Lehrstuhl für Mensch-Maschine-Kommunikation
Technische Universität München

Confidence Measurement Techniques in Automatic Speech Recognition and Dialog Management

Tibor Fabian

Vollständiger Abdruck der
von der Fakultät für Elektrotechnik und Informationstechnik
der Technischen Universität München
zur Erlangung des akademischen Grades
eines **Doktor-Ingenieurs**
genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr.-Ing. Klaus Diepold

Prüfer der Dissertation:

1. apl. Prof. Dr.-Ing., Dr.-Ing. habil. Günther Ruske
2. Univ.-Prof. Dr.-Ing. habil. Rüdiger Hoffmann,
   Technische Universität Dresden

Die Dissertation wurde am 22.11.2007 bei der
Technischen Universität München eingereicht und durch die
Fakultät für Elektrotechnik und Informationstechnik
am 17.04.2008 angenommen.

Danksagung

Mein besonderer Dank gilt meinem Doktorvater, Prof. Günther Ruske, der meine Begeisterung für die automatische Spracherkennung geweckt hat und die Entstehung dieser Arbeit ermöglicht und gefördert hat. Ich möchte mich auch herzlich bei Matthias Thomae bedanken für die kreative Zusammenarbeit in der Forschungsgruppe Speech am Lehrstuhl für Mensch-Maschine-Kommunikation der Technischen Universität München.

Für ihre unermüdliche Geduld und für die liebevolle Unterstützung danke ich ganz besonders meiner Frau Agnes.

Tibor Fabian

München, Juni 2008

Abstract

Reliable confidence measures are essential to the basis of decisionmaking for enriching human-machine speech interaction with necessary intelligence in ergonomic dialog management. In addition to a survey of the state of the art in confidence measurement, this work also provides classification of methods derivated from several points of view and describes possible fields of application. The thesis includes comparative evaluation results of different computation algorithms which apply posterior probability as the hypothesis confidence measure in HMM-based speech recognition.

The key contribution of the dissertation is the description of several utilization techniques that rely on confidence measurement and are intended to enhance the performance of speech recognition systems. A new confidence-guided approach is presented to control the pruning of the Viterbi search process dynamically by taking variable search quality into consideration to fit time-variant requirements. The thesis explores dialog management strategies and several aspects of improving user acceptance in speech-based applications by the use of confidence measurement.

Contents

Glossary

ABNF	Augmented Backus-Naur Form
ACD	Adaptive Control Dynamic (Pruning)
ASR	Automatic Speech Recognition
CER	Confidence Error Rate
CFG	Context-free Grammars
CGD	Confidence-guided Dynamic (Pruning)
CM	Confidence Measurement
EER	Equal Error Rate
HMI	Human-machine Interafce
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
NLL	Normalized Log Likelihood
NLU	Natural Language Understanding
NN	Neural Network
ODINS	One-stage Decoder for Interpretation of Natural Speech
OOG	Out-of-grammar
OOV	Out-of-vocabulary
ROC	Receiver Operating Characteristic
UV	Utterance Verification
VUI	Voice User Interface
VoiceXML	Voice Extensible Markup Language
W3C	World Wide Web Consortium
WER	Word Error Rate
WGD	Word Graph Density
XML	Extensible Markup Language

List of Tables

List of Figures

Chapter 1

Introduction

Speech-based human-machine communication has made substantial gains in popularity over the last decade. Aside from other communication channels, e.g. visual or haptic, speech has enormous potential to fulfill high expectations of ergonomic human-machine interfaces (HMI). Speech is qualified to be the best choice for controlling machines by means of commands, not only for professionals to free their hands while performing other tasks at the same time, but also in the commercial sector to implement more ergonomic user interfaces for the ever-increasing complexity of household devices. Millions of people use speech interfaces every day to control their computers, mobile phones, navigation systems and also voice portals to settle their affairs with service companies or to manage bank accounts, to mention a few examples. Innumerable speech applications have been already developed for many branches of business, particularly in the telecommunication sector. There is a clear tendency that speech applications are in the process of becoming truly ubiquitous.

There are two primary supporting forces that keep this process moving forward: the natural method of speech communication for humans and the unquenchable demand of companies to automate services. However, speech communication among humans is a very complex process, hence its realization between human and machine is a real challenge. This is because speech communication operates on different levels of information transfer, i.e. acoustic-phonetic, syntactic and semantic levels, and only the perfect interplay of those levels makes it spontaneous, just as humans interact by nature.

State-of-the-art speech applications are not yet able to achieve that level of perfection. In fact, there remains a long stretch before all difficulties are sorted out concerning human-machine speech interaction. Humans, for example, master speech communication under very noisy circumstances since they can readily determine the confidence in their speech understanding under different conditions. One of the main pillars of spontaneous speech dialog among humans is to be able to confirm speech content during a dialog if necessary, but only if necessary. Otherwise the dialog flow can be crippled by too many queries and make speech interaction frustrating. On the other hand too few queries or inadequate ones can cause misunderstanding or lead the dialog to a dead end.

Many speech-based interfaces support only a limited number of commands or phrases which need to be known by the user. Therefore the usage of those interfaces implies a learning phase just as with other, non speech-based HMIs. But these limitations give the user a feeling of inconvenience and uncertainty and therefore decrease the ergonomic quality of the service. Speech has huge potential to support design and implementation of HMIs at a high ergonomic level without the need to learn special commands or a limited number of phrases. Unfortunately, due to the restrictions of current automatic speech recognition (ASR) systems and performance issues, today's speech HMIs still require some extra learning for their use. Nevertheless the major target of speech-based interfaces remains unchanged: to frame speech communication as naturally as possible in the form of ergonomic dialogs. The industrial term for the knowledge of dialog flow design is voice user interface (VUI) design. This is a collection of methods based on scientific principles in language technology, linguistics and psychology, and it is also proven by industrial experience.

The primary weakness of current ASR systems is deeply-rooted in the principle of the stochastic approach of the speech decoding process: it searches for the best hypothesis of possible candidates defined in the dictionary or grammar for a given speech utterance. Therefore, it always produces a best hypothesis result even if the current speech utterance is not contained in the list of possible candidates. To overcome this problem the grammar is often extended with what are known as *out-of-vocabulary* (OOV) phrases as a general catch-all for unrecognizable user utterances. OOV should always become the best hypothesis result if none of the grammar entries match the speech utterance. However, the quality of OOV detection suffers as the vocabulary size increases and therefore it is not always applicable for large vocabulary systems. Also confidence scores are used to rate the certainty of the recognized phrase, which is unalterable in human-machine communication. The computed confidence scores can be used to evaluate the quality of the recognition result and hence to control an interactive dialog between human and machine. The use of confidence-based decisions allows a more ergonomical and user-friendly dialog management.

