin for improvement, as the performance under matched condition is not much higher than for the mismatched models. The behavior on GSM data is similar as for PSTN data.

Figures 5.1 and 5.2 present the results for baseline (channel specific) models with varying number of Gaussian components on the different portions of the test data, arranged by the value of the SNR for PSTN and GSM channels, respectively; the bars are a histogram of the test data according to the measurement (SNR) and the lines are the WER on the data in each bin (values are the center of the bins). The legends are the same for all figures: curves marked with circles, diamonds, squares and triangles represent, respectively, results for models with 1, 2, 4, and 8

Gaussian components.

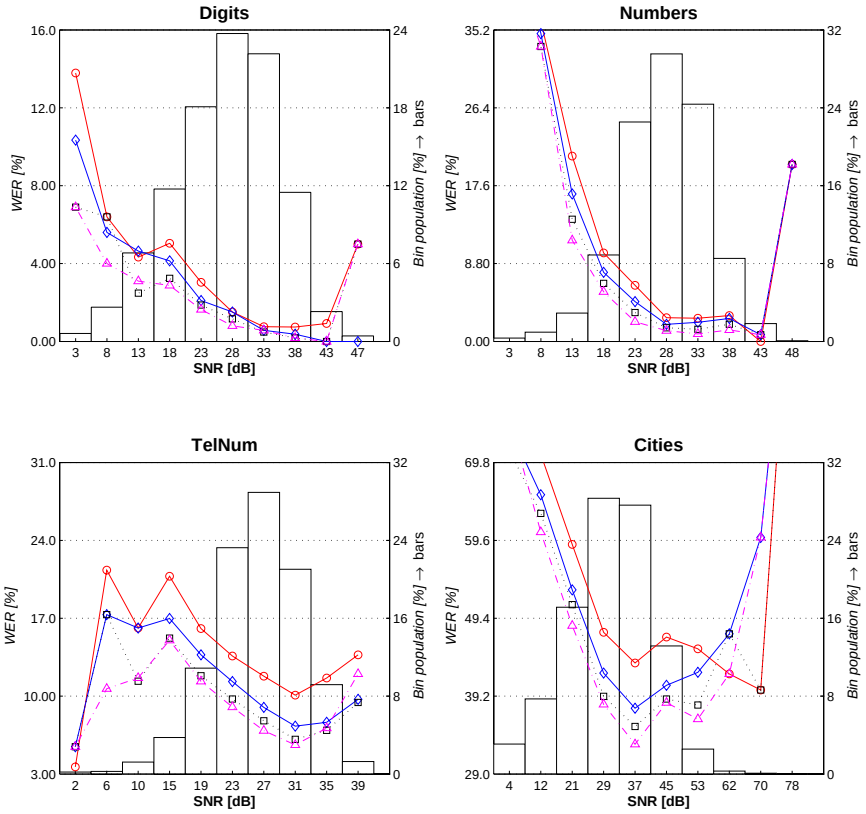


Figure 5.1: WER for baseline models on different values of SNR, PSTN test data (see text for details).

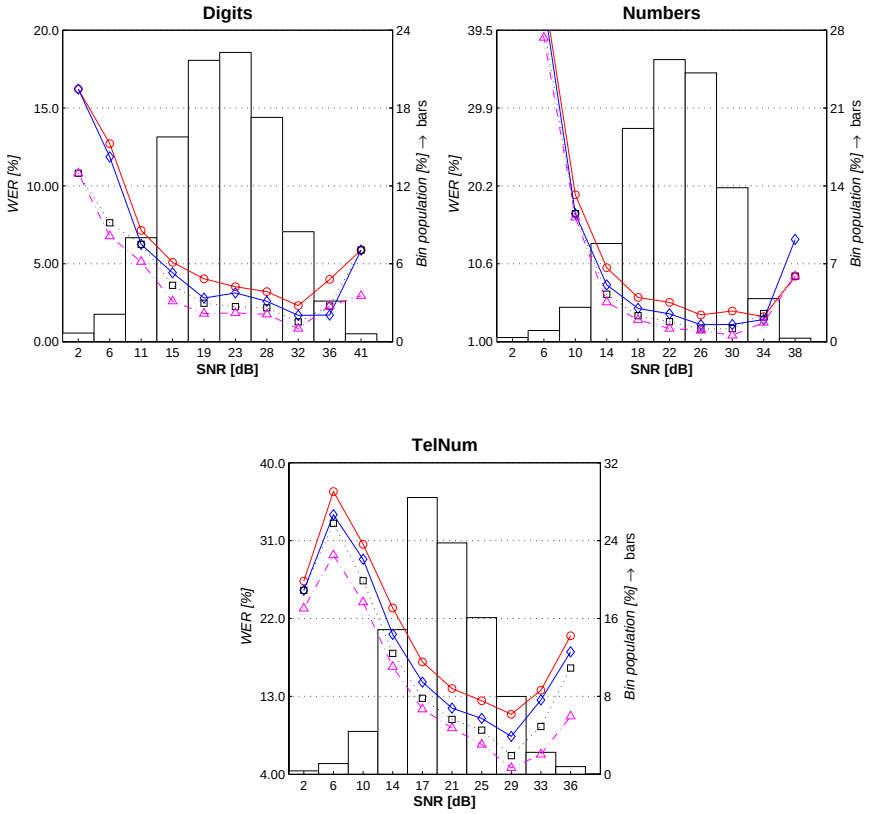


Figure 5.2: WER for baseline models on different values of SNR, GSM data (see text for details).

Tables 5.7 and 5.8 show the partition of test data (in % of words) according to the value of the SNR for PSTN and GSM data, respectively; the first column corresponds to utterances that could not be forced-aligned to the models (single Gaussian densities are used for the alignment), due to all paths being pruned. Three values of SNR are used: "low", "medium" and "high" SNR, and the thresholds are chosen in order to split the training data (PSTN and GSM) into three chunks of comparable size as seen in Table 5.9; where the numbers between parenthesis are the percentage of each chunk with respect to the total amount of data in each channel.

Corpus	Range of SNR			
	Not Aligned	SNR $< 20dB$	$20dB \leq$ SNR $< 28dB$	SNR $\geq 28dB$
`Digits`	0.02%	20.40%	31.62%	47.96%
`Numbers`	0.55%	12.51%	39.45%	47.49%
`TelNum`	3.46%	13.14%	45.97%	37.43%
`Cities`	3.49%	16.63%	20.58%	59.30%

Table 5.7: Partition of PSTN test data per SNR.

We can observe that most of the data are concentrated on the higher quality (lower noise)

part in the case of PSTN and that they are concentrated on the lower SNR for GSM; it is expected that the lower the SNR the lower the performance.

Corpus	Range of SNR			
	Not Aligned	SNR $< 20dB$	$20dB \leq$ SNR $< 28dB$	SNR $\geq 28dB$
`Digits`	0.02%	40.79%	40.97%	18.22%
`Numbers`	1.03%	30.28%	48.57%	20.13%
`TelNum`	9.56%	48.83%	34.66%	6.95%

Table 5.8: Partition of GSM test data per SNR.

As we will see when the results are presented, the error rate is much higher for the utterances that could not be forced aligned; there is a possibility of a transcription error but this last option is not analyzed because it requires a lot of tedious manual analysis.

Channel	Range of SNR			
	Not Aligned	SNR $< 20dB$	$20dB \leq$ SNR $< 28dB$	SNR $\geq 28dB$
PSTN	0.8% (1.0)	16.5% (21.1)	26.8% (34.2)	34.3% (43.8)
GSM	0.2% (0.8)	10.2% (47.3)	8.6% (39.8)	2.6% (12.1)

Table 5.9: Partition of training data per SNR range.

5.1.4 Average Vowel Energy – AVE

As the estimation of the SNR is sensitive to errors in the alignments of low energy speech frames, e.g. plosive or fricative sounds, we decide to use an estimate of the average energy only for observation vectors marked as vowels; semi-vowels are not used in the estimation, just to simplify the process.

Recognition results for matched and *unmatched* models and data are presented in Tables 5.10 and 5.11 for PSTN and GSM channels, respectively.

		Models		
		AVE $< 72dB$	$72dB \leq$ AVE $< 76dB$	AVE $\geq 76dB$
Data	AVE $< 72dB$	8.14	*10.45*	*14.76*
	$72dB \leq$ AVE $< 76dB$	*7.25*	6.73	*7.93*
	AVE $\geq 76dB$	*10.25*	*8.23*	8.31

Table 5.10: `TelNum` results with matched and *mismatched* models and data, AVE dependent (AVED) models and PSTN test data.

		Models		
		AVE $< 72dB$	$72dB \leq$ AVE $< 76dB$	AVE $\geq 76dB$
Data	AVE $< 72dB$	14.70	*17.68*	*24.27*
	$72dB \leq$ AVE $< 76dB$	*12.00*	11.44	*14.17*
	AVE $\geq 76dB$	*12.07*	*10.92*	11.43

Table 5.11: `TelNum` results with matched and *mismatched* models and data, AVE dependent (AVED) models and GSM test data.

Similarly to what was observed for SNRD models in the match/mismatch results (cf. Tables 5.5 and 5.6) the models trained on the mid-range of values perform (slightly) better on the higher-range than the matched models (PSTN data). But for the AVE, there is much more contrast between the matched and mismatched results, for both PSTN and GSM data, therefore AVE is a better source of variability than SNR, for our purposes.

Baseline results on each section of test data are presented in Figures 5.3 and 5.4 with varying number of Gaussian components for PSTN and GSM channels, respectively; The legends are the same for both figures: curves marked with circles, diamonds, squares and triangles represent results for models with 1, 2, 4, and 8 Gaussian components, respectively.

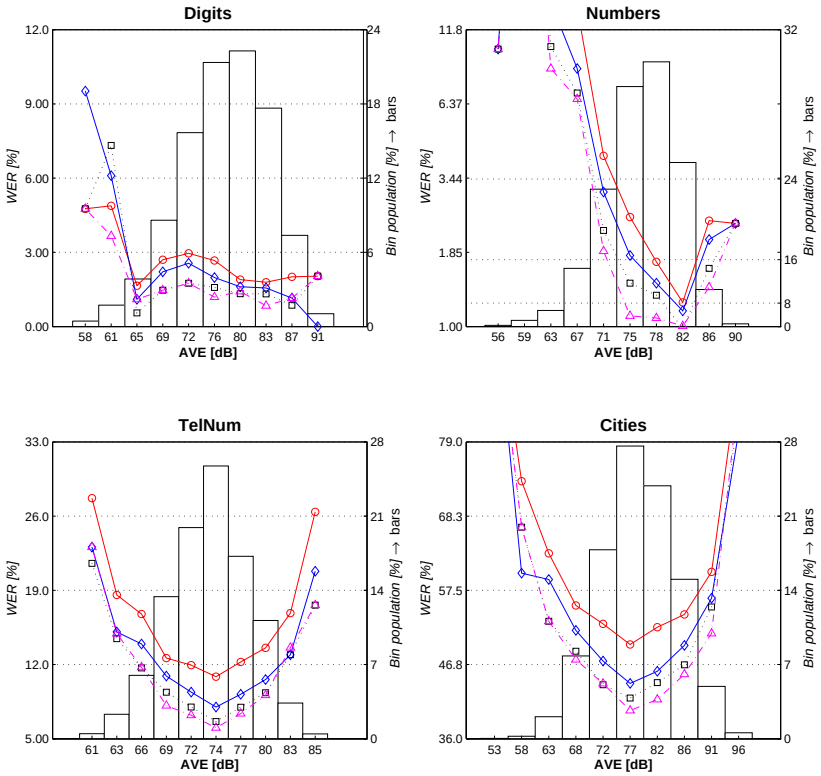


Figure 5.3: WER for baseline models for different values of AVE, PSTN test data (see text for details).

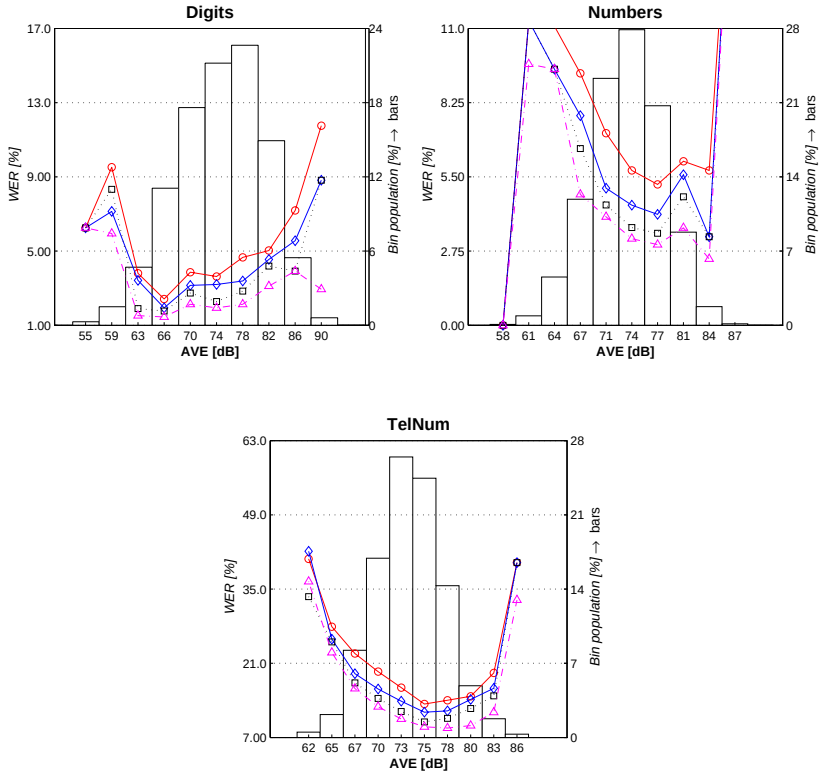


Figure 5.4: WER for baseline models on different values of AVE, GSM data (see text for details).

The number of words in the evaluation tasks are presented in Tables 5.12 and 5.13 for PSTN and GSM data, respectively.

Corpus	Range of AVE			
	Not Aligned	AVE $< 72dB$	$72dB \leq$ AVE $< 76dB$	AVE $\geq 76dB$
`Digits`	0.02%	20.23%	20.91%	58.84%
`Numbers`	0.55%	20.64%	25.35%	53.46%
`TelNum`	3.46%	31.92%	35.26%	29.36%
`Cities`	3.49%	16.46%	17.88%	62.17%

Table 5.12: Partition of PSTN test data per AVE.

Corpus	Range of AVE			
	Not Aligned	AVE $< 72dB$	$72dB \leq$ AVE $< 76dB$	AVE $\geq 76dB$
`Digits`	0.02%	33.80%	21.57%	44.61%
`Numbers`	1.03%	37.74%	33.19%	28.04%
`TelNum`	9.56%	30.94%	35.83%	23.67%

Table 5.13: Partition of GSM test data per AVE.

Again, the thresholds are chosen to split the training data as uniformly as possible; the PSTN and GSM corpora are combined and the breakdown of the training data is presented in Table 5.14.

Channel	Range of AVE			
	Not Aligned	AVE $< 72dB$	$72dB \leq$ AVE $< 76dB$	AVE $\geq 76dB$
PSTN	0.8% (1.0)	26.2% (34.0)	23.3% (29.7)	27.7% (35.3)
GSM	0.2% (0.8)	7.9% (36.5)	7.1% (33.0)	6.4% (34.1)

Table 5.14: Division of training data per AVE.

5.1.5 SNR and AVE

Tables 5.15 and 5.16 show the WER results of an arrangement of the data according to both measurements at a time, PSTN and GSM data respectively (4 components). The values between parentheses are the percentage of data in each bin and we use the same bin edges as for the training (the center values are indicated in the tables and change due to different maxima and minima values for the different tasks).

AVE	SNR		
	10.51	*24.00*	*38.97*
62.97	3.65 (6.95)	0.84 (5.05)	0.00 (1.16)
73.00	3.65 (5.79)	1.70 (13.67)	0.00 (8.53)
84.18	3.58 (7.67)	1.64 (12.91)	0.77 (38.28)

(a) Digits

AVE	SNR		
	10.11	*24.00*	*39.05*
61.88	15.68 (5.84)	4.34 (5.76)	2.17 (0.64)
73.00	9.24 (5.11)	2.41 (19.00)	1.26 (9.88)
83.75	3.42 (1.62)	2.42 (14.91)	1.49 (37.23)

(b) Numbers

AVE	SNR		
	10.03	*24.00*	*34.77*
64.69	11.32 (6.49)	10.86 (13.77)	8.89 (0.16)
73.00	10.98 (5.03)	7.68 (25.86)	6.01 (18.27)
81.34	20.70 (2.08)	10.89 (7.99)	7.05 (20.34)

(c) Telephone numbers

AVE	SNR		
	10.01	*24.00*	*55.05*
60.57	65.59 (3.88)	38.96 (3.91)	36.65 (2.53)
73.00	58.49 (5.83)	37.22 (6.34)	34.61 (13.13)
87.04	61.80 (7.53)	51.64 (11.05)	37.73 (45.79)

(d) City names

Table 5.15: WER on PSTN data for baseline models, arranged by SNR and AVE.

AVE	SNR		
	10.02	*24.00*	*35.34*
61.46	3.77 (16.12)	0.27 (6.58)	0.00 (1.72)
73.00	3.00 (14.28)	1.84 (13.62)	1.74 (3.07)
83.83	6.19 (10.40)	2.92 (20.78)	2.39 (13.44)

(a) Digits

AVE	SNR		
	10.15	*24.00*	*34.19*
62.93	8.26 (15.55)	4.09 (7.40)	0.00 (0.01)
73.00	6.89 (12.35)	2.82 (30.19)	2.57 (6.17)
82.37	12.30 (2.70)	2.89 (11.48)	3.26 (14.15)

(b) Numbers

AVE	SNR		
	10.17	*24.00*	*33.11*
65.42	18.39 (15.10)	14.55 (3.48)	— n/a
73.00	14.18 (29.66)	8.57 (23.49)	5.57 (2.09)
81.51	15.42 (9.23)	10.00 (11.36)	8.66 (5.59)

(c) Telephone numbers

Table 5.16: WER on GSM data for baseline models, ranked by SNR and AVE.

The results show that the error rate tends to be higher for low SNR and it improves with the increase of the SNR. When the energy level is too low or too high, the error rate increases; on the lower side because the signal is degraded by noise and on the higher side, because the signal tends to be distorted by saturation, and possibly due to people increasing the effort to speak. On the `Cities` task (the data are from a real-world application) it was observed a tendency towards high energy utterances; that could be due to automatic gain adaptation.

5.2 Phoneme Position

The idea underlying position-dependent units is to exploit prosodical effects on the words to create "extra" models for phonemes in different positions in a word; e.g. in French the stress appears mainly on the last syllable of a word (actually on the last syllable in the same "breath group" [1]). Words that have just one phoneme are also likely to change the way the phoneme is uttered compared with the pronunciation of the phoneme when it appears elsewhere in a word.

We adopt a different strategy from the one currently used to determine when it is appropriate to provide an extra model for a phoneme at a given position. The currently used strategy is

[1] *groupe de souffle*

based on *tagged clustering* [Paul 1997]. Tagged clustering is similar to the usual tree-based state tying scheme [Young 1994]; the difference is that tags are attributed to the phonemes and the set of questions is modified to include specific questions about the different tags. In this case, the tags would be related to the position within a word, i.e. "initial", "final", and "single"; the other position is not actually tagged and corresponds to a generic word-internal phoneme.

The strategy adopted in this work is to estimate the distribution of the duration of each phoneme in the different positions and by comparison of the resulting distributions, it is decided to create extra models for a particular phoneme position. We chose to proceed like this because usual clustering uses only the acoustical information present on the observation vectors (or the parameters of single Gaussian models) to decide whether an extra model is needed or not; it is also possible to create extra models for all positions at which a phoneme can appear, but this may result into a waste of parameters, as it is not likely that every position would require a different model. The distributions of the duration for all possible positions are estimated from the same (forced) alignments that are used to estimate the parameters of mixture models.

The duration distributions are compared using the symmetric Kullback-Leibler divergence [Basseville 1989]; which for two distributions f and g is given by:

$$KL_{symmetric}(\mathrm{f},\mathrm{g}) = \frac{1}{2}\int_{\mathcal{X}} \left[\mathrm{f}(x)\log(\frac{\mathrm{g}(x)}{\mathrm{f}(x)}) + \mathrm{g}(x)\log(\frac{\mathrm{f}(x)}{\mathrm{g}(x)})\right] \mathrm{d}x$$

The divergence can be seen as a "distance" between the two distributions. The divergence between all 4 instances of each phoneme duration distributions are calculated and then a model is assigned to the instances which are farther than a threshold from the others; "clusters" of instances that are separated from the others have a model of their own. To identify the extra instances, a suffix is added to the phone symbols:

- "_i": phoneme at initial position
- "_f": idem, final
- "_s": single phoneme word

Word-internal receive no suffix; if, for example, instances for word final and single phoneme word are clustered, we assign a suffix "_fs" to the phoneme. As the number of phoneme is small (36 for French), the instances were hand-clustered, but it could be automatically clustered as well. Table 5.17 shows the phonemes with their usual IPA symbols with the ASCII symbols that are

used in our system and the instances that resulted from the clustering procedure; instances in the table with the same mark ("1", "2", "3", and "4") are those that are grouped together and have the same model.

Phoneme symbol		Instance / suffix			
IPA	ASCII	initial	final	single	internal
		"_i"	"_f"	"_s"	no suffix
p	p	1	2	3	3
t	t	1	2	3	3
k	k	1	2	3	3
b	b	1	2	3	3
d	d	1	2	3	1
g	g	1	1	2	2
m	m	1	2	1	1
n	n	1	1	2	3
ɲ	nj	1	2	3	3
ŋ	ng	1	2	1	1
f	f	1	2	1	1
v	v	1	2	1	1
s	s	1	2	1	1
z	z	1	2	1	1
ʃ	ch	1	2	1	1
ʒ	ge	1	1	2	2
l	l	1	1	2	3
r	r	1	1	2	2
j	j	1	2	3	3
w	w	1	2	2	2
ɥ	Y	1	2	2	2
u	u	1	2	3	2
o	au	1	2	2	2
a	a	1	2	2	2
ɛ	ai	1	3	2	3
ɔ	o	1	2	2	2
ɑ̃	an	1	2	2	2
ɛ̃	in	1	3	2	3
ɔ̃	on	1	2	1	1
e	ei	1	2	3	3
i	i	1	2	3	1
y	y	1	2	3	2
ə	e	1	2	1	1
œ	oe	1	2	3	3
ø	eu	1	2	3	4
œ̃	un	1	2	2	1

Table 5.17: Position dependent phones.

Only the phoneme "ø" has all four possible instances; the average number of alternatives per phoneme is 2.5. Table 5.17 is divided, from top to bottom, as occlusives, nasals, aspirants, liquids, semi-vowels and vowels.

5.3 Model Combination Schemes

Sometimes, but not always, it is possible to identify some of the sources of variability from data and then load the appropriate model parameters. Thus we decide to combine the multiple

models into the grammar describing a particular task and let the decoding process chose the best path. We try to take advantage of the well-known fact that when data and models do not match, the resulting alignment score is quite low; thus during decoding, the models which best match the data should provide higher acoustical scores and the beam search procedure should prune out the irrelevant competing hypotheses. Three combination schemes are evaluated, as described in the next sections.

5.3.1 Syntactical

There is a completely separate path in the grammar for each one of the models; for an isolated word task this is equivalent to putting the different pronunciations in parallel, as illustrated in Figure 5.5 for the word "sept" (seven) with *female* (marked with "_f") and *male* ("_m") models; the dotted transition corresponds to a null transition (no symbol associated), the arcs correspond to the units (phones).

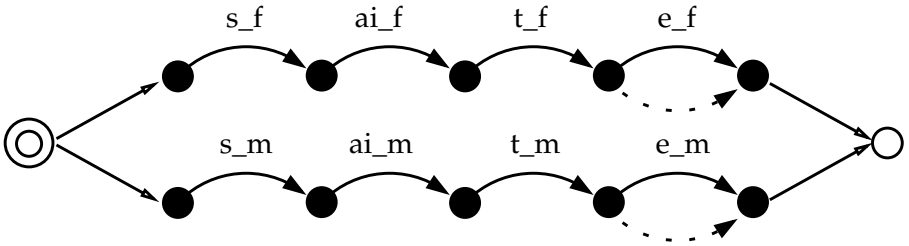


Figure 5.5: Syntactical level combination.

This scheme tries to profit from the possible correlation between the models coming from the same source of variability.

5.3.2 Lexical

Each phone is represented by all its models in parallel, as shown in Figure 5.6.

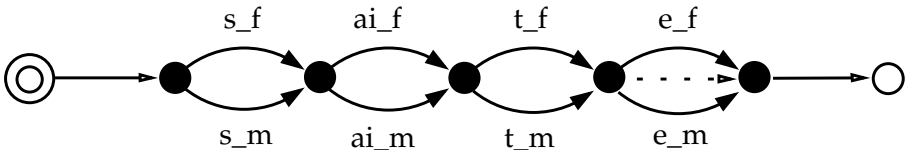


Figure 5.6: Combination at lexical level.

It should be noted that even if the paths in the *Syntactical* scheme are a subset of those in the *Lexical*, the recognition results are different. The idea behind this scheme is to allow local deviations from the source of variability, i.e. in the case of gender dependent models, an

utterance produced by a female speaker may be locally better modeled with models trained on male data.

5.3.3 Acoustical

There is a single model per phone, but the Gaussian mixtures have components coming from all sources of variability (in this case gender and channel) as illustrated in Figure 5.7; this figure shows the Gaussian components used in the mixture for state number 2 of phone /s/ which come from the various specialized models.

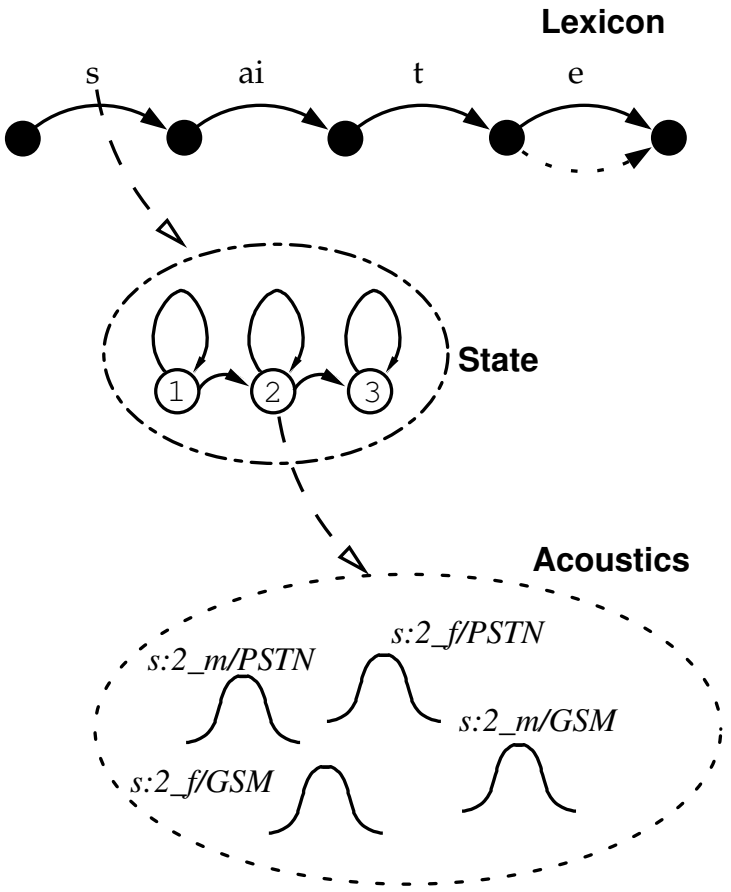


Figure 5.7: Acoustical level combination.

In this figure are represented the Gaussian components coming from gender dependent models (let these models be called *source models*) trained on PSTN and on GSM data, that are used in state 2 of the phone /s/. In the case that the source models have Gaussian mixture densities, the mixture weights in the combined models are proportional to the original weights, i.e. the original weights are scaled by the number of source models ($w_{\text{comb}} = \frac{w_{\text{source}}}{N}$, where N is the number of source models).

5.4 Experiments

Models (Gaussian mixtures) for all classes are trained as described in Section 1.5.1; models are trained for each class-specific data chunk Firstly, single Gaussian models (context-independent) are estimated and used to align the corresponding training data; note that all single Gaussian models are initialized from PSTN models. Context-dependent models are initialized with the CI models and the same training process described in Section 1.5.1 is used to estimate parameters for Gaussian mixture models.

5.4.1 Position Dependent Models

The pronunciation was adapted according to the models that were obtained from the analysis of the duration distribution as explained in Section 5.2, the data are aligned with single Gaussian models and only PSTN data are used.

Figure 5.8 illustrates the results for position dependent (PD) and baseline models, marked with triangles and circles, respectively; confidence intervals on the baseline results are also shown.