VUI design implies different strategies to deal with these difficulties on the level of dialog flow control. The most important point is to link the capabilities of the underlying ASR system to the dialog strategies for a specific speech application. During the dialog design phase marginal conditions such as the following are to be clarified:

- *Scope of the application* to be covered with the dialog in order to provide access to all necessary user tasks via the speech HMI.

- *Degree of naturalness* to achieve during human-machine communication. This expectation is highly dependent on the target user group; for example, it depends on users' age and technical savvy. Users with little or no experience should be guided through the application, with only a restricted set of input choices at each dialog step to avoid confusion.

- *Robustness* to withstand interference from the users' environment such as background noise or poor audio channel quality. These disturbing factors should be warded off by appropriate error handling procedures.

- *Dialog length*, which generally should not be extended for a specific task in order to avoid a frustrating user experience and to keep the application target in focus.

- *Confirmation steps* required to fulfill other expectations like robustness or error handling. It is also important to decide what kind of confirmation step, implicit or explicit confirmation, is the best choice for a specific task. Implicit confirmation allows a more natural dialog flow but is not as robust as the explicit counterpart, which is a more restricted choice.

- Usage of *mixed-initiative* dialog forms which are able to fulfill expectations of both so-called power users and also first-time users (or those with little experience in using speech applications). This flexibility is achieved by the handling of both complex and simple user input at a specific dialog point. In the case of simple user phrases, the dialog asks for further details not yet specified with a simple user input but which are necessary for the dialog. Power users, on the other hand, are allowed to phrase all necessary information in one complex utterance.

- *Fallback strategies* which are necessary at different levels of error handling.

The successful implementation of all of these conditions requires control instruments that are able to direct the dialog flow at every dialog step depending on the content of the user input. Speech recognizers deliver the best hypothesis result for the user utterance. From the dialog's perspective, however, the information regarding the degree of accuracy of the hypothetical result is just as important as the content itself. Dialog control strategies presuppose appropriate confidence scores, whose high quality is vitally important for the satisfaction of the user.

Therefore, ASR systems must be endowed with confidence measurement techniques to allow them to compute *confidence scores* as an assessment of their confidence in the recognition result. Confidence scores can be then applied at a higher level of speech processing, i.e. in dialog control, to render the dialog flow more appropriate for human-machine communication. Aside from computation of the scores, confidence measurement techniques can also improve efficiency of speech recognition algorithms as will be shown later on in this work. Furthermore they can also be utilized for supervised training and adaptation algorithms.

1.1 Motivation, Field of Application

For the reasons mentioned above, confidence measurement (CM) is one of the main areas of current research activities concerning speech recognition. CM techniques can be arranged into two main groups depending on their operational scope within the speech recognition process. On the one hand, confidence scores can be assigned to a specific unit of the recognition result (e.g. words, phrases or sentences); on the other hand, confidence measurement approaches also have the ability to optimize algorithms within the speech decoding process in order to make it more efficient.

The difficulty in computing reliable confidence scores is anchored deeply in the fundamentals of the speech decoding process. Using Bayes' rule,

$$p(W|X) = \frac{p(X,W)}{p(X)} = \frac{p(X|W)p(W)}{p(X)}, \tag{1.1}$$

it is generally sufficient to work with relative likelihood in order to select the best recognition result by comparing the probabilities $p(X,W)$ which describe the production of a word sequence W by an acoustic observation sequence X. For the selection criteria of the best hypothesis, i.e. highest probability $p(X,W)$, the observation probability $p(X)$ in the denominator of Equation 1.1 can be omitted, since it is independent of W. Omitting $p(X)$ saves computation time but in that case hypothesis probabilities can only be used as relative values and therefore they are no longer appropriate as a confidence measure. This is because confidence measures need to imply an absolute statement regarding the certainty of a specific hypothesis and not only a comparison among alternative hypotheses.

As a result, either the decoding process is extended with additional computation of observation probability or alternative quantities of the decoding process are used to generate confidence scores. A vast body of literature has been published on the evaluation and rating of those alternative measurement techniques and their diverse combinations. However, Wessel *et al.* (2001) show that the best performance is achieved by the posterior probability based confidence measurement which uses the observation probability as the basis of confidence score computation. Especially if the word unit boundaries of similar alternative hypotheses are considered, very high CM quality can be achieved. For this reason, Wessel *et al.* (2001) carry out the computation of $p(X)$ on the *word graph* as a post-processing step to the decoding process.

Another main field of application of confidence measurement techniques is for improving the decoding process itself, which means that further optimization of ASR is possible through use of appropriate CM methods. This is due to the fact that the practical implementation of the speech decoding process is only suboptimal. This means that several ASR modules contain some simplifications of the underlying mathematical theorems for feasibility as shown later in Chapter 2. Without simplification it is generally not possible to perform the entire decoding or training algorithms by definition owing to limitations of computational time and performance. Such simplifications are, for example, limitations in the number of hidden Markov models (HMMs) which can be used for acoustic modeling or the complexity of their Gaussian probability density functions.

Also Bayes' rule in Equation 1.1 (page 4), which shows the link between the acoustic model probability $p(X|W)$ and the language model probability $p(W)$ as a simple multiplication operation, cannot be realized strictly by definition. This decoding rule is only optimal if the underlying acoustic and language models are optimal; in other words they are based on optimal probability distribution functions. In the practical implementation of a decoding process, however, this cannot be the case. One of the limitations is, for example, that feature vectors are not independent of each other as normally stated in the definition of the acoustic models.

Another well-known example for simplification of the decoding process is the pruning applied for the search process. In order to save computational effort and memory, not all hypotheses but only a limited number of them are processed during the decoding. At each time frame, hypotheses designated as bad candidates are pruned. But the question is: how can we keep only a limited number of hypotheses without jeopardizing our main goal of finding the best one? How can we ensure at every time frame that the hypothesis is retained which will become the best one once the decoding of the utterance has finished? This dilemma is always a trade-off between *efficiency* and *accuracy*. Improving efficiency calls for pruning as many hypotheses as possible to keep the size of the search space as small as possible. Improving accuracy, on the other hand, calls for keeping the best hypothesis until the very end of the decoding process. Extensive pruning, however, increases the risk of losing the best hypothesis at a specific pruning step. Classical pruning approaches are generally based on hypothesis score rating like *beam pruning*, or they specify the amount of hypotheses to be pruned as preset values such as *histogram pruning*. These approaches work without considering the actual quality of hypotheses. In contrast to classical pruning approaches this work presents a novel dynamic pruning technique which was developed to consider the quality of the hypotheses in the search space at each time frame. This is carried out by computation of normalized log likelihood based confidence measurement in real time to achieve higher efficiency in pruning.