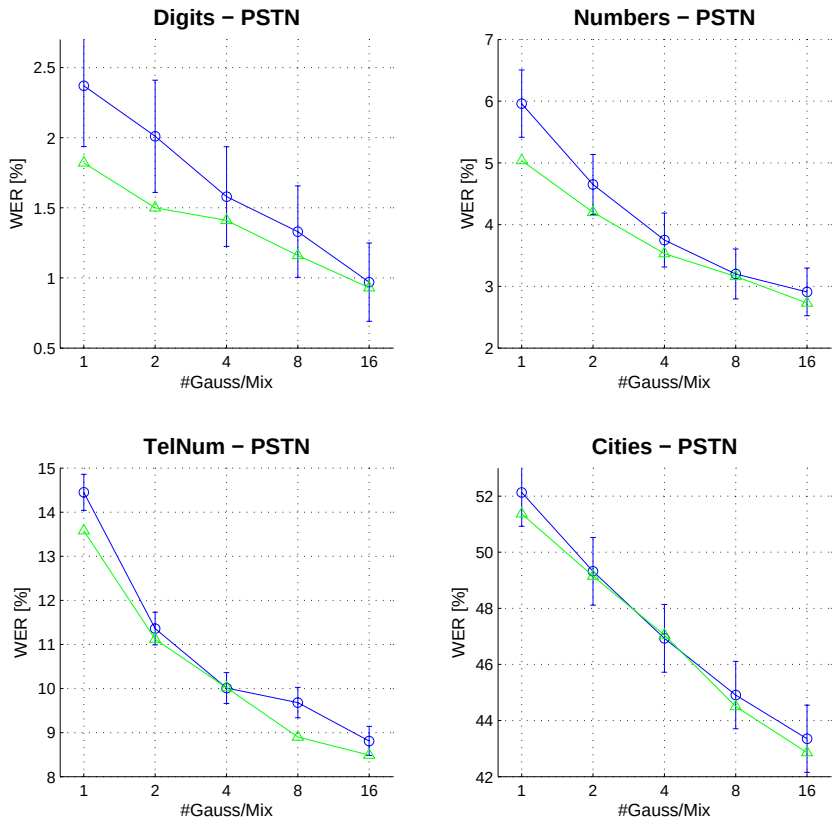


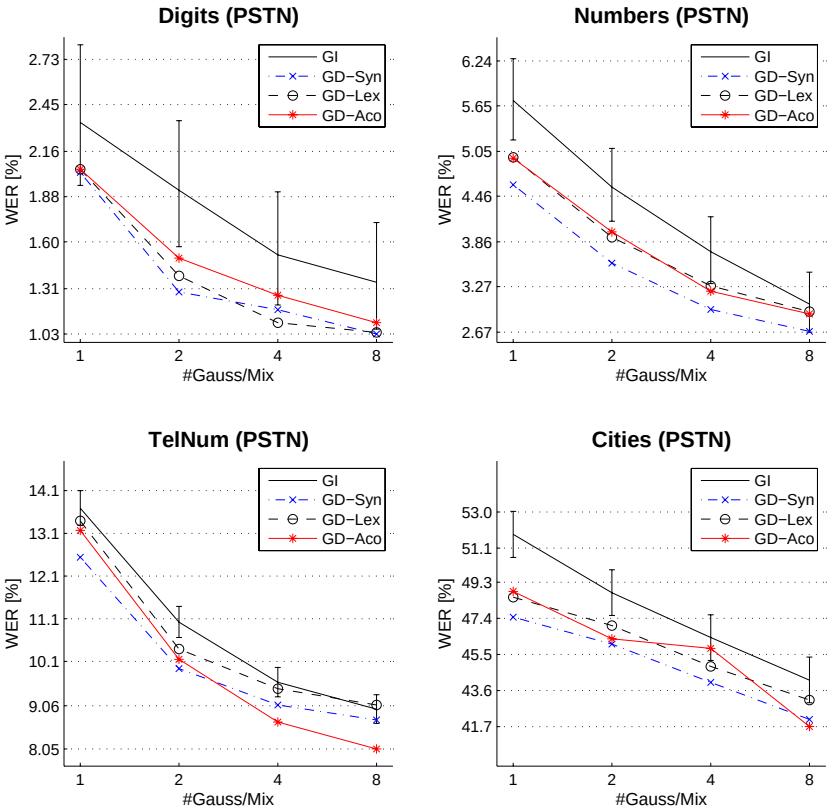
Figure 5.8: Results for position dependent (PD) and baseline models.

Overall the results are better with PD models, but the gain is not always statistically significant. The advantage reduces with increasing number of components per mixture; we believe that this shows that above a certain complexity of the models, there is not much to gain with position dependent modeling schemes and that ML estimation of a single large model yields similar performance.

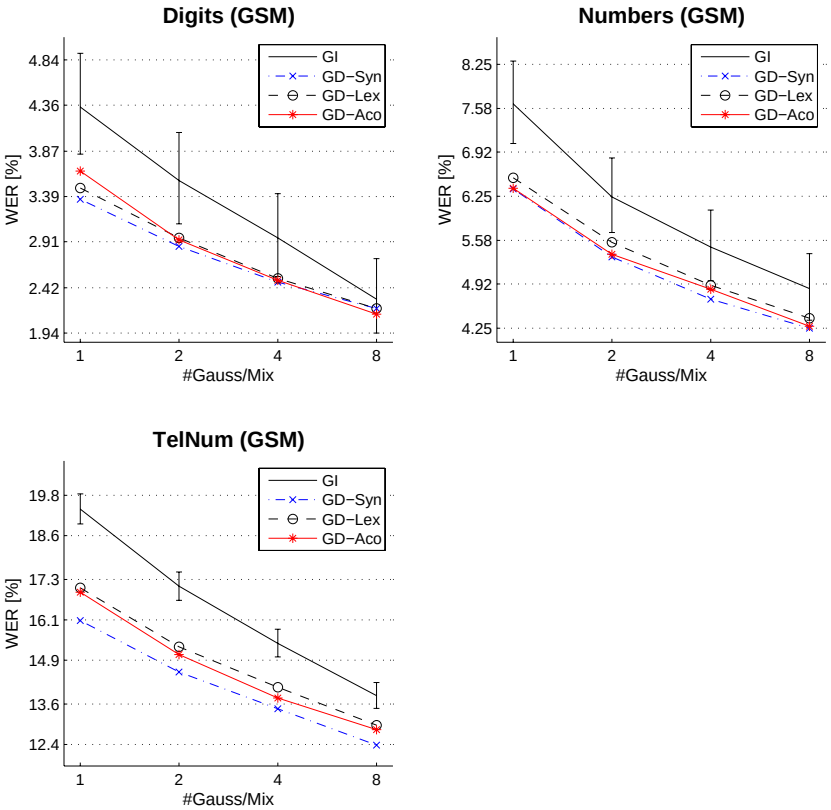
5.4.2 Gender Dependent models

Training data (PSTN and GSM channels) are split into female and male speakers and models are trained on each chunk then combined with the three combination schemes; models for silence (leading, trailing and short pause) are trained on all data. The results, along with baseline gender independent (GI) models are presented in Figure 5.9, with increasing number of Gaussian components in the models; for the combined models, the number of components is the one for the GD models, thus the combined models have twice as many Gaussian densities in the pool.

GD models provide higher performance (statistically significant) for most of the tasks and number of components. The combination scheme that yields the best results is task-dependent; on the `Digits` task *Syntactical* and *Lexical* schemes are better, on `Numbers` the syntactically combined models clearly outperform the others while on `TelNum` , the best scheme is between *Syntactical* and *Acoustical*, the latter yielding higher performance with increasing number of components, finally, on `Cities` the three schemes perform about on the same level, the syntactically combined models being the most consistent with increasing model complexity.



(a) PSTN channel.

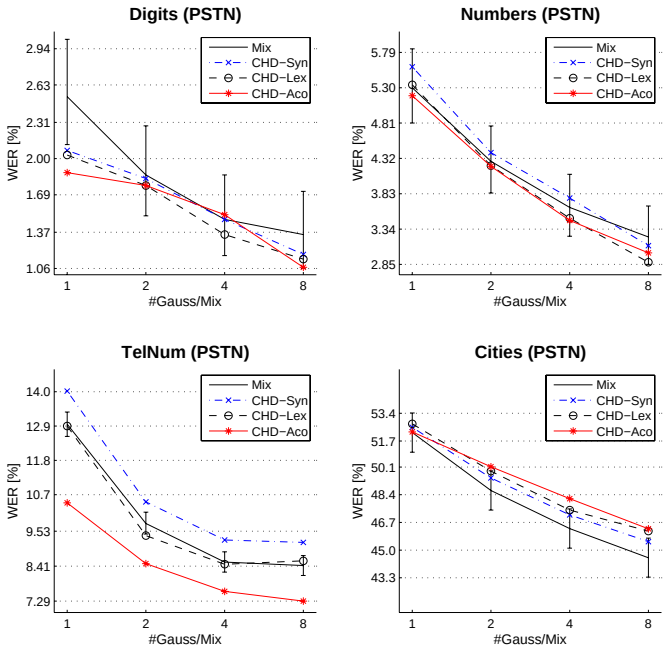


(b) GSM channel.

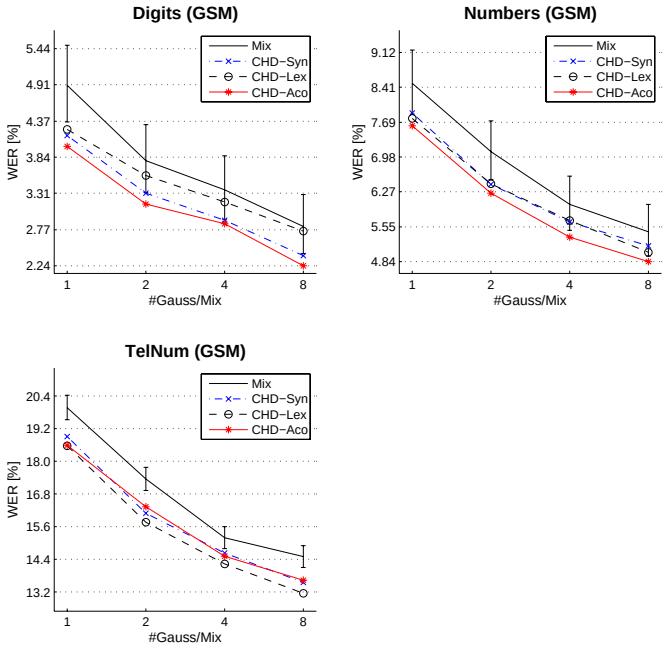
Figure 5.9: Results for gender dependent (GD) and baseline (GI) models.

5.4.3 Channel Dependent Models

The models trained separately on PSTN and on GSM data are now combined, resulting in models that can be used with data coming from any of the channels. The results are presented in Figure 5.10; the baseline results are for mixed training models.



(a) PSTN data.



(b) GSM data.

Figure 5.10: Results for channel dependent (CHD) and baseline (Mix) models.

The results on PSTN data are not consistent across the tasks; we will concentrate the analysis on the two most difficult tasks: `TelNum` and `Cities` . On the telephone number task, acoustically combined models statistically outperform the other schemes (baseline as well) and the *Syntactical* models yield lower performance than all the others; the latter results are quite unexpected because we thought from the beginning that in the case of mismatch, the matched model would "dominate" the decoding process and the performance would not degrade. On the city name task, the combined models are always outperformed by the baseline models.

On GSM data, the combined models outperform the baseline. Acoustically combined models yield better performance on most of the tasks, except on recognition of telephone numbers, where the *Lexical* scheme gives the best results.

5.4.4 Models Dependent on the SNR

Training data (PSTN and GSM channels) are split according to the value of the SNR and SNR-dependent models are trained on each chunk and then combined; models for silence (leading, trailing and short pause) are trained on all data. Results on PSTN data are presented in Figure 5.11 and in Figure 5.12 for the GSM channel.

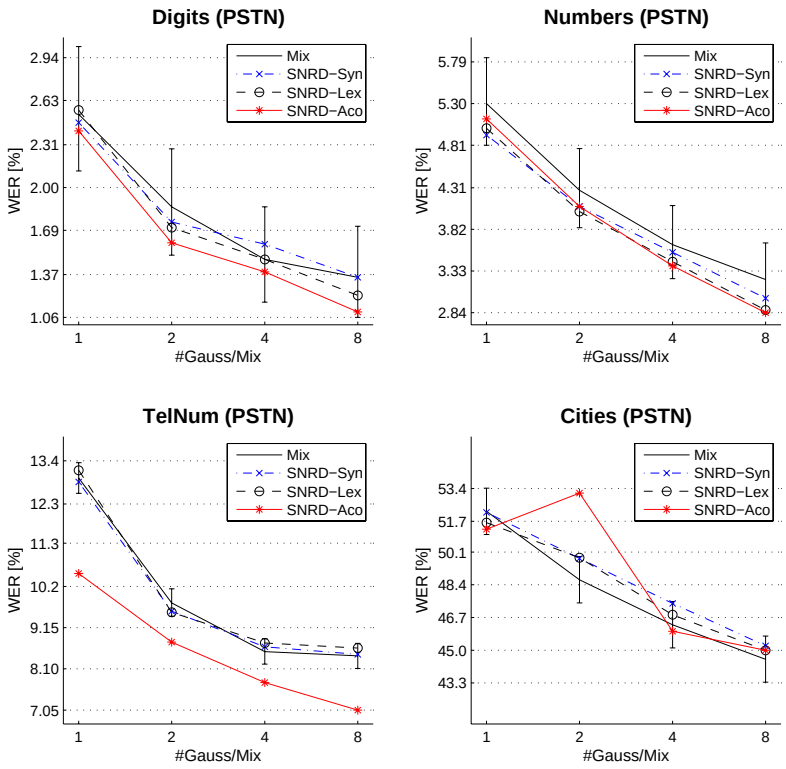


Figure 5.11: Results for SNR dependent (SNRD) and baseline (Mix) models, PSTN channel.

On PSTN data, there is no clear winner scheme, but overall the acoustically combined models provide the highest performance. It is interesting to note that on telephone numbers, the *Acoustical* combination scheme performs quite well. On the city name task and for acoustically combined models, there is an outlier for 2 components, due to an error in the script doing the test (unfortunately, we could not rescue the correct result).

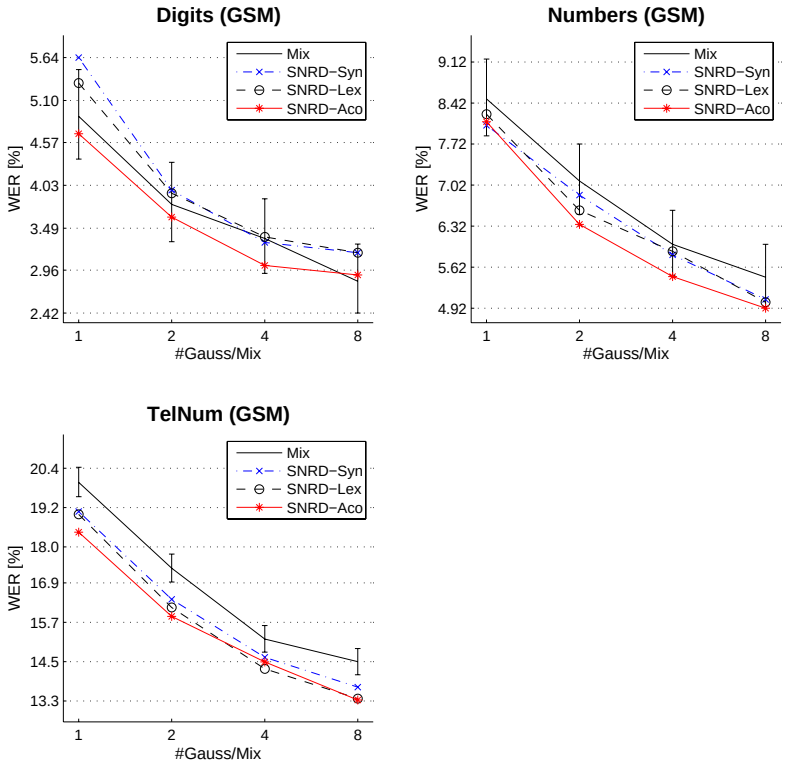
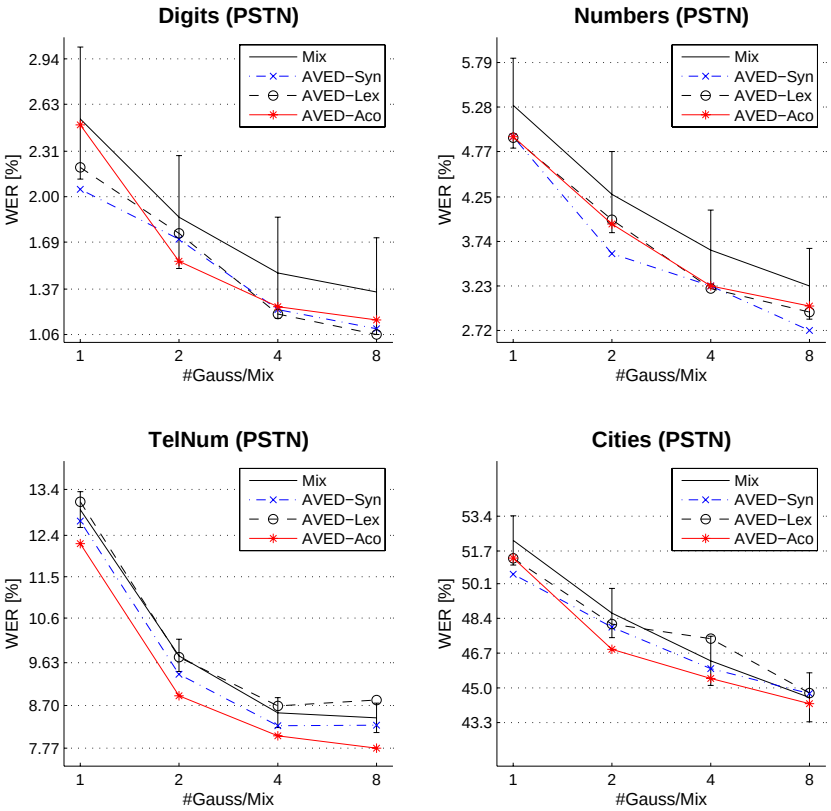


Figure 5.12: Results for SNR dependent (SNRD) and baseline (Mix) models, GSM channel.

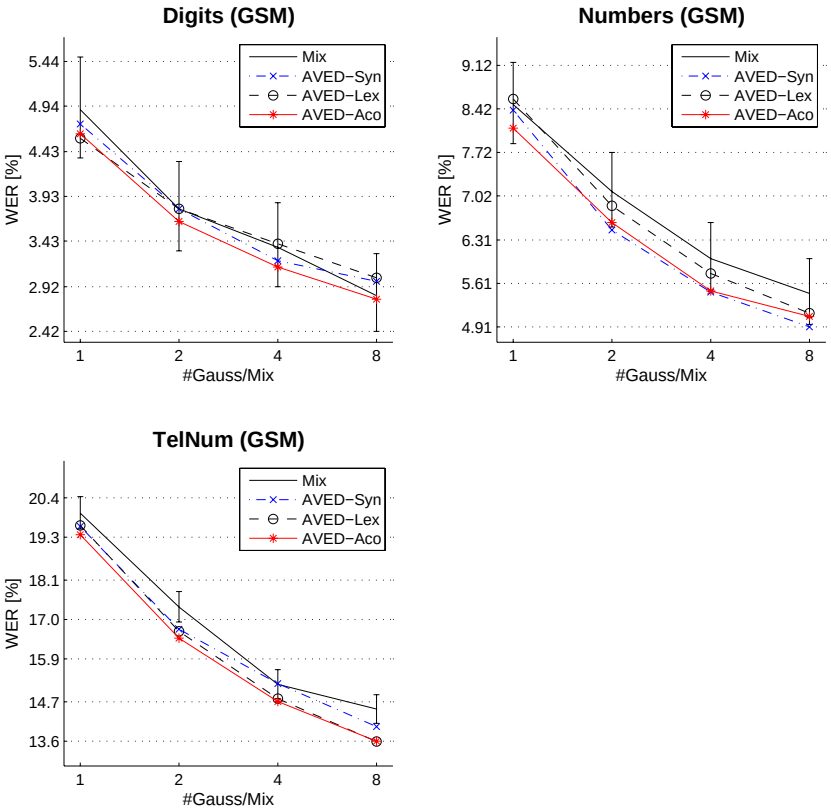
When recognizing GSM data, the acoustically combined models yield the highest performance overall; for the `TelNum` task, the gain in performance is statistically significant. But there are three times more Gaussian densities in the combined models than in the baseline models.

5.4.5 Models Dependent on the AVE

Again, models are trained on specific chunks of the training corpora (PSTN and GSM), selected according to the average vowel energy on the vocalic segments, resulting into AVE dependent (AVED) models. Results are presented in Figure 5.13 for PSTN and GSM data.



(a) PSTN channel.



(b) GSM channel.

Figure 5.13: Results for AVE dependent (AVED) and baseline (Mix) models.

We can see that the acoustical scheme is the best overall; these models yield most of the time the lowest error rates which are also outside the confidence interval, being significantly better than the baseline models. Again, the task that most benefits from the combined models is `TelNum` ; there is also a slight improvement on the city name task.

5.4.6 Combining GD and CHD Models – 4Models

All four models (female-PSTN, female-GSM, male-PSTN, and male-GSM) are combined with the proposed schemes; the results for PSTN and GSM channel are respectively reported in Figure 5.14 and Figure 5.15. There are now 4 times more Gaussian densities in the resulting combined model than in the baseline models (mixed style training).

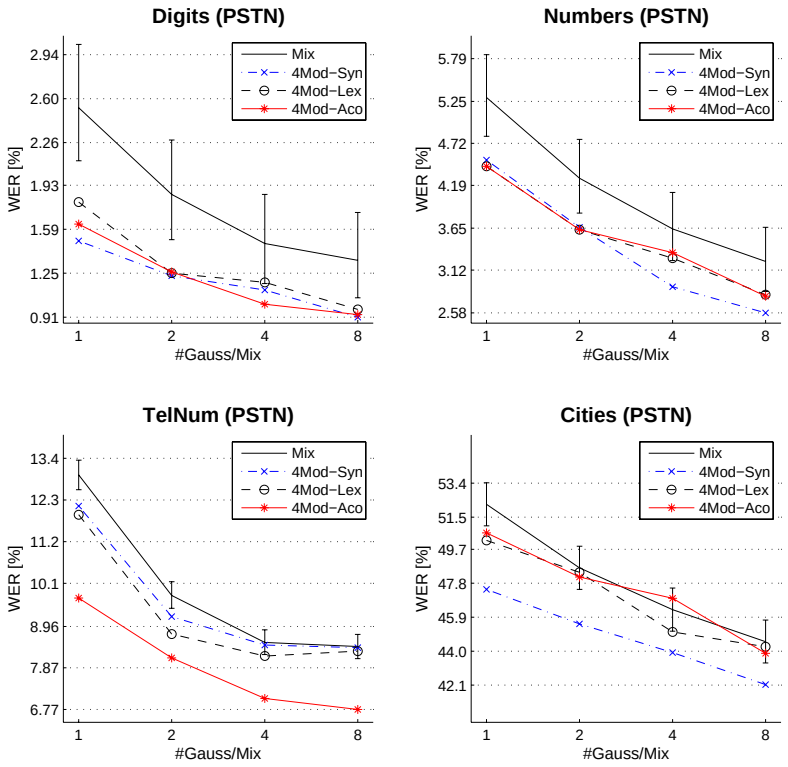


Figure 5.14: Results for GD+CHD and baseline (Mix) models, PSTN channel.

As already observed, there is not always a clear advantage of a scheme over the others. On PSTN data, there is a clear advantage for one scheme or the other for the most difficult tasks. Acoustically combined models are better on telephone numbers and syntactically models are better to recognize city names; in both cases, there is a significant improvement over the baseline results. When GSM data are recognized, the combined models always outperform baseline models; the *Acoustical* scheme is superior on the easier tasks (`Digits` and `Numbers`)

and syntactically combined models are better on `TelNum` .

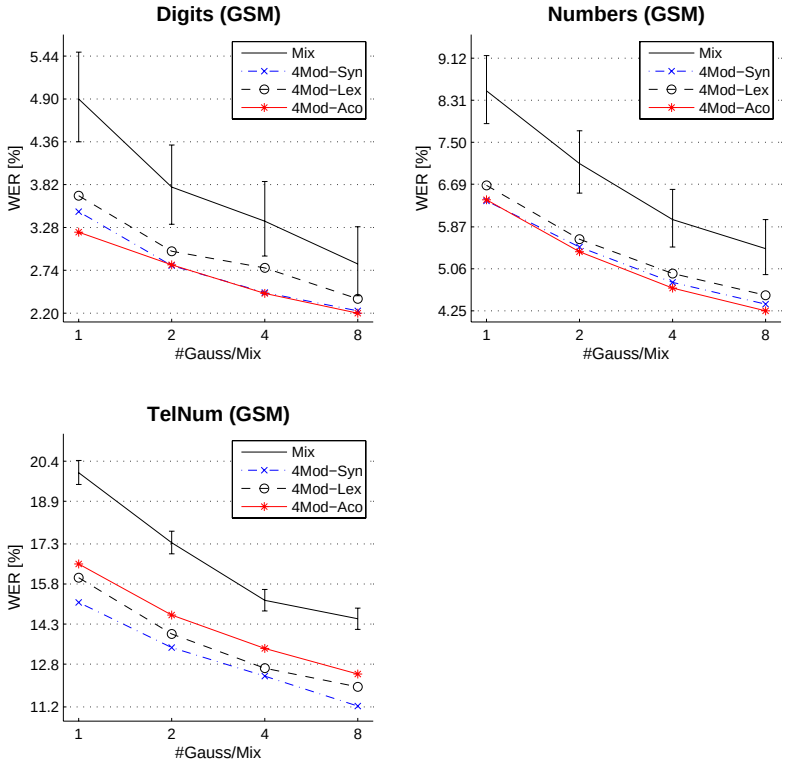
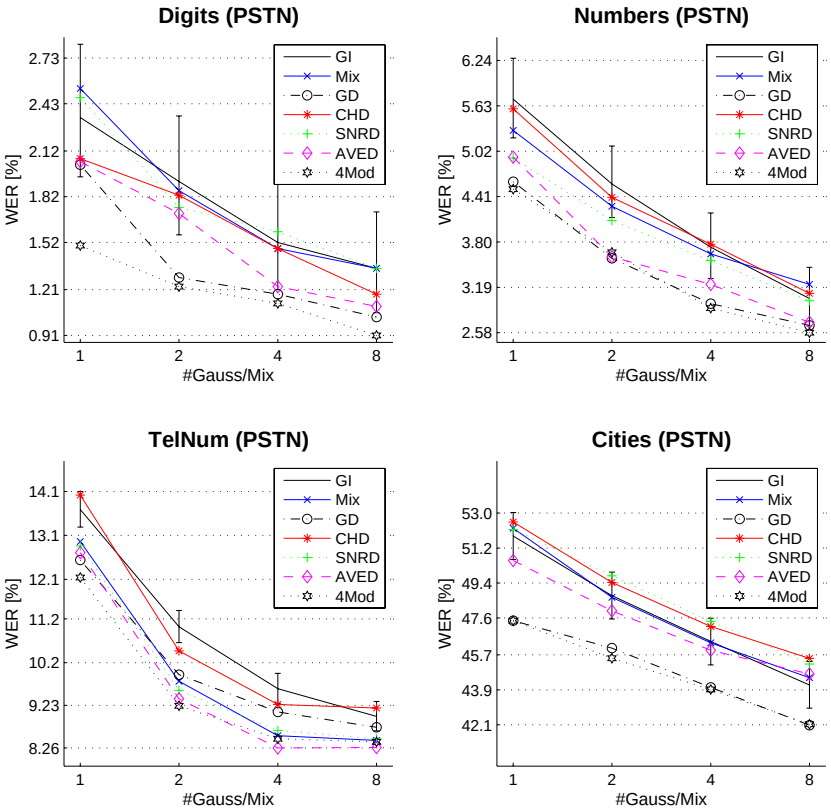


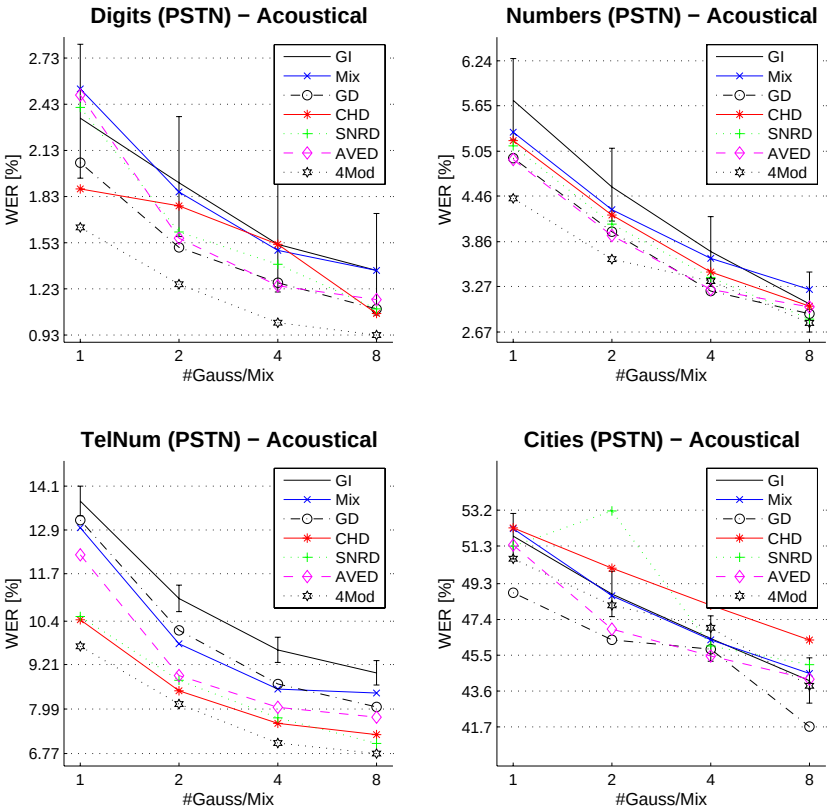
Figure 5.15: Results for GD+CHD and baseline (Mix) models, GSM channel.