The thesis summarizes the results of the author's research activities carried out in collaboration with the Institute for Human Machine Communication at the Techische Universität München. Results of experiments are presented concerning CM quality and its impact on dialog control. Basic CM approaches and algorithms are explained and a survey of the abundant literature is presented on the state of the art in confidence measurement techniques resulting from the surge of interest concerning CM in automatic speech recognition. The dissertation describes several utilization techniques of confidence measurement which are intended to enhance performance in automatic speech recognition and user acceptance in speech-based applications.

1.2 Thesis Overview

A general overview of a state-of-the-art HMM-based speech decoding process is given in Chapter 2. The main areas are explained, i.e. acoustic modeling, Viterbi search, word graph and language modeling. The description of the main modules and algorithms in this section elucidate the details of CM techniques which are described later on in this work. Also potential problems, weaknesses and suboptimal algorithms within speech decoding are pointed out in order to present ideas for utilization of confidence measurement techniques.

Chapter 3 on page 27 presents a review of literature on different confidence measurement techniques and shows their basic underlying ideas. The methods are classified in groups based on the level (position) of their action within the speech decoding algorithms such as word and semantic level confidence measurements. Some utilization examples are also presented in this section along with a comparison between word level and semantic level confidence measurement used in One-stage Decoder for Interpretation of Natural Speech (ODINS). This chapter classifies confidence predictor features which are usually merged by neuronal network for building joint confidence in order to improve the quality of a single CM feature.

Chapter 4 on page 59 describes a confidence-guided dynamic (CGD) pruning approach which was developed based on the experimental results of this work. CGD pruning improves the efficiency of Viterbi search pruning by utilizing the normalized log likelihood (NLL) as confidence measure and makes the decoding process faster. This chapter describes the computation of NLL scores of active hypotheses in HMM-based speech recognition environment in real time for each time frame and shows how can it be utilized as the basis for pruning decisions. A comparison between CGD pruning and an adaptive pruning technique, based on adaptive control, is also given in this section.

Chapter 5 on page 79 provides examples of how confidence measurement can serve dialog flow control in human-machine speech communication. It also shows which additional knowledge sources are appropriate for improving quality of confidence scores in practical example applications. This chapter discusses main aspects of user interaction together with underlying dialog architectures. Potential problem sources in speech-based interaction are classified. The major goal of this chapter is to show CM utilization techniques which have the potential to improve user acceptance by better application ergonomics.

Conclusions and outlook are presented in Chapter 6.

Chapter 2

HMM-Based Speech Recognition

Speech recognition is a complex *decoding process* which translates speech into its corresponding textual representation. Because of the stochastic nature of speech, *stochastic models* are used for its decoding by modeling relevant acoustic speech features. The decoding process can be expressed mathematically to find the sequence of most likely words $\hat{W}$ with the *maximum a posteriori* (MAP) probability $p(\hat{W}|X)$, conditioned on the sequence of acoustic feature vectors X, as follows:

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}}\, p(W|X), \tag{2.1}$$

where $\mathcal{W}$ is the space of all possible word sequences. Using Bayes' decision rule from Equation 1.1 (page 4) the maximum can be found with minimal risk of error and the equation above can be formed as follows:

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}}\, \frac{p(X|W)\, p(W)}{p(X)}. \tag{2.2}$$

For further simplification, the observation probability $p(X)$ can be omitted, since it is independent of any word sequence W:

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}}\, p(X|W)\, p(W), \tag{2.3}$$

where $p(W)$ describes the probability of the word sequence W and therefore it is also called the language model probability. $p(X|W)$ is called the acoustic model probability as it describes the probability of a sequence of acoustic observations X conditioned on the word sequence W.

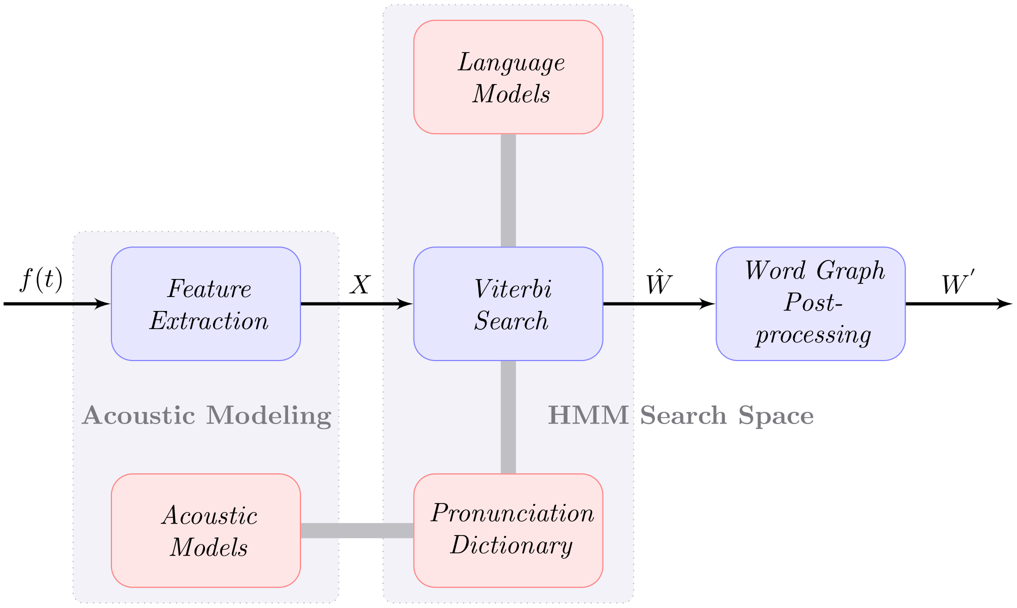

Figure 2.1: Schematic overview of speech recognition modules. The signal flow is shown by the speech signal $f(t)$ → feature vector sequence X → word sequence $\hat{W}$ → and modified word sequence $W^{'}$. The HMM search space is comprised of acoustic models, pronunciation dictionary and language models.

This chapter gives a general overview of the main modules of HMM-based ASR systems. The description of methods and algorithms provides the reader with a basic understanding of the speech decoding process. Therefore, it can be seen as a preparation for the main part of the thesis which focuses on confidence measurement techniques. Methods described later in this work refer back to ASR basics shown here.

The structure of this chapter follows the main modules of the speech recognition process as illustrated schematically in Figure 2.1. Different feature extraction methods are described in Section 2.1 on page 11 which are based on human speech production and perception processes. Construction of HMM-based acoustic models and their training methods are also shown in this section. Language models, presented in Section 2.2 on page 17, are necessary for the computation of $p(W)$, the probability of the word sequence, and also they are often used to narrow the search space in order to improve recognition performance. The Viterbi search process is optimized to find the best hypothesis among all competitive search paths; it is described in Section 2.3 on page 19. Several pruning strategies are also explained in that section which are designed to make the search process more effective. How search results can be represented and post-processed in a compact structure, the word graph, is shown in Section 2.4 on page 22. Problems and weaknesses of the whole decoding process are discussed in Section 2.5 on page 24.