5.4.7 Performance Comparison of the Sources of Variability

The results of the previous sections are now presented in another way to compare the different sources of variability; the two best overall schemes (*Syntactical* and *Acoustical*) are used. The results are presented in Figure 5.16 and in Figure 5.17, for PSTN and GSM data, respectively.

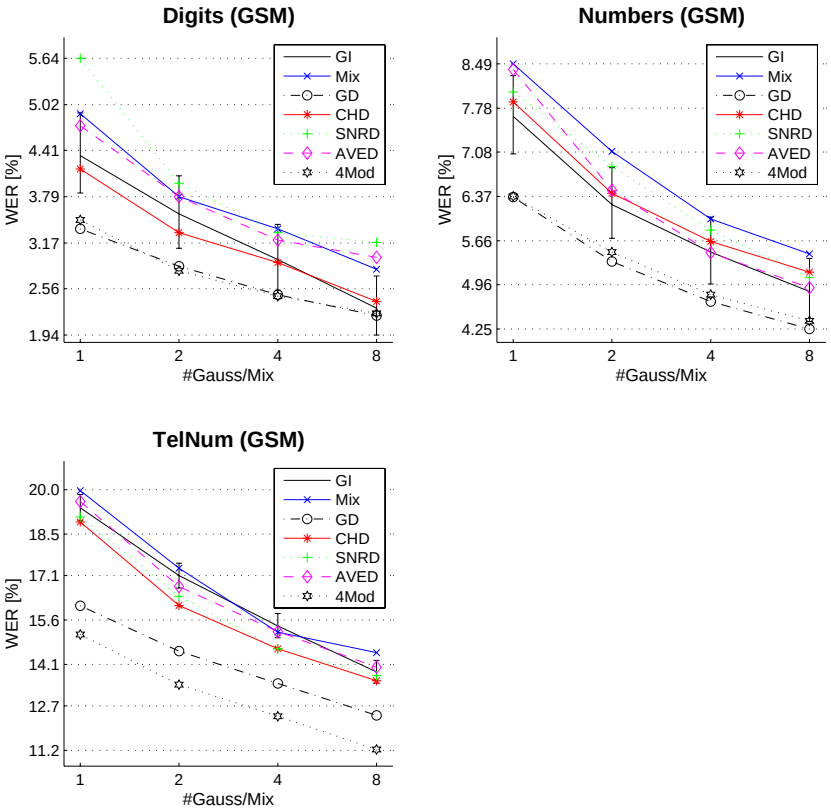


(a) Syntactical.

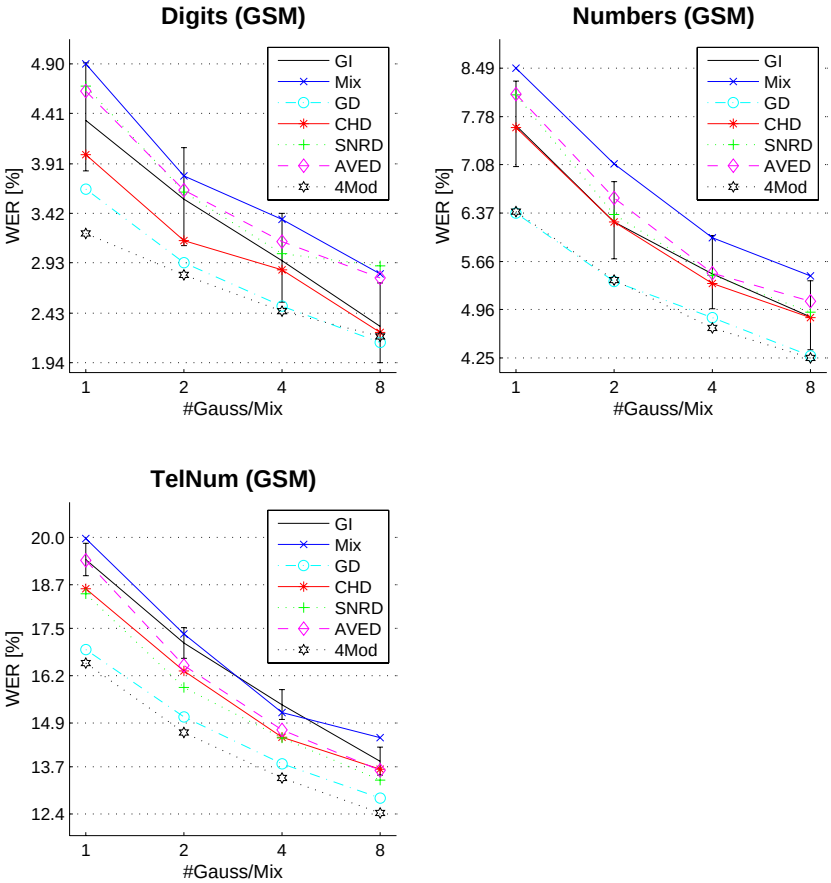


(b) Acoustical.

Figure 5.16: Results for combined models PSTN test data.



(a) Syntactical.



(b) Acoustical.

Figure 5.17: Results for combined models GSM test data.

Looking at the results using PSTN data, it can be seen that the best models result from the combination of gender and channel dependent models (denoted "4Mod"), followed by the gender dependent models; the next best are models dependent on the average vowel energy. As it could be expected, there is only a small gain of the 4Mod compared to GD models with the syntactically combination; this can be foreseen because GD models are quite sensitive to the mismatch (in gender), thus the hypothesis with correct gender stands out in the decoding process. For acoustically combined models, the behavior is similar with respect to the best sources (4Mod or GD).

With GSM data, 4Mod and GD models yield the highest performance, but this time the next to best are combined channel dependent models. This can be expected as the models are trained in isolation thus the difference in the amount of training data does not affect the resulting models.

5.5 Discussion and analysis

5.5.1 Adapted and Combined Models

To assess the effects of training data sparsity due to the split into chunks, we conducted an experimental test in which the chunk-specific models are obtained through adaptation of the baseline models, instead of re-training as in the previously reported experiments. Adaptation was performed only for SNR dependent models.

Three adaptation experiments are performed: 4 components mixture models are used in the first two experiments, the difference is that in one experiment the data are re-aligned (`Adap1`), and in the other experiment the same alignments as previously used to train the models (cf. Section 5.4.4) are used for adaptation (`Adap2`). We try to check whether the quality of the "alignments" has impact on the adaptation or not. In the third experiment, mono Gaussian models are adapted and then the resulting alignment is used to train mixture models up to 4 components (`Adap3`). The latter experiment aims at the improvement of the back-off parameters (remind that we back-off to mono Gaussian models). Models with 4 components are used because it is less likely to incur into estimation errors due to lack of data than for models with more components; moreover the resolution is "higher" than for models with fewer (like single Gaussian densities) components.

Models are adapted using class-based Maximum Likelihood Linear Regression

(MLLR) [Gales 1996, Gales 1997, Delphin-Poulat 2001] as described in Section 1.5.4. We adapt the means and variances of the Gaussian components only but not the mixture weights. The results for all adaptation experiments are presented in Tables 5.18 and 5.19, along with reference results for mixed training and the trained SNR dependent models, for *Acoustical* and *Syntactical* combination schemes, respectively.

Models	TelNum		Cities
	PSTN	GSM	PSTN
Mixed training	8.54	15.19	46.32
SNDR (re-training)/Aco	7.75	14.49	45.99
Adap1/Aco	7.39	13.93	45.59
Adap2/Aco	7.49	14.14	46.91
Adap3/Aco	7.65	14.23	45.04

Table 5.18: Results for baseline and adapted SNRD models, acoustical combination.

The results show that there is a slight advantage when models are adapted instead of trained (with the usual procedure). It can be also seen that the "quality" of the alignments has influence on the results, as error rate is lower for `Adap1` than for `Adap2` models.

The idea behind the third adaptation experiment was to evaluate the influence of the back-off parameters (single Gaussian); there is a small improvement over the trained models, showing that it is indeed important to have acceptable quality models to back-off to.

Models	TelNum		Cities
	PSTN	GSM	PSTN
Mixed training	8.54	15.19	46.32
SNDR (re-training)/Aco	8.66	14.63	47.73
Adap1/Syn	8.24	14.49	45.84
Adap2/Syn	8.63	14.72	46.60
Adap3/Syn	8.64	14.81	46.95

Table 5.19: Results for baseline and adapted SNRD models, syntactical combination.

The higher performance of the *Acoustical* combination scheme suggests that it is important to have multi-modal densities, but that their estimation should be somewhat controlled, because the usual data-driven estimation is not sufficient to use the larger number of parameters in the best possible way by itself. With the multiple training chunks the parameter estimation is somewhat "guided" and then it is possible to train "sharper" components, increasing performance.

5.5.2 Gender and Channel

Here we analyze the results, trying to understand how the different sources of variability are linked to the errors. For that we use the results with the *Syntactical* combination scheme because the recognition output for these models has a tag that indicates the path in the grammar that generated the most likely hypothesis. To make the analysis simpler only the results for models with 4 Gaussian components are examined.

In Tables 5.20 and 5.21 the WERs for utterances identified with the same variability class and with other classes are presented, on PSTN and GSM data respectively; the values between parentheses are the percentage of utterances with tag of the same and other classes.

Models	Digits		Numbers		TelNum		Cities	
	Same	Other	Same	Other	Same	Other	Same	Other
GD	0.84 (93.58)	6.25 (6.42)	2.79 (96.11)	5.78 (3.84)	8.69 (93.80)	10.90 (6.20)	40.92 (88.37)	52.40 (11.63)
CHD	1.50 (84.44)	1.36 (15.56)	3.39 (81.05)	5.11 (18.95)	9.16 (88.44)	8.39 (11.56)	44.08 (88.14)	60.93 (11.86)
4Mod	0.84 (77.61)	2.08 (22.39)	2.65 (78.27)	3.63 (21.73)	8.23 (79.73)	8.39 (20.27)	40.41 (81.99)	53.78 (18.01)

Table 5.20: WER for models and data of same (and other) variability class, PSTN data.

Models	Digits		Numbers		TelNum	
	Same	Other	Same	Other	Same	Other
GD	2.08 (90.32)	6.10 (9.68)	4.15 (92.59)	9.11 (7.41)	12.65 (93.70)	15.12 (6.30)
CHD	2.49 (83.08)	4.96 (16.92)	5.57 (95.13)	5.11 (4.87)	13.93 (91.57)	17.60 (8.43)
4Mod	1.87 (78.27)	4.61 (21.73)	4.22 (89.95)	8.25 (10.05)	11.48 (88.59)	13.99 (11.41)

Table 5.21: WER for models and data of same (and other) variability class, GSM data.

The results in Tables 5.20 and 5.21 show that in general, when models for another variability class provide the most likely recognition hypothesis, the error rates are the highest; it is curious that in some cases (with channel dependent models) the error rate can be lower even if the models are from another class. Note that models of the same class give the best recognition hypothesis most of the time; gender is the class that resulted in the highest frequency of "same" model and data.

When the 4 models are combined, we can select the recognition results in function of the

gender or channel only, or we can consider that the data and models are from the same source only if models of the same gender and the same channel give the most likely hypothesis . In Tables 5.22 and 5.23 the results are grouped looking at only one of the sources at a time, with PSTN and GSM data respectively.

Class		Task			
		`Digits`	`Numbers`	`TelNum`	`Cities`
Gender	Same	0.79 (93.43)	2.73 (96.05)	8.09 (93.54)	41.44 (89.14)
	Other	5.79 (6.57)	6.03 (3.95)	10.93 (6.46)	51.21 (10.86)
Channel	Same	1.15 (82.56)	2.76 (81.17)	8.40 (84.78)	40.95 (89.31)
	Other	0.97 (17.44)	3.31 (18.83)	7.41 (15.22)	56.87 (10.69)

Table 5.22: WER for models and data of same (and other) variability class (gender and channel), "4Mod" combined models and PSTN data.

Class		Task		
		`Digits`	`Numbers`	`TelNum`
Gender	Same	1.98 (90.07)	4.22 (92.59)	11.58 (93.83)
	Other	6.83 (9.93)	9.68 (7.41)	14.67 (6.17)
Channel	Same	2.19 (86.39)	4.58 (96.65)	11.64 (93.73)
	Other	4.20 (13.61)	5.86 (3.35)	13.65 (6.27)

Table 5.23: WER for models and data of same (and other) variability class (gender and channel), "4Mod" combined models and GSM data.

We can see that using models from different gender than the data generates the most errors (but that is less frequent); when the channel is not the same, the degradation is less severe. The exception is the `Cities` task, where the channel is more important than the gender; but remember the gender tags for this task were obtained automatically, so there are some errors in the gender tags for this task.

It is curious that for `TelNum` (PSTN), the error rate is lower for utterances that were recognized with models not from the same source of variability; this can be explained by some low SNR utterances that are more likely to be better recognized with "noisier" GSM-like models

than with PSTN models (that are trained with higher SNR utterances).

5.5.3 SNR and AVE

The results for SNR and AVE dependent models are presented separately from those with gender and channel dependent models because the data do not include the utterances that were not aligned to the models, hence a somewhat different test set. The data not taken into account in the SNR and AVE sets are those corresponding to the "Not Aligned" columns of tables 5.7 and 5.8. Tables 5.24 and 5.25 present the results for PSTN and GSM data, respectively.

Class		Task			
		`Digits`	`Numbers`	`TelNum`	`Cities`
SNR	Same	1.05 (62.13)	2.62 (66.70)	6.97 (63.04)	40.00 (56.81)
	Other	2.45 (37.87)	3.83 (33.30)	8.84 (36.96)	51.19 (43.19)
AVE	Same	1.19 (74.61)	2.65 (78.08)	7.13 (79.97)	42.89 (75.87)
	Other	1.33 (25.39)	2.91 (21.92)	7.73 (20.03)	44.65 (24.13)

Table 5.24: WER for models and data of same (and other) variability class (SNR and AVE), PSTN data.

Class		Task		
		`Digits`	`Numbers`	`TelNum`
Gender	Same	3.13 (59.29)	5.37 (60.75)	12.48 (64.42)
	Other	3.56 (40.17)	4.15 (39.25)	10.79 (35.58)
Channel	Same	3.45 (62.50)	4.56 (70.46)	12.77 (77.51)
	Other	2.78 (34.74)	4.44 (29.54)	11.12 (22.49)

Table 5.25: WER for models and data of same (and other) variability class (SNR and AVE), GSM data.

For these sources of variability (SNR and AVE), using models with same class tag as the source data does not degrades the recognition results as much as for the other sources (notably the gender); the exception is the `Cities` task and SNR dependent models, where there is a large difference in performance. It is also curious that with GSM data and the more "difficult" tasks

(2-digit numbers and telephone numbers) the performance is higher when the models that were used for recognition are from a different class than the data. The percentage of models with the largest likelihood in decoding with different class tag is larger than for the other sources, suggesting that the boundaries set for splitting the training data are not optimal. With the thresholds that were used to split the training data, the data in each chunk is less homogeneous, particularly for the lower values (either SNR or AVE). Thus the data should not be split into chunks of equal size, but into chunks that are as homogeneous as possible.

We now focus the analysis on the results on `TelNum` (both PSTN and GSM) and `Cities` tasks because much more data are available for these tasks. In Figure 5.18 the distributions of the SNR and AVE for both PSTN and GSM data are illustrated for the training data (PSTN and GSM) and the tasks of interest. In most of the cases the modes of the distributions are not far (3 – 7 dB) from the modes observed in training data, with exception to the SNR of GSM telephone numbers; but the error rates for SNR dependent models are not higher than for AVE dependent models, even if the distance between the modes of the distributions (training/test) is larger. Therefore the constraint that the distributions of data (train and test) should match (as close as possible) for high performance is not always valid.

The energy levels for the `Cities` task are higher than those observed in the training data; this is reflected in the distribution of both SNR and AVE for this task. On this task, AVE dependent models are better than SNR dependent models, even if the offset between the two distributions seems higher for the AVE than for the SNR.

Unfortunately the results do not show a correlation between the apparent "match" between the distributions of the measurements (AVE or SNR) for training and test data and performance. There are probably other factors that interact with each other, resulting into this or that performance; such interaction can not be represented by a single measurement on the signal as we tried here.

5.6 Summary

In this chapter we evaluated the use of multiple models for each unit in a speech recognition system. Those models are trained on (almost) homogeneous subsets of the training data and they are combined at three different representation levels, namely *Syntactical*, *Lexical*, and *Acoustical*, for recognition. The homogenous subsets of the training data were chosen based

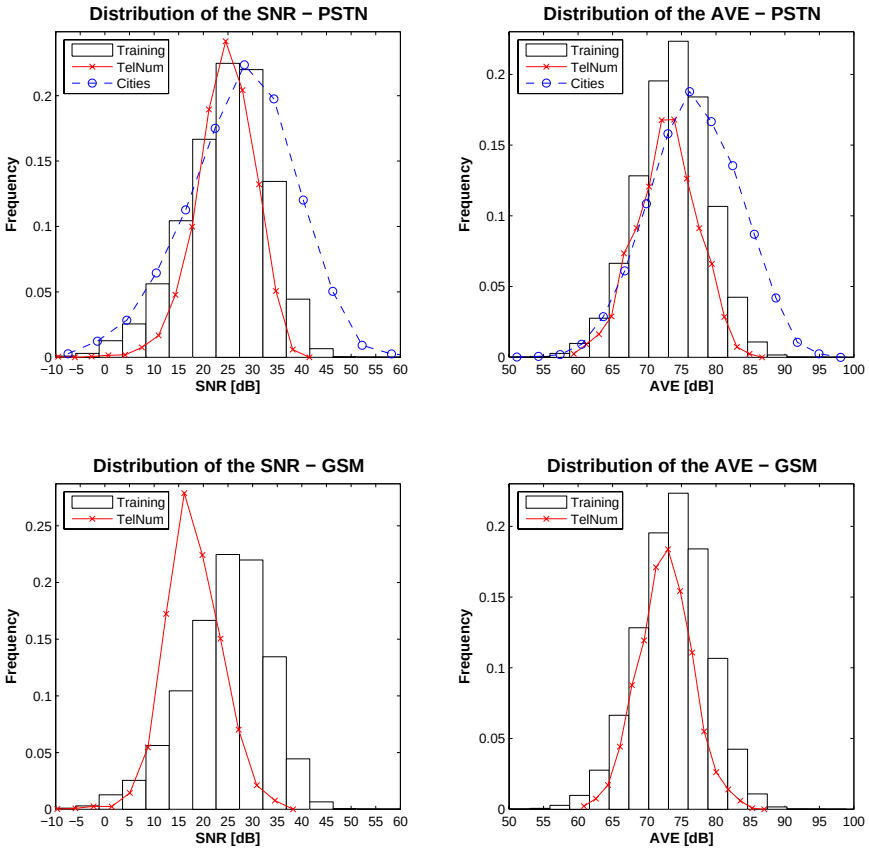


Figure 5.18: Distribution of SNR and AVE of training and test data for `TelNum` and `Cities` .

on a *a priori* choice of variability sources; the most common sources are obviously gender and transmission channel but we also trained models dependent on the signal-to-noise ratio and the average energy on the segments corresponding to vowels.

We find that models that are dependent on the gender and on the transmission channel yield the highest performance, followed by simple GD models and AVE dependent models. Gender dependent modeling is common-place in the current state-of-the-art in acoustical modeling, so there is no surprise, but it is surprising that it was not possible to improve from it with SNR or AVE dependent models for example. One explanation is that even if both measures are somewhat utterance specific, gender does not change during the time course (maybe some segments locally matches better one gender or the other) whereas SNR or AVE may change drastically and locally; for example the prosodical contour of the phrases leads to different levels of energy during the utterance. For SNR dependent models, the *Lexical* combination scheme performed slightly better than the *Syntactical* but the same behavior was not observed for AVE dependent models; but the AVE is a measure that depends much more on the speaker than the SNR (provided the Lombard effect [Junqua 1995] is overlooked), and also less affected by the

noise level.

Regarding the combination methods, the *Acoustical* scheme provided the higher performance overall on the different tasks and settings, followed by the *Syntactical* scheme. We then analyzed the effect of the higher number of Gaussian components for the silence units for the acoustically combined models; the other schemes use the same silence models (leading, trailing and short pauses) as the mixed trained models. There is a possibility that the "richer" silence model is indeed improving performance due to the best results on telephone numbers, which is the unique continuous speech task; performance on continuous speech is known to be highly dependent on the quality of the silence models. The silence models for the acoustically combined models were replaced by the models from the mixed training and only a slight degradation was observed, thus there is no bias towards the effect of silence model.

Another aspect of multiple modeling was the use of position dependent units, where a phoneme may have different models if the distribution of its duration in a particular position within a word differs significantly from the distributions in other positions. In this approach there is no need to combine the different models, as they are selected directly from the different pronunciation variations. There is a significant gain for context-independent models; for CD models the gain is smaller, mostly because the various context-specific models are already alleviating some of the variation, e.g. the contexts going from (into) periods of silence. This approach diverges from the usual tagged clustering because it relies solely on the duration to decide when it is appropriate to include many models for a given unit, instead of relying on the acoustical information represented by the Gaussian parameters. A comparison with tagged clustering would indicate which method is better from the point of view of performance; the presented approach is much faster from the computational point of view.

Chapter 6

Conclusion

Current state-of-the-art automatic speech recognition systems employ a representation of speech that is a concatenation of atomic sub-word units (phoneme-like) that model speech sounds of short duration (ranging from 30 to 100 ms). These systems also have a single model per unit which may not be able to exactly represent all the variation that the human speech presents, variations that are not only intrinsic to speech but also due to environmental (real-world) conditions. Two problems arise in the realm of acoustic modeling: the choice of atomic units and the types of models that can be used to represent those units.

In this work we investigated the acoustical modeling using longer-term units, such as syllables and multi-phone units, in place of phone-like units; these longer units are more adequate to capture long-term dependencies between the feature vectors that are used in almost all recognition systems today than the usual phoneme-like units. There are some issues to this approach that result from the large number of units, such as the explosion in the number of parameters (particularly when context-dependent units are used) and the problem of unseen units, i.e. units that are necessary to recognize a given task but were not present during model training. We presented solutions to these problems that made possible the use of context-dependent long units.

Regarding the types of models used to represent the basic units, we adhere to the usual phoneme-like units but trained different models for each one (multiple modeling) on subsets of the training data that are chosen *a priori* based on different "sources" of variability; actually the data are selected according to the possible values for each source, e.g. female and male for gender, or PSTN and GSM for the transmission channel, etc. We chose some usual sources (or classes) of variability, such as gender, transmission channel (PSTN or GSM), signal to noise ratio (SNR)

and an alternative measure, namely the average energy on vowel-like segments (AVE), and the position of the phoneme in the word. Specific modeling for gender and transmission channel is adopted by many current systems and similar modeling experiments are included to provide benchmark performance measurements. SNR in most of the works published in the literature is not measured (estimated) from the signal but is the result of adding noise with appropriate energy to the clean speech signal, resulting into artificially noisy speech. Here we estimated the SNR from the signal representing the kind of speech that an actual recognition system is likely to face within applications. The AVE is a measure that circumvents the estimation problems of the SNR by using the vowel segments that are more energetic and usually better identified than other sounds such as fricatives and stops.

We combined the resulting multiple models at three different representation levels that we named *syntactical, lexical* and *acoustical*. The higher abstraction level is the syntactical, where there is a path in the grammar for each possible value that the measurement can have for a given source (e.g. the values for the SNR are "low", "medium" and "high") and the lowest abstraction level is the acoustical one, where we simply gathered all the Gaussian mixtures of the different models into a single mixture. The lexical level is intermediary and contrary to the syntactical level, a decoding hypothesis may use models from different values of the variability source from phoneme to phoneme.

Models that depend on the phoneme position within the words are not new [Wang 1996]: we innovate in the procedure that decides where it is appropriate to add an extra model for a phoneme, given its position. The decision was based on the temporal information resulting from a forced-alignment of the training data; thus we captured some prosodical effects on the pronunciation of the words that are not likely to be captured when the decision is based on acoustical measurements only and the proposed approach required very low computational resources.

Next we list the contributions that resulted from this work.

6.1 Contributions

6.1.1 Training

Regarding the training procedure for HMMs, there are two contributions in the present work, one is an analysis of the effects of using a fixed alignment between the training data and the model's densities for faster training of Gaussian mixture models, and the other contribution is

an algorithm to initialize the parameters of mixture models from the (fixed) alignments.

Fixed Alignments

Alignment of training data and models (forced Viterbi decoding) is responsible for the major part of the time spent during model training. In this thesis we analyzed the effects of reducing the number of times that the alignment is calculated by fixing it for a few iterations and only then re-calculating the alignment; we calculated the alignment with single Gaussian models and kept it fixed for the rest of the training iterations relating to mixture models estimation. The computations reduce then to find the most likely element in the mixture aligned to a given datum (feature vector) at each time.

The time needed for each re-estimation iteration was reduced by a factor of forty (40), compared to the usual procedure. There was only a slight degradation of the performance (if any was observed), for both alternatives: re-calculating the alignments every few iterations or keeping it fixed from the mono Gaussian models.

Sequential Clustering Algorithm

Another contribution of this work is an algorithm (named Sequential Clustering Algorithm – SCA) that is used to initialize Gaussian mixtures from aligned data. It generates mixture models without the split/train iterations, greatly reducing the model production time. Moreover, all the data available for each mixture is "seen" during the initialization, what is not possible with other clustering algorithms such as k-means, due to computational complexity.

This algorithm is not very sensitive to its parameters (notably the membership distance) and was used with success in the initialization of models for PSTN and GSM networks with similar settings, and also to initialize models for long units, like syllables, without much degradation of performance.

6.1.2 Long Units

The contributions of this thesis are the contextual modeling of long units (syllables and multi-phone units), a method to deal with unseen syllables and the comparison between long units and traditional allophonic units.

Another aspect is the use of multi-phone units; the idea to use mutual information in the choice of units was first presented in [Bazzi 2001], in the context of detection of out-of-vocabulary

words. We independently arrived at the same idea, but here it is applied to selecting units in vocabulary modeling.

Multi-phone units outperformed both syllables and allophonic units; one of the advantages over syllables is the smaller number of units, thus parameter estimation is more robust. This advantage is reflected in the experiments with adapted models, where the gain over allophonic units was even higher; we believe that with more training data (more or less represented by the adaptation experiments) longer units should yield even higher performance compared to allophonic units.

The "average occurrence" measure, that takes into account the number of occurrences for each long unit model in the training data, proved to be useful in breaking down unseen syllables into smaller units that have trained models and then selecting the most appropriate in terms of amount of data that was used to train the models. To avoid always backing-off to a phone decomposition, we fixed a threshold on the contribution of the number of training samples for these alternative decompositions to an experimentally determined value which minimized the number of recognition errors.

6.1.3 Multi-Modeling

The main contributions on this topic are the experimental comparison of many sources of variability and the comparison of three model combination schemes.

Performance Analysis for many Variability Sources

We analyzed the performance of models using different variability sources. The sources are speaker gender, transmission channel, SNR and AVE. The latter AVE is a measure that we introduced to be more robust than SNR because the detection of vowels is more reliable than for other classes of phonemes and also because the detection of silence periods is known to be unreliable in some situations, e.g. noisy conditions.

Gender dependent (GD) modeling resulted to be the best compromise between performance and model complexity, followed by models dependent on the AVE; GD models are quite common and the results just confirmed that gender is indeed an important source of variability in speech. More work has to be done on AVE-dependent modeling, regarding the choice of the threshold used to split the training data in function of the value of the AVE.

Model Combination

We combined models at three abstraction levels (from higher to lower): syntactical, lexical and acoustical.

It is interesting to note that the acoustically combined models performed better in general that the other combination schemes. This reinforces the belief that detailed models (many components) are needed for high performance, but it also shows that it is possible to add some external knowledge in the choice of data used to estimate parameters, in order to optimize the training of the parameters; otherwise, these models would not outperform the reference models trained on the whole training database. So the answer to the question posed in Chapter 5 is that it is better to have a single but detailed model for each unit than trying to combine multiple models at a higher abstraction level (syntactical or lexical).

One hypothesis to be further pursued is to enlarge the search beam for combined models to cope with the larger number of decoding hypotheses to assess if a tight beam is not pruning too much for this type of modeling.

Multi-Modeling According to Phoneme Position

Position dependent models can be obtained either by adding extra models everywhere or by tagged clustering [Reichl 1999, Paul 1997]. By using the temporal information it is possible to capture prosodical influence on the realization of the phonemes at different positions, what is not possible on the acoustics alone.

Using this method we could rapidly decide where it is appropriate to create extra models for a given phoneme at different positions within a word. There was no significant gain with context-dependent models, but if context-independent models have to be used, there is a lot to gain using position-dependent models.

6.2 Future Directions

We present here some ideas that complete and expand the present work.

6.2.1 Different Topologies for Contextual Factorization

In the topology used, the contextual states are linked to a single leftmost (rightmost) internal states. It would be interesting to add multiple internal states, e.g. one for each broad phonetic

class (plosives, stops, etc), then link those (internal) states with the corresponding left/right contextual states. While this approach increases the number of parameters, it provides more detailed modeling which should improve the representation of the transitional parts.