2.1 Acoustic Modeling

The first step toward building an automated speech recognition system is to create a module for acoustic representation of speech. The main goal of this module is the computation of the acoustic model probability $p(X|W)$ from Equation 2.3 (page 9). Two main branches of possible model types have gained popularity, namely *neural networks* (NNs) and *hidden Markov models* (HMMs). NNs have their main advantage in modeling non-linear speech characteristics, whereas HMMs convince through their simplicity, flexibility, reusability and optimized training algorithms. HMMs are commonly used for stochastic modeling, especially in the field of automated speech recognition. This is because they have been found to be eminently suited to the task of acoustic modeling. For that reason most of the experiments for this work were carried out on the well known and widely used Hidden Markov Model Toolkit (HTK) open-source ASR system which, as its name indicates, is based on HMMs (for details see Young, 1994a). A detailed description of the theory of HMMs is proposed in Rabiner (1989); Rabiner & Juang (1993).

The hidden Markov model is a (first order) Markov model whose topology is optimized for the task of speech recognition. It is strictly a left-to-right model consisting of states and transitional edges as the example shows in Figure 2.2 on page 12. It is called *hidden* because the state sequence is effectively hidden from the resulting sequence of observation vectors. The number of states depends on the speech unit modeled by the HMM. Possible speech units are phones or phone groups (e.g. biphones or triphones), syllables, words or even sentences. The decision of which unit is the most suitable for a specific speech decoding task is always a trade-off between flexibility and trainability:

- *Flexibility*: The smaller the speech unit, the fewer different models are needed to describe a specific language domain. For example, there are about forty phones which describe the entire German language, whereas a couple thousand syllables, several ten thousand words and an almost infinite number of possible sentences are necessary in order to achieve the same level of coverage. The drawback of small speech units is often their insufficiency in modeling speech characteristics at the unit transition boundaries, so called co-articulation effects. Characteristics of speech units, vowels and consonants for example, depend on their predecessor and successor units. Positions, shapes and sizes of the articulators of the vocal tract vary while it builds a specific sound, depending on the sounds which are to be made before and after.

- *Trainability*: When a great number of speech units are needed for a specific speech domain, a huge amount of training material is necessary to train the corresponding acoustic models. Age, gender and different dialects need to

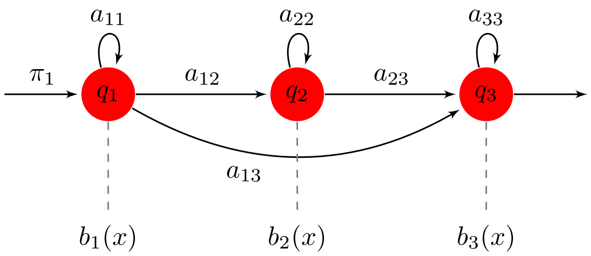

Figure 2.2: Three-state hidden Markov model, where q_i marks the state i which generates the feature vector x with the probability $b_i(x)$, a_{ij} is the probability of the transition from state i to state j and π_1 denotes the initial probability of the first state.

> be considered in the definition of the training set. The drawback of large speech units (sentences or words), which provide a high-quality description of co-articulation effects, is the limitation in the amount of training material. Therefore it is often only feasible to train them for a highly limited domain.

For reasons mentioned above, the clear tendency is to use rather small speech units like diphones and triphones with a sufficient amount of training material to allow speech recognition independently of the speaker's age, gender or dialect. On the other hand, diphones and triphones are also capable of modeling co-articulation effects in an acceptable manner.

Improved performance of speech unit representation can be achieved through use of the so-called context-dependent speech units (see Rabiner & Juang, 1993). This is because context-independent subword units are not always optimal in representing the corresponding speech units in all contexts, i.e. the effects of predecessor and successor sounds, sound stress and even the word where the speech units occur. Improved quality in acoustic speech modeling can be achieved by extending the set of subword units with context-dependent ones, such as context-dependent triphones, multiple phone units or word-dependent units, which also consider context-dependent speech features.

The link between the speech signal and the corresponding speech units is made by acoustic modeling. This means that the emission of a sequence of acoustic feature vectors X is modeled by a sequence of stationary stochastic processes which, on the other hand, are associated with a state sequence of a stochastic HMM described by probability density functions. If acoustic models are given for all classes of speech units, then it is possible to compute the class-specific probability density functions for a given feature vector sequence X. The emission of those probability density functions can be then used for later classification during the search process (see Section 2.3 on page 19).

An example of a three-state HMM as it is typically used as a diphone or triphone acoustic model is shown in Figure 2.2. In this model the state q_i produces the feature vector x with the state emission probability $b_i(x)$, and the probability of making the transition from state i to state j is marked as a_{ij}. The estimation of the acoustic model probability $p(X|W)$ in Equation 2.3 (page 9) is equivalent to the computation of the probability of the observation sequence $X = (x_1, x_2 \ldots, x_T)$ of the length T given the state sequence $Q = (q_1, q_2, \ldots, q_T)$ and the model parameter set λ as follows:

$$p(X|W) = p(X|Q;\lambda) = \prod_{t=1}^{T} p(x_t|q_t;\lambda). \tag{2.4}$$

2.1.1 Feature Extraction

HMM-based acoustic models need to be trained first and then deployed for the operation of speech recognition. Training, on the other hand, requires the definition of a set of acoustic features which are then described by the model parameters. In order to achieve best speech recognition results, detailed knowledge of human speech production and perception has been developed over past several decades. Psychoacoustic details of human acoustic perception were adapted to the field of speech recognition (see Zwicker, 1982).

The main tasks of the acoustic feature extraction procedure are the conversion of the analog speech signal to its discrete representation and the extraction of the relevant acoustic features in terms of best speech recognition capability. For this reason the speech signal is recorded and transformed to a quantized digital signal for further signal processing steps. The sampling rate for the quantization depends on the data transmission medium; e.g. typically 8 kHz in the telecommunication sector. A stationary signal for processing is required for subsequent signal processing steps. Speech can be considered stationary if it is portioned into small parts of sufficient size (i.e. frames). For this reason, as the next step following quantization, equidistant windows with a length of 20-40 milliseconds are extracted from the speech signal repeatedly at an interval of 10-20 milliseconds.

Further processing of the speech signal is based on speech processing steps which are similar to human audio preprocessing. Functional modeling of the specific loudness, as described in Ruske (1994), divides the speech relevant frequency range into 22 channels using bandpass filtering which model the excitation level of the human ear. Mel frequency cepstral coefficients (MFCCs), on the other hand, use the mel scaled filter bank. Mel filters have a triangular bandpass frequency response characteristic. Bandwidth and distance of the filters are determined by the constant mel frequency interval as described in detail in Rabiner & Juang (1993). Thus the mel spectrum consists of the output of mel filters and can be used for the computation of the cepstral coefficients.