6.2.2 Front- and Back-ends

It is not clear how HMMs deal with long units without a dedicate duration model. Our experiments with duration-oriented topologies did not show that they are superior to a simple concatenation of phone models; moreover, the results presented in the literature did not show a consistent advantage of duration models. Another type of front-end should be devised to deal with units that span different time periods, i.e. windows of different lengths are used depending on the characteristics of the signal and the units that are modeled; this implies finding an appropriate back-end that models those features.

Frame-based feature extraction may be not the most appropriate front-end; examination of the results showed that many errors like the pair of words *Paris* and *Arry* happen because the distinguishing feature between the words is not detected. The stop release of the phoneme /p/ is missed partly because it fits into the time slot of one observation vector and its time of occurrence is lost; a simple detector could determine the times for the releases and then the corresponding hypothesis could be boosted during decoding, expanding the work presented in [Bartkova 1999] (prosody-based post-processing).

Keeping the current couple MFCC&HMM, one proposition is to build many "feature detectors" which run in parallel to the usual front-end and then a rule-based and expert designed block re-scores the different hypotheses based on the detected features . The features may be on different time scales (with different frame-rates) and may be as simple as the detection of energy peaks or its direction of change. The idea is not to combine the probability scores of the various detectors, but to penalize the competitive paths given what was detected in the speech signal.

The same goal can be attained by re-scoring multiple decoding hypotheses (N-best list or word graph) with the expert block and a list of confusions that can be made if one or more features are not "detected" with usual modeling/decoding approaches. The detection of the features should be reliable enough to reduce the likelihood of introducing more errors.

This approach is similar to having multiple heterogeneous models that are specific for certain measurements on the speech signal that should help to distinguish between phenomena that

cannot be fully (or in an adequate way) distinguished with the usual MFCC/HMM paradigm.

6.2.3 Data Selection and Redundancy

It is clear from the analysis of the distribution of SNR and AVE of training and test data that models recognize well data that are similar to data seen during training.

What we want to stress here is that data should be carefully chosen during training and techniques similar to Active Learning [Cohn 1996, Hasenjäger 2000] should be pursued, not only to reduce the amount of data to be manually labeled [Kamm 2004], but also to reduce the redundancy of training data.

We looked at the parameters of the mixtures and we saw that there are many components that have a large mass; as the dynamic range of mixture weights and Gaussian scores is quite different, those components (with large weight) are frequently chosen as the "best" (most likely). This happens even if the large-mass components are farther from the observation vector than the nearest component. We propose to use some form of "conscience" (similarly to what has been done in Kohonen self-organizing map [Sieno 1988]) to avoid those "frequent winner" components, increasing the efficiency of the modeling, by better exploitation of the parameters.

6.2.4 Local vs. Global Measurements

AVE-dependent models showed good results, better than SNR dependent models. What can be criticized about these measures is that they are utterance-wide measures, but they can vary locally and the global measurement is just an average. As beam-search decoding (in our system and many others) is a synchronous process, it is not possible to recover if the correct hypothesis is removed (which can happen early in the case of a poor local match).

One way to deal with the local variation is to combine the models at another intermediate level between syntactical and acoustical and that is more restrictive than the lexical scheme. The idea (coming from VTLN [Lee 1996]) is to increase the number of possible instances and perform local measurements of the source of variability and based on its value, restrict the possible paths, as illustrated in Figure 6.1 in the case of five possible values for the source and the French word *sept*. A local measurement is made at a given time instant giving the value "1", thus the paths for values "3", "4" and "5" are disabled in the concurrent decoding hypothesis. Path "2" remains active as a second-opinion.

Proceeding in this way makes it is possible to reduce the alternative paths and also to "track"

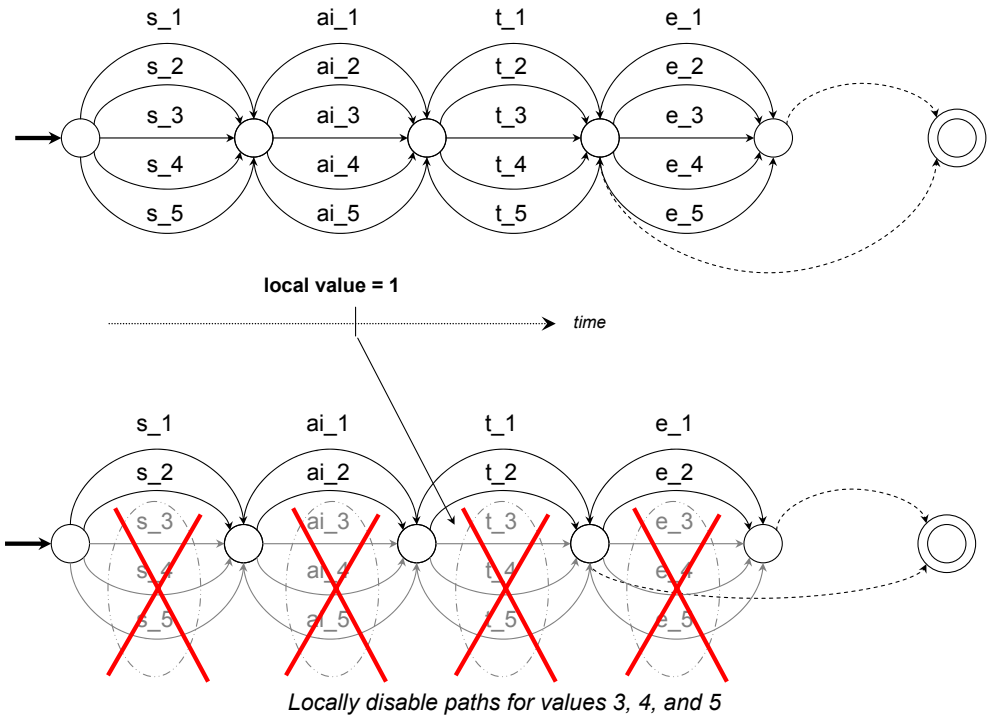


Figure 6.1: Alternative lexical level combination.

local changes in the course of decoding.

Bibliography

[Ariki 1994] Ariki Y. and Doi K., Phoneme recognition improvement by restricting training section in concatenated HMM training, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 253–257, Adelaide, 1994.

[Atal 1983] Atal B., Efficient coding of LPC parameters by temporal decomposition, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 81–84, Boston, 1983.

[Bahl 1991] Bahl L. R., de Souza P. V., Gopalakrishnan P. S., Nahamoo D. and Picheny M. A., Decision trees for phonological rules in continuous speech, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 185–188, Ontario, 1991.

[Bakis 1976] Bakis R., Continuous speech recognition via centisecond acoustic states, in *91st Meeting of the Acoustical Society of America*, 1976.

[Barrault 2005] Barrault L., Mori R. D., Gemello R., Mana F. and Matrouf D., Variability of automatic speech recognition systems using different features, in *Proceedings of the International Conference on Spoken Language Processing*, Lisbon, 2005.

[Bartkova 1991] Bartkova K. and Jouvet D., Modelization of allophones in a speech recognition system, in *Proceedings of the XII International Congress on Phonetical Sciences*, volume 4, pages 474–477, Aix-en-Provence, 1991.

[Bartkova 1999] Bartkova K. and Jouvet D., Selective prosodic post-processing for improving recognition of french telephone numbers, in *Proceedings of the European Conference on Speech Communication and Technology*, 1999.

[Basseville 1989] Basseville M., Distance measures for signal processing and pattern recognition, *Signal Processing*, 18:249–369, 1989.

[Bazzi 2001] Bazzi I. and Glass J., Learning units for domain-independent out-of-vocabulary word modeling, in *Proceedings of the 7th. European Conference on Speech Communication and Technology*, volume 1, pages 61–64, Aalborg, 2001.

[Bilmes 1998] Bilmes J. A., A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models, Technical Report TR-97-021, International Computer Science Institute, 1998.

[Boll 1979] Boll S. F., Suppression of acoustic noise in speech using spectral subtraction, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-27:113–120, 1979.

[Brown 1991] Brown M. K. and Wilpon J. G., A grammar compiler for connected speech recognition, *IEEE Transactions on Signal Processing*, 39(1):17–28, 1991.

[Brugnara 1991] Brugnara F. et al., A Parallel HMM Approach to Speech Recognition, in *Proceedings of the European Conference on Speech Communication and Technology*, pages 1103–1106, 1991.

[Brugnara 1992] Brugnara F. et al., A family of parallel hidden markov models, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 377–380, 1992.

[Cohn 1996] Cohn D., Ghahramani Z. and Jordan M., Active learning with statistical models, *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[Cole 1997] Cole R. et al., Alphadigit corpus, OGI, 1997, URL `http://www.cse.ogi.edu/CSLU/corpora/alphadigit`.

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

[Deligne 1997a] Deligne S. and Bimbot F., Inference of variable-length acoustic units for continuous speech recognition, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 1731–1734, Munich, 1997.

[Deligne 1997b] Deligne S. and Bimbot F., Inference of variable-length linguistic and acoustic units by multigrams, *Speech Communication*, 23(3):223–241, 1997.

[Delphin-Poulat 2001] Delphin-Poulat L., Comparison of techniques for environment/application adaptation in a telephony context, in *Adaptation Methods for Speech Recognition*, ISCA Tutorial and Research Workshop, pages 139–142, Sophia Antipolis, France, 2001.

[Dempster 1977] Dempster A. P., Laird N. M. and Rubin D. B., Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[Deshmukh 1999] Deshmukh N., Ganapathiraju A. and Picone J., Hierarchical search for large vocabulary conversational speech recognition, *IEEE Signal Processing Magazine*, 16(5):84–107, 1999.

[Duda 2000] Duda R. O., Hart P. E. and Stork D. G., *Pattern Classification*, Wiley-Interscience, 2 edition, 2000.

[Ephraïm 1984] Ephraïm Y. and Malah M., Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-32(6):1109–1121, 1984.

[Finke 1997] Finke M. and Rogina I., Wide context acoustic modeling in read vs. spontaneous speech, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, pages 1743–1746, Munich, Germany, 1997.

[Fiscus 1997] Fiscus J. G., A Post-Processing System to Yield Reduced Word Error Rates: Recogniser Output Voting Error Reduction (ROVER), in *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding*, pages 347–352, 1997.

[Fujimura 1975] Fujimura O., Syllable as a Unit of Speech Recognition, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-23(1):82–87, 1975.

[Furui 1981] Furui S., Cepstral analysis technique for automatic speaker verification, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29:254–272, 1981.

[Furui 1986] Furui S., On the role of spectral transition for speech recognition, *The Journal of the Acoustical Society of America*, 80(4), 1986.

[G. David Forney 1973] G. David Forney J., The Viterbi algorithm, *Proceedings of the IEEE*, 61(3):268–278, 1973.

[Gales 1996] Gales M., The generation and use of regression class trees for MLLR adaptation, Technical report, Cambridge University Engineering Department, 1996.

[Gales 1997] Gales M., Maximum likelihood linear transformations for HMM-based speech recognition, Technical report, Cambridge University Engineering Department, 1997.

[Ganapathiraju 2001] Ganapathiraju A. et al., Syllable-based large vocabulary continuous speech recognition, *IEEE Transactions on Speech and Audio Processing*, 9(4):358–366, 2001.

[Gauvain 1986] Gauvain J.-L., A syllable-based isolated word recognition experiment, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, pages 57–60, Tokyo, 1986.

[Gauvain 2003] Gauvain J.-L., Lamel L., Schwenk H., Adda G., Chen L. and Lefevre F., Conversational telephone speech recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 212–215, Hong Kong, 2003.

[Godfrey 1992] Godfrey J., Holliman E. and McDaniel J., SWITCHBOARD: Telephone Speech Corpus for Research and Development, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 517–520, 1992.

[Goldenthal 1994] Goldenthal W., *Statistical Trajectory Models for Phonetic Recognition*, Ph.d. thesis, M.I.T., 1994.

[Gong 1995] Gong Y., Speech recognition in noisy environments: a survey, *Speech Communication*, 16(3):261–291, 1995.

[Gopalakrishnan 1991] Gopalakrishnan P., Kanevsky D., Nadas A. and Nahamoo D., An inequality for rational functions with applications to some statistical estimation problems, *IEEE Transactions on Information Theory*, 37:107–113, 1991.

[Grammont 1965] Grammont M., *Traité de Phonétique*, Librairie Delagrave, Paris, France, 1965.

[Greenberg 1999] Greenberg S., Speaking in shorthand – A syllable-centric perspective for understanding pronunciation variation, *Speech Communications*, 29:159–176, 1999.

[Hain 2001] Hain T., *Hidden Model Sequence Models for Automatic Speech Recognition*, Ph.d. thesis, University of Cambridge, 2001.

[Hain 1999] Hain T. and Woodlan P. C., Dynamic HMM selection for Continuous Speech Recognition, in *Proceedings of the European Conference on Speech Communication and Technology*, volume 3, pages 1327–1330, 1999.

[Hansen 2001] Hansen M. H. and Yu B., Model selection and the principle of minimum description length, *Journal of the American Statistical Association*, 96(454):746–774, 2001.

[Hartley 1958] Hartley H., Maximum likelihood estimation from incomplete data, *Biometrics*, 14:174–194, 1958.

[Hasenjäger 2000] Hasenjäger M., *Active data selection in supervised and unsupervised learning*, Ph.d., Universität Bielefeld, 2000.

[Hermansky 1990] Hermansky H., Perceptual linear predictive (PLP) analysis of speech, *Journal of the Acoustical Society of America*, 87:1738–1752, 1990.

[Hirsch 1995] Hirsch H. and Ehrlicher C., Noise estimation techniques for robust speech recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 153–156, 1995.

[Hirsch 2000] Hirsch H. G. and Pearce D., The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions, in *Proceedings of the ISCA Tutorial and Research Workshop on Automatic Speech Recognition*, 2000.

[Hopcroft 2001] Hopcroft J., Motwani R. and D.Ullman J., *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, 2nd edition, 2001.

[Hu 1996] Hu Z., Schalkwyk J., Barnard E. and Cole R., Speech recognition using syllable-like units, in *Proceedings of the International Conference on Spoken Language Processing*, volume 2, pages 1117–1120, 1996.

[Hunt 1988] Hunt M., Evaluating the performance of connected word speech recognition systems, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, pages 457–460, 1988.

[J. P. Openshaw 1994] J. P. Openshaw J. S. M., On the limitations of cepstral features in noise, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 49–53, 1994.

[Jelinek 1997] Jelinek F., *Statistical Methods for Speech Recognition*, MIT Press, 1997.

[Jiménez 1995] Jiménez V. M., Marzan A. and Monné J., A Comparison of two exact algorithms for finding the N-best sentence hypothesis in continuous speech recognition, in *Proceedings of the European Conference on Speech Communication and Technology*, pages 1071–1074, Madrid, 1995.

[Jordan 2004] Jordan M. I., Graphical models, *Statistical Science (Special Issue on Bayesian Statistics)*, 19:140–155, 2004.