Perceptual linear prediction (PLP) is a combination of the discrete Fourier transformation (DFT) and linear prediction (LP) techniques, as presented in Makhoul

(1975). PLP is a way of warping spectra in order to minimize the differences between speakers while preserving relevant speech information. A detailed description of PLP can be found in Hermansky & Junqua (1988); Hermansky (1990). An analysis of the ability of PLP to describe vowels independently of the gender of the speaker is given in Fabian & Vicsi (1999). The main steps of PLP analysis are as follows:

- *spectral analysis*, as the first step, means fast Fourier transformation (FFT) on *Hamming windows* with a typical window length of 20 milliseconds and subsequent conversion to power spectral density,
- *critical-band spectral resolution* warps the power spectrum onto a Bark scale using the following approximation:

$$\Omega(\omega) = 6\ ln \left(\omega/1200\pi + \sqrt{(\omega/1200\pi)^2 + 1} \right), \tag{2.5}$$

  then the Bark scaled spectra and the spectra of the critical band filter are convolved as a simulation of the ear's frequency resolution,
- *equal loudness preemphasis* is a compensation for the unequal perception of loudness at different frequencies using the equal-loudness curve:

$$E(\omega) = \frac{10^{27}\left(\omega^2 + 58.6\ 10^6\right)\omega^4}{\left(\omega^2 + 6.3\ 10^6\right)^2 \left(\omega^2 + 0.38\ 10^9\right)\left(\omega^6 + 9.58\ 10^{26}\right)}, \tag{2.6}$$

- *intensity-loudness conversion* is based on the following relation between perceived loudness and intensity: $L(\omega) = I(\omega)^{\frac{1}{3}}$,
- *autoregressive modeling* is carried out as inverse DFT as the last step of the PLP analysis.

As a further development of the PLP approach there is another speech feature representation technique known as RASTA-PLP, relative spectral transform - perceptual linear prediction as presented in Hermansky & Morgan (1994). RASTA is a speech feature extraction technique that is more robust to steady-state spectral factors in the speech channel such as distortion or noise on a telephone line.

The other main task of the feature extraction module is data reduction, which has the goal of decreasing the dimension of the speech feature vectors while keeping as much relevant information as possible for the classification step. Data reduction is carried out either by selection of the elements of the feature vector or via a transformation algorithm such as linear or non-linear discriminant analysis (LDA or NLDA). LDA maximizes the ratio of *inter-class variance* to the *intra-class variance* in a particular sequence of acoustic data vectors in order to achieve maximal separability (see Ruske, 1994). NLDA is a representation where the extraction of discriminant parameters of the data set is performed by multi-layer perceptrons (a special sort of NN) where each hidden layer computes its output as non-linear transformation of the layer's input data (see Reichl *et al.*, 1996).

2.1.2 Training of HMMs

The next step to follow definition of the relevant acoustic features is the training of the acoustic model parameters. In this context, training means the computation of model parameters based on appropriate training material in order to emulate the stochastic nature of the speech signal. Therefore, the training material needs to be representative for the speech domain for whose recognition the acoustic models will be used later. Over iterations through the training data, efficient reestimation approaches used by standard training methods converge to a local optimum. During this task, however, overfitting to the training material is to be avoided as otherwise recognition accuracy can decrease as a consequence. There are several well-established training methods such as the maximum likelihood (ML) or maximum a posteriori (MAP) approaches described in Rabiner & Juang (1993); Ruske (1994); Schukat-Talamazzini (1995).

The main goal of the ML training is to maximize the emission probability $p(X|\lambda)$ of an HMM of the class c for a given sequence of the training feature vector $X = (x_1, x_2 \dots, x_T)$:

$$\lambda_{ML} = \underset{\lambda}{\operatorname{argmax}}\, p(X|\lambda). \tag{2.7}$$

Therefore, the parameter set λ of the HMM is to be found, which fulfills this criterion. Baum-Welch training and Viterbi training are commonly used implementations of the ML training approach. One main characteristic of Viterbi training is the direct assignment of speech frames to HMM states described as $p(x_t = i)$, which is the probability of assignment from feature vector x_t of frame t to the HMM state i. The Baum-Welch training algorithm is more flexible and allows overlaps in the frame to state assignment during the training procedure.

One possible realization of the Baum-Welch training approach is based on the *forward-backward algorithm* described in Rabiner & Juang (1993). Reestimation is carried out based on the computation of forward probability, defined as

$$\alpha_t(i) = p(x_1, x_2, \dots, x_t, q_t = i|\lambda), \tag{2.8}$$

meaning the probability of the partial sequence of feature vectors $x_1, x_2, \dots, x_t$ and state i at time t, given the model parameter set λ, with the following recursion formula:

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right) b_j(x_{t+1}), \qquad 1 \leq j \leq N, \;\; 1 \leq t \leq T-1 \tag{2.9}$$

where a_{ij} is the probability of the transition from state i to j and $b_j(x_{t+1})$ is the probability that the feature vector x can be produced (emitted) in state j at time $t+1$. The initialization of recursion is as follows: $\alpha_1(i) = \pi_i b_i(x_1)$, for $1 < i < N$, where π_i is the probability of state i at time t. N is the number of possible i states, $1 \leq i \leq N$, at time t from which state j can be reached at time $t+1$. Termination is given by

$$p(X|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{2.10}$$

The backward probability $\beta_t(i)$ for $1 < i < N$ and $T > t \geq 1$ can be computed analogously as follows:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(x_{t+1})\, \beta_{t+1}(j), \qquad 1 \leq i \leq N, \;\; t = T-1, \;\; T-2, \ldots, 1 \tag{2.11}$$

with the arbitrary initial condition $\beta_T(i) = 1$ for all i, $1 < i < N$.

In contrast to the Baum-Welch training algorithm, the Viterbi training is less expensive computationally but it delivers comparably good results of model quality if a sufficient amount of training material is available. As the first step of Viterbi training, the HMM parameters are estimated based on an initial segmentation (rough guess) of the training data. Then the Viterbi decoding algorithm (see Section 2.3 on page 19) is used with these HMMs in order to find the best state sequence of the training material which is then considered as the new segmentation of the data. This new segmentation again allows the reestimation of HMM parameters and with it the reinitialization of the HMMs. Each iteration successively improves the estimation of the acoustic model probability. The training procedure is finished when no further significant improvement can be achieved.

MAP training has the goal of optimizing the maximum a posteriori probability of the model parameter set λ conditioned on the feature training vector sequence X. The optimization criterion is stated as follows:

$$\lambda_{MAP} = \underset{\lambda}{\operatorname{argmax}}\, p(\lambda|X) = \underset{\lambda}{\operatorname{argmax}}\, p(X|\lambda)\, p(\lambda), \tag{2.12}$$

where the denominator $p(X)$ is omitted since maximization is independent of it. MAP training can be also used successfully for retraining of HMM parameters in speaker adaptation tasks as described in Fabian (1999). Detailed description and a comparison of both ML and MAP training algorithms are given in Pfau (2000).

2.2 Language Modeling

The *language model* (LM), also known as *grammar*, is used to estimate the probability $p(W)$ for Equation 2.3 (page 9), which describes the probability of the estimated sequence of words. The LM can be defined as a context-free grammar (CFG), stochastic model (n-gram) or a combination of the two. Context-free grammars are used by simple speech recognition systems where the input sentences are often modeled by strict grammars. CFGs allow only utterances which are explicitly covered/defined by the grammar. Since CFGs of reasonable complexity can never foresee all the spontaneous variations of the user's input, n-gram language models are preferred for the task of large vocabulary spontaneous speech recognition.

N-gram language models represent an n-th order stochastic Markov model which describes the probability of word occurrences conditioned on the prior occurrence of n-1 other words. The probabilities are obtained from a large speech corpus and the resulting models are called unigram, bigram or n-gram language models depending on their complexity. The assumption to build such an LM is that the probability of a specific n-gram can be estimated from the frequency of its occurrence in a training set. The simplest n-gram is the unigram language model, which means a priori probabilities $p(w)$ attached to each word w. $p(w)$ describes the frequency of the specific word N_w normalized by the total number of words:

$$p(w) = \frac{N_w}{\sum_{i=1}^{M} N_i}, \tag{2.13}$$

where M is the number of different words in the training set. More generally, n-gram language models are defined as follows:

$$p(W) = p(w_1, ..., w_n) = \prod_{i=1}^{n} p(w_i|w_0, ..., w_{i-1}), \tag{2.14}$$

where the probability of the next word w_i depends on the history of words so far and w_0 is chosen to handle the initial condition. This factorization means that the complexity of the language model grows exponentially with the length of the history h_i. This complexity can be reduced with the mapping Φ which divides the space of histories into equivalence classes:

$$p(w_i|h_i) \approx p(w_i|\Phi(h_i)). \tag{2.15}$$

For trigram ($n = 3$), the above definition of word sequence probability becomes:

$$p(W) \approx \prod_{i=1}^{3} p\left(w_i | w_{i-2}, w_{i-1}\right), \tag{2.16}$$

where only the two most recent words of the history are considered.

The complexity of a language model varies widely depending on the speech domain and on the order/degree of the model. An important measurement of their complexity is the *perplexity*, a very useful measurement to compare language models. Simply speaking, the perplexity represents the average number of words which could follow a specific word. The higher the perplexity the more complex the language model. Complex language models can affect (reduce) speech recognition accuracy, since they directly influence the size of the search space. The perplexity is derived from the entropy of the information theory and is computed based on the average log probability on a per word basis with the following definition:

$$lp = -\frac{1}{k} \sum_{i=1}^{k} log_2 \left(p(w_i | h_i)\right), \tag{2.17}$$

where k denotes the total number of words and $p(w_i|h_i)$ is the language model probability. The language model perplexity is defined as 2^{lp}. Table 2.1 contains example perplexities of trigram language models for different domains.

Speech Domain	**Perplexity**
Radiology	20
Emergency Medicin	60
Journalism	105
General English	247

Table 2.1: Trigram language model perplexities for different speech domains, Roukos (1995).

Since speech applications continue to gain currency, several industrial standards regarding grammar content declarations have already been adopted (see W3C-Grammar, 2004). The examples in Appendix B on page 105 compare the standard formats ABNF and XML for a small-grammar example.

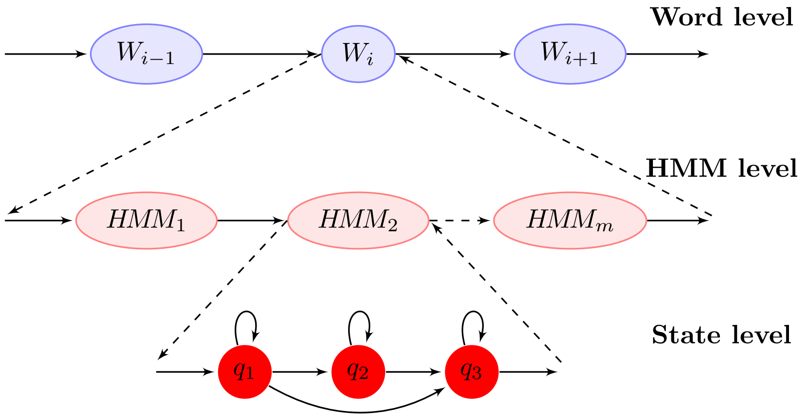

Figure 2.3: Schematic representation of different levels of the search space. In this hierarchy, words are connected based on language model rules, whereas HMMs are connected as per the pronunciation dictionary. HMMs, at the lowest level, consist of states as shown in Figure 2.2 on page 12.

2.3 Viterbi Search

The search space of the speech decoding process is given by a network of HMM states. The connection roles within this network are defined at different hierarchy levels such as the word, the HMM and the state level as shown in Figure 2.3 on page 19. Words are connected based on language model roles, whereas each word is constructed of HMMs defined by the pronunciation dictionary. The primary objective of the search process is to find the optimal state sequence in this network associated with a given speech utterance. There are several approaches to the search process such as stack and N-best decoding as described detailed in Schukat-Talamazzini (1995). This section gives an overview of the commonly used Viterbi search algorithm.

2.3.1 Viterbi Algorithm

The Viterbi algorithm is an application of the dynamic programming principle and it performs the maximum likelihood decoding (see Forney, 1973). It solves problems of unknown timescale, unknown word boundaries and unknown word sequence. The search space can be built on a static or dynamic basis. In the case of static search space, all allowed connections of HMMs are already defined at the beginning of the speech recognition task, for example context-free grammars (CFGs). The context of this search space is independent of partial results from the decoding process in progress (for example a limited list of phrases allowed in the grammar). Dynamic search, on the other hand, means that the search space is expanded dynamically depending on the partly-recognized content, e.g. unigram word loop. A schematic representation of the connected search space is shown in Figure 2.4 on page 20. In

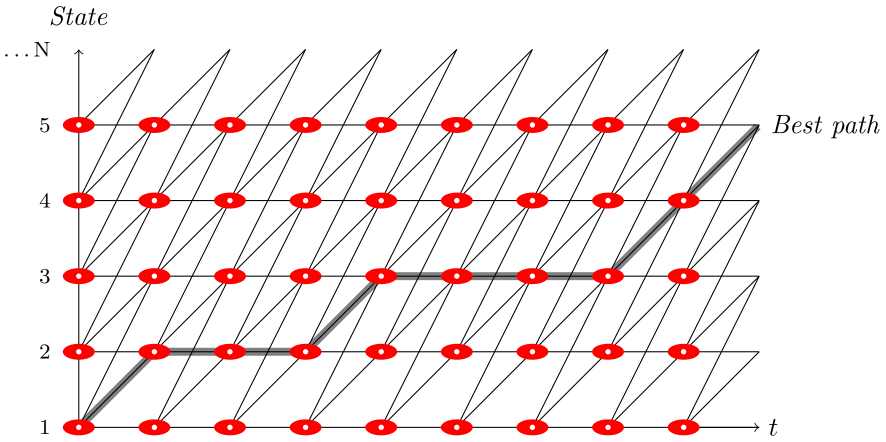

Figure 2.4: Schematic representation of the search space (trellis). Each node corresponds to one state of the state level in Figure 2.3 on page 19; also the best path is marked in the search space.

this representation, nodes mean states and edges show their possible connections to each successor state.

The Viterbi algorithm provides a solution of finding the optimal state sequence $Q = (q_1, q_2, \ldots, q_T)$ or consequently the optimal word sequence $W(Q,t)$, associated with a given sequence of feature vectors $X = (x_1, x_1, \ldots, x_T)$. In the logarithmic space this is equivalent to the computation of the best hypothesis score (highest probability) along a single path of the state network. It is a common practice to take logarithms of the model parameters and implement the Viterbi algorithm without multiplications in order to avoid problems with the number representation of computers. The steps of the Viterbi search can then be described as follows (see Rabiner & Juang, 1993):

- *preprocessing*

$$\begin{aligned} \tilde{\pi}_i &= ln(\pi_i), && 1 \leq i \leq N \\ \tilde{a}_{ij} &= ln(a_{ij}), && 1 \leq i, \quad j \leq N \\ \tilde{b}_i(x_t) &= ln\left(b_i(x_t)\right), && 1 \leq i \leq N, \;\; 1 \leq t \leq T \end{aligned}$$

- *initialization*

$$\begin{aligned} &\tilde{\delta}_1(i) = ln\left(\delta_1(i)\right) = \tilde{\pi}_i + \tilde{b}_i(x_1), && 1 \leq i \leq N \\ &\psi_1(i) = 0, && 1 \leq i \leq N \end{aligned}$$

- *recursion*

$$\tilde{\delta}_t(j) = ln\left(\delta_t(j)\right) = \max_{1\leq i\leq N}\left(\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}\right) + \tilde{b}_j(x_t)$$
$$\psi_t(j) = \operatorname*{argmax}_{1\leq i\leq N}\left(\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}\right), \qquad 2 \leq t \leq T,\ \ 1 \leq j \leq N$$

- *termination*

$$\tilde{P}^* = \max_{1\leq i\leq N} \tilde{\delta}_T(i)$$
$$q_T^* = \operatorname*{argmax}_{1\leq i\leq N} \tilde{\delta}_T(i)$$

- *backtracking*

$$q_t^* = \psi_{t+1}\, q_{t+1}^*, \qquad t = T-1,\ T-2, \ldots, 1$$

where π_i is the initialization probability of the state i at time t, a_{ij} is the probability of the state transition from state i to j and $b_j(x_t)$ is the emission probability of the feature vector x in state j at time t. N is the number of possible i states, $1 \leq i \leq N$, at time t from which the state j can be reached at time $t+1$. The quantity

$$\delta_t(i) = \max_{q_1,q_2,\ldots,q_{t-1}} P\left(q_1, q_2, \ldots, q_{t-1}, q_t = i,\ x_1, x_2, \ldots, x_t|\lambda\right) \tag{2.18}$$

describes the best probability for the first t observation vectors along a single path which ends in state i and consequently this implies for the first $t+1$ observations:

$$\delta_{t+1}(j) = \max_i \left(\delta_t(i)a_{ij}\right) b_j(x_{t+1}). \tag{2.19}$$

The array $\psi_t(j)$ keeps track of the argument that maximizes Equation 2.19 for each time frame t and state j in order to be able to retrieve the best state sequence during the final backtracking step.

Apart from the backtracking step the Viterbi algorithm is equivalent to the forward algorithm as described in Equation 2.9 (page 16), where the summation is replaced by a maximum operation:

$$\alpha_t(j) = \max_i \left(\alpha_{t-1}(i)a_{ij}\right) b_j(x_t). \tag{2.20}$$

The maximum likelihood is given by

$$\alpha_N(T) = \max_i \alpha_T(i)a_{iN} \tag{2.21}$$

and the direct computation of the likelihoods using log likelihoods for the recursion formula Equation 2.20 becomes

$$\gamma_t(j) = ln\left(\alpha_t(j)\right) = \max_i \left(\gamma_{t-1}(i) + ln(a_{ij})\right) + ln(b_j(x_t)), \tag{2.22}$$

which is the basis of the Viterbi algorithm as shown in the recursion step above.

2.3.2 Pruning

The most time-consumptive phase of the recognition process is the search process. Managing alternative hypotheses for each time frame can be very costly in terms of time and memory resources depending on the complexity of the search network. The size of the Viterbi search space for HMM-based ASR usually increases non-linearly with the vocabulary size. This is why several pruning strategies have already been proposed to reduce the time consumption of the recognition process.

Probability-based pruning controls the beam width B_{set} of the Viterbi search process at each time frame and retains only those hypotheses whose score is no less than a threshold from the score of the best hypothesis. The threshold is generally set for the entire recognition process and it must be determined over a distinct training set. However, the number of hypotheses which can be discarded depends on the distribution of the hypotheses' scores. If they are close to each other only a few of them can be pruned. Such worst case situations might force the ASR system to perform a complete search without any significant reduction in computational effort.

Rank-based pruning avoids this problem by limiting the absolute number of alternatives to a fixed value. In contrast to the beam width technique, rank pruning controls the number of hypotheses allowed for each time step independently of their distribution. For this reason all alternative hypotheses have to be ranked by their log probabilities, keeping only the best N_{max} hypotheses. The main disadvantage of this method is that two passes through all hypotheses are required and the ranking can be very time costly. To improve the efficiency of the ranking procedure, usually a histogram of the hypotheses' scores is computed - *histogram rank pruning*. As shown in Tran *et al.* (1994), in large vocabulary continuous speech recognition, ranked-based histogram pruning usually performs better than probability-based pruning. It is a common practice to combine both probability-based and rank pruning, which allows better results to be achieved due to memory saving and reduction of computational time effort while maintaining an acceptable level of recognition accuracy.

These classical pruning techniques generally use constant pruning thresholds over the entire search procedure. Both B_{set} of the probability-based pruning and N_{max} of the rank-based approach are predefined thresholds which have to be justified during cross validation tests. In the classical pruning case, these thresholds are not adjusted dynamically to fit time-variant requirements precisely. This work presents a dynamic pruning approach which solves this problem by taking variable search quality into consideration in order to perform the decoding most economically.

2.4 The Word Graph

The output of an HMM-based speech recognizer can be thought of as a single sequence of words, in a list of N-best word sequences or in a word graph of partially overlapping word hypotheses. The word graph is an efficient data structure for

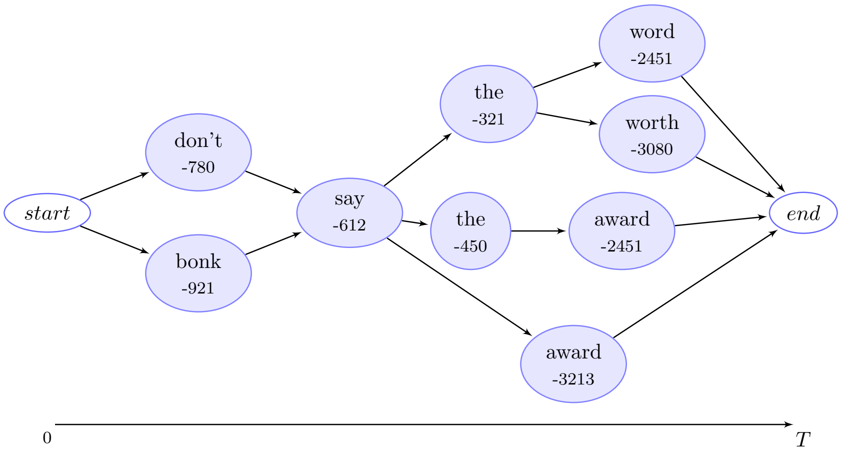

Figure 2.5: Word graph example where word hypotheses are represented by nodes and the corresponding acoustic scores by node weights.

representing large numbers of acoustic hypotheses compactly. The number of alternative word hypotheses is several orders of magnitude larger in word graphs than in typical N-best lists.

Word graphs are directed acyclic graphs, which consist of nodes connected with arcs and also contain time alignment information as well as acoustic and language model scores for each hypothesized word. Therefore, the word graph is also a proper and efficient interface between speech recognizer and linguistic processors. Speech recognition applications, especially those with large vocabulary, can benefit from such an efficient data structure. Their search procedure is generally performed in two passes. The first pass is a time-synchronous Viterbi search through a lexical tree. Context-dependent cross-word acoustic models and trigram or n-gram language models can be applied in the first pass of the search. The first pass results in a graph of the N-best word sequences, the word graph.

The terms *word lattice* and *word graph* were often used earlier to differentiate between specific stages in the recognition process. The net of words generated directly by backtracking multiple alternative N-best paths stored during the Viterbi decoding was often referred as the word lattice. This roughly constructed word lattice, however, might contain word hypotheses although they did not fit any time alignment of all possible sentences which cover the duration of a speech utterance. Many such word hypotheses of the lattice could be eliminated by converting the word lattice to a word graph during a second pass of the search procedure, as shown in Harper & Helzerman (1995).

Today, the terms word lattice and word graph are frequently interchanged in the literature (see Johnson & Harper, 1999) and both are often used as synonyms to refer to one of the following possible graph structures:

- *Mealy* format is comprised of words represented by arcs and arc weights which correspond to acoustic scores (log probabilities),
- *Moore* format consists of words represented by nodes and acoustic scores which are represented by node weights.

During conversion of a Mealy format into a Moore format, each arc in the Mealy format becomes a node of the Moore format and each node in Mealy format becomes multiple arcs.

The present work does not differentiate between these formats by the usage of different terms, like lattice or graph. Rather the term word graph is used everywhere to refer to both possible graph structures. Differences in their formats are always explained explicitly in order to maintain clarity.

Word graphs generated by acoustic recognition engines often are not compact and should be post-processed to keep down the size of the graph. They can contain spurious sentence hypotheses which can be pruned by using syntactic or semantic constraints. Linguistic processors, for instance, can rule out individual sentences which are grammatically incorrect.

The size of the word graph can be reduced very efficiently by aligning words from all possible hypotheses. As a result of this normalization technique the word graph is converted to the so-called *consensus hypothesis* (also known as *sausage*) presented by Mangu *et al.* (1999, 2000). Finally, word graphs also form a good basis for computing word posterior probabilities to provide a good measure of word confidence. Details of this post-processing step are shown in Chapter 3.

2.5 Problems, Weaknesses

After giving an overview of the HMM-based speech recognizer and the description of its main modules and their underlying algorithms we can conclude the following: HMM-based speech recognizers solve the task of speech recognition based on Bayes' optimal decoding rule. However, the direct estimation of this optimal role is generally not feasible in practice, and therefore current ASR technology applies some potentially limiting assumptions for reasons of feasibility.

One such potentially limiting assumption, for example, is that the speech signal is stationary over time frames of about 20 milliseconds, which serve as the basis of feature extraction steps (s. Section 2.1.1 on page 13). Another limitation is that Bayes' decoding rule is optimal only if the underlying acoustic and language models are based on optimal probability distribution functions. The number of models, however, is limited by the amount of training data available for model parameter estimation. Moreover, speech feature vectors are not independent from each other in contrast with the definition of the acoustic models, as normally stated.

As shown in Pfau (2000) and Fabian *et al.* (2001), variations in speech rate often lead to decreased speech recognition accuracy. HMMs are generally capable of handling time warping by abiding in a specific state with the state transition probability a_{ii} as shown in Figure 2.2 on page 12. Variations in speech rate, however, affect not only the duration of the speech units but also their spectral characteristics. If the model parameters are trained on speech samples with an average speech ratio, the general case, especially fast speech, might lead to remarkable degradations in accuracy due to mismatch situations between training and test data. Improvement in robustness against variations in speech rate can be achieved, for example, by modeling pronunciation variants for different speech rates categories as shown in Pfau (2000).

Reduction in the number of alternative search hypotheses during the Viterbi search (pruning) also affects the optimal decoding role. A partial search path can be discarded before it becomes part of the best hypothesis.

These limitations might induce a decrease in speech recognition accuracy, since they are deviations from the optimal decoding rule. Loss in accuracy, however, can be mitigated by applying different confidence measurement techniques as presented later in this work.

Chapter 3

Confidence Measurement Techniques

This chapter's focus is to survey the abundant literature on different confidence measurement techniques developed through extensive research activities in the field of speech recognition during recent decades in order to allow the assessment of the reliability of recognition hypotheses. Classification of confidence measurement techniques is presented in this chapter based on their algorithms and on their fields of application. Also shown are results of evaluation carried out in the scope of this work. Due to its outstanding performance as a confidence measure, a more comprehensive description of the computation details of the posterior probability is provided. Finally, limitations in confidence measurement quality and in operational deployments are discussed.

3.1 Classification of Methods

A quite appropriate general definition of confidence measurement is: *"A function which quantifies how well a model matches some speech data; where the values of the function must be comparable across utterances."* as