[Jouvet 1994] Jouvet D., Bartkova K. and Stouff A., Structure of allophonic models and reliable estimation of the contextual parameters, in *Proceedings of the International Conference on Spoken Language Processing*, volume 1, pages 283–286, Yokohama, 1994.

[Juang 1997] Juang B., Chou W. and Lee C., Minimum classification error rate method for speech recognition, *IEEE Transactions on Speech and Audio Processing*, 5, 1997.

[Junqua 1995] Junqua J.-C. and Haton J.-P., *Robustness in Automatic Speech Recognition: Fundamentals and Applications*, Kluwer Academic Publishers, 1995.

[Jurafsky 2001] Jurafsky D., Ward W., Jianping Z., Herold K., Xiuyang Y. and Sen Z., What kind of pronunciation variation is hard for triphones to model?, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 577–580, Salt Lake City, 2001.

[Kamm 2004] Kamm T., *Active Learning for Acoustic Speech Recognition Modeling*, Ph.d., Johns Hopkins University, 2004.

[Kapadia 1998] Kapadia S., *Discriminative Training of Hidden Markov Models*, Ph.d., Cambridge University, 1998.

[Knuth 1997] Knuth D. E., *The Art of Computer Programming*, volume 2: Seminumerical algorithms, Addison-Wesley, Reading, Massachusetts, 1997.

[Korkmazskiy 1997] Korkmazskiy F., Juang B.-H. and Soong F., Generalized mixture of HMMs for continous speech recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, pages 1443–1446, 1997.

[Lauritzen 1996] Lauritzen S. L., *Graphical Models*, Oxford University Press, 1996.

[Lee 1996] Lee L. and Rose R. C., Speaker normalization using efficient frequency warping procedures, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 353–356, 1996.

[Leonard 1984] Leonard R. G., A database for speaker independent digit recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 42.11.1–4, 1984.

[Ljolje 2000] Ljolje A., Hindle D. M., Riley M. D. and Sproat R. W., The AT&T LVCSR-2000 System, in *NIST 2000 Speech Transcription Workshop*, 2000.

[Macias-Guarasa 2003] Macias-Guarasa J., Ordonez J., Montero J., Ferreiros J., Cordoba R. and D'Haro L., Revisiting scenarios and methods for variable frame rate analysis in automatic speech recognition, in *Proceedings of the European Conference on Speech Communication and Technology*, pages 1809–1812, 2003.

[Makhoul 1975] Makhoul J., Linear prediction: A tutorial review, *Proceedings of the IEEE*, 63(4):561–580, 1975.

[Markov 2003] Markov K. et al., Noise and Channel Distortion Robust ASR System for DARPA SPINE2 Task, *IEICE Transactions of Information & Systems*, E86-D(3):497–504, 2003.

[Matsuda 2004] Matsuda S. et al., Speech recognition system robust to noise and speaking styles, in *Proceedings of the International Conference on Spoken Language Processing*, pages 844–847, 2004.

[Matsumura 1994] Matsumura T. and Matsunaga S., Towards non-uniform unit HMMs for speech recognition, in *Proceedings of the 2nd. IEEE Workshop on Interactive Voice Technology for Telecommunications Applications*, pages 89–92, Kyoto, 1994.

[Matsumura 1995a] Matsumura T. and Matsunaga S., Non-uniform unit based HMMs for continuous speech recognition, *Speech Communication*, 17(3-4):321–329, 1995.

[Matsumura 1995b] Matsumura T. and Matsunaga S., Non-uniform unit HMMs for speech recognition, in *Proceedings of the 4th. European Conference on Speech Communication and Technology*, volume 1, pages 499–502, Madrid, 1995.

[Matsunaga 1995] Matsunaga S., Matsumura T. and Singer H., Continuous speech recognition using non-uniform unit based acoustic and language models, in *Proceedings of the 4th. European Conference on Speech Communication and Technology*, volume 3, pages 1619–1622, Madrid, 1995.

[Meneido 2000] Meneido H. and ao P. Neto J., Combination of acoustic models in continuous speech recognition hybrid systems, in *Proceedings of the International Conference on Spoken Language Processing*, Beijing, 2000.

[Messina 2004a] Messina R. and Jouvet D., Context dependent "long units" for speech recognition, in *Proceedings of the International Conference on Spoken Language Processing*, page TuC2101p.1, Jeju Island, Korea, 2004.

[Messina 2004b] Messina R. and Jouvet D., Sequential clustering algorithm for gaussian mixture initialization, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 833–836, 2004.

[Mirghafori 1996] Mirghafori N., Fosler E. and Morgan N., Towards Robustness to Fast Speech in ASR, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, pages 335–338, 1996.

[Morgan 1993] Morgan N. and Bourlard H., *Connectionist Speech Recognition: A Hybrid Approach*, Kluwer, 1993.

[Murphy 2002] Murphy K., *Dynamic Bayesian Networks: Representation, Inference and Learning*, Ph.d., University of California, Berkeley, 2002.

[Nanjo 2002] Nanjo H. and Kawahara T., Speaking-rate dependent decoding and adaptation for spontaneous lecture speech recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 725–728, 2002.

[Neti 1997] Neti C. and Roukos S., Phone-specific gender-dependent models for continuous speech recognition, in *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding*, Santa Barbara, USA, 1997.

[Ney 2000] Ney H. and Ortmanns S., Progress in Dynamic Programming Search for LVSCR, *Proceedings of the IEEE*, 88(8), 2000.

[Nock 1997] Nock H. J., Gales M. J. F. and Young S. J., A comparative study of methods for phonetic decision-tree state clustering, in *Proceedings of the 5th European Conference on Speech Communication and Technology*, volume 1, pages 111–114, Rhodes, 1997.

[Noetzel 1991] Noetzel A., Robust syllable segmentation of continuous speech using neural networks, in *Proceedings of the IEEE Electro International Conference Record*, pages 580–585, New York, 1991.

[O'Neill 1998] O'Neill P., Vaseghi S., Doherty B., Tan W. H. and McCout P., Multi-phone strings as subword units for speech recognition, in *Proceedings of the International Conference on Spoken Language Processing*, number 178, Sydney, 1998.

[Ostendorf 1997] Ostendorf M. and Singer H., HMM Topology Design using Maximum Likelihood Successive State Splitting, *Computer, Speech and Language*, 11(1):17–42, 1997.

[Paul 1997] Paul D. B., Extensions to phone-state decision-tree clustering: Single tree and tagged clustering, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 1487–1490, Munich, 1997.

[Pfau 1997] Pfau T., Beham M., Reichl W. and Ruske G., Creating large subword units for speech recognition, in *Proceedings of the 5th. European Conference on Speech Communication and Technology*, volume 3, pages 1191–1194, Rhodes, 1997.

[Pfau 1998] Pfau T. and Ruske G., Creating hidden markov models for fast speech, in *Proceedings of the International Conference on Spoken Language Processing*, pages 255–258, 1998.

[Pfitzinger 1996] Pfitzinger H., Burger S. and Heid S., Syllable detection in read and spontaneous speech, in *Proceedings of the International Conference on Spoken Language Processing*, pages 1261–1264, Philadelphia, 1996.

[Picone 1990] Picone J., Continuous speech recognition using hidden markov models, *IEEE Acoustics, Speech, and Signal Processing Magazine*, pages 26–41, 1990.

[Plannerer 1993] Plannerer B. and Ruske G., A continuous speech recognition system using phonotactic constraints, in *Proceedings of the European Conference on Speech Communication and Technology*, volume 2, pages 859–862, Berlin, 1993.

[Price 1988] Price P., Fisher W. M., Bernstein J. and Pallett D. S., The DARPA 1000-word resource management database for continuous speech recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 651–654, New York, USA, 1988.

[Rabiner 1993] Rabiner L. R. and Juang B.-H., *Fundamentals of Speech Recognition*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[Rabiner 1989] Rabiner L. R., Juang B. H. and Wilpon J. G., HMM Clustering for Connected Word Recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 405–408, 1989.

[Rabiner 1986] Rabiner L. R., Wilpon J. G. and Juang B.-H., A segmental k-means training procedure for connected word recognition, *AT&T Tech. Journal*, 64(3):21–40, 1986.

[Reichl 1999] Reichl W. and Chou W., A unified approach of incorporating general features in decision tree based acoustic modeling, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 2377–2380, Phoenix, 1999.

[Rissanen 1978] Rissanen J., Modeling by shortest data descriptions, *Automatica*, 14:465–471, 1978.

[Rosenberg 1994] Rosenberg A., Lee C.-H. and Soong F., Cepstral channel normalization techniques for HMM-based speaker verification, in *Proceedings of the International Conference on Spoken Language Processing*, pages 1835–1838, Yokohama, 1994.

[Sankar 1998] Sankar A., Experiments with a Gaussian Merging-Splitting Algorithm for HMM Training for Speech Recognition, in *Proceedings of the DARPA workshop*, pages 99–104, 1998.

[Schukat-Talamazzini 1993] Schukat-Talamazzini E. G. and Niemann H., ISADORA - a Speech Modelling Network Based on Hidden Markov Models, *Computer, Speech and Language*, 1993.

[Schukat-Talamazzini 1992] Schukat-Talamazzini E. G., Niemann H., Eckert W., Kuhn T. and Rieck S., Acoustic modelling of subword units in the ISADORA speech recognizer, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–577–I–580, San Francisco, 1992.

[Schultz 1998] Schultz T. and Waibel A., Adaptation of pronunciation dictionaries for recognition of unseen languages, in *Proceedings of the workshop on Speech and Communication (SPECOM)*, pages 207–210, St.Petersburg, Russia, 1998.

[Schwartz 1985] Schwartz R., Chow Y., Kimball O., Roucos S., Krasner M. and Makhoul J., Context-dependent modeling for acoustic-phonetic recognition of continuous speech, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 1205–1208, Tampa, 1985.

[Schwartz 1990] Schwartz R. and Chow Y.-L., The N-best algorithm: An efficient and exact procedure for finding the N most likely sentence hypothesis, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 81–84, Albuquerque, 1990.

[Schwarz 1978] Schwarz G., Estimating the dimension of a model, *The Annals of Statistics*, 6(2):461–464, 1978.

[Sieno 1988] Sieno D. D., Adding a conscience to competitive learning, in *Proceedings of the IEEE International Conference on Neural Networks*, volume 1, pages 117–124, 1988.

[Song 1998] Song M. G. et al., Speech recognition in car noise environments using multiple models according to noise masking levels, in *Proceedings of the International Conference on Spoken Language Processing*, pages 1065–1068, 1998.

[Stadermann 2001] Stadermann J., Stahl V. and Rose G., Voice activity detection in noisy environments, in *Proceedings of the European Conference on Speech Communication and Technology*, pages 1851–1854, 2001.

[Takahashi 1995] Takahashi S. and Sagayama S., Four-level tied-structure for efficient representation of acoustic modeling, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 520–523, Detroit, 1995.

[Tchorz 2003] Tchorz J. and Kollmeier B., SNR Estimation Based on Amplitude Modulation Analysis With Applications to Noise Suppression, *IEEE Transactions on Speech and Audio Processing*, 11(3), 2003.

[Tomokiyo 2000] Tomokiyo L. M., Handling Non-native Speech in LVCSR: A Preliminary Study, in *Incorporating Speech Technology in Language Learning (InSTIL)*, 2000.

[Tukey 1962] Tukey J. W., Bogert B. P. and Healy M. J. R., *The frequency analysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and shape-cracking*, chapter 15, pages 209–243, Proceedings of the Symposium on Time Series Analysis, Wiley, 1962.

[Utsuro 2002] Utsuro T. et al., A Confidence Measure Based on Agreement Between Multiple LVCSR Models – Correlation Between Pair of Acoustic Models and Confidence – , in *Proceedings of the International Conference on Spoken Language Processing*, pages 701–704, 2002.

[Utsuro 2004] Utsuro T. et al., An Empirical Study on Multiple LVCSR Model Combination by Machine Learning, in *HLT/NAACL*, pages 13–16, 2004.

[Valtchev 1995] Valtchev V., *Discriminative methods in HMM-based speech recognition*, Ph.d., Cambridge University, 1995.

[Wang 1996] Wang X., Pols L. C. W. and ten Bosch L. F. M., Analysis of context-dependent segmental duration for automatic speech recognition, in *Proceedings of the International Conference on Spoken Language Processing*, 1996.

[Wegmann 1999] Wegmann S., Zhan P. and Gillick L., Progress in broadcast news transcription at dragon systems, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 33–36, Phoenix, USA, 1999.

[Witt 1999] Witt S. M., *Use of speech recognition in computer-assisted language laerning*, Ph.d., University of Cambridge, 1999.

[Wrede 2001] Wrede B., Fink G. A. and Sagerer G., An investigation of modelling aspects for rate-dependent speech recognition, in *Proceedings of the European Conference on Speech Communication and Technology*, 2001.

[Xu 1996] Xu L. and Jordan M., On convergence properties of the EM algorithm for Gaussian mixtures, *Neural Computation*, 8:129–151, 1996.

[Young 1992] Young S. J., The general use of tying in phoneme-based HMM speech recognisers, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 569–572, San Francisco, 1992.

[Young 1994] Young S. J., Odell J. J. and Woodland P. C., Tree-based state tying for high accuracy acoustic modelling, in *ARPA Workshop on Human Language Technology*, pages 286–291, 1994.

[Yu 1995] Yu H.-J. and Oh Y.-H., A neural network using non-uniform units for continuous speech recognition, in *Proceedings of the 4th. European Conference on Speech Communication and Technology*, volume 3, pages 1677–1680, Madrid, 1995.

[Yu 1996] Yu H.-J. and Oh Y.-H., A neural network using acoustic sub-word units for continuous speech recognition, in *Proceedings of the International Conference on Spoken Language Processing*, pages 506–509, Philadelphia, 1996.

[Yu 1997] Yu H.-J. and Oh Y.-H., A neural network for 500 vocabulary word spotting using acoustic sub-word units, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 3277–3280, Munich, 1997.

[Zhan 1997] Zhan P. and Waibel A., Vocal tract length normalization for large vocabulary continuous speech recognition, technical report CMU-CS-97-148, Carnegie Melon University, 1997.

[Zhang 2004] Zhang Z., Sugimura T. and Furui S., A Tree-structured Clustering Method Integrating Noise and SNR for Piecewise Linear-Transformation-Based Noise Adaptation, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, volume 1, pages 981–984, 2004.

[Zheng 2000] Zheng J., Franco H. and Stolcke A., Rate-dependent acoustic modeling for large vocabulary conversational speech recognition, in *NIST Speech Transcription Workshop*, 2000.

[Zhong 1999] Zhong L., Shi Y. and Liu R., A dynamic neural network for syllable recognition, in *Proceedings of the International Joint Conference on Neural Networks*, pages 2997–3000, Washington - DC, 1999.

[Zhu 2000] Zhu Q. and Alwan A., On the use of variable frame rate analysis in speech recognition, in *Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing*, 2000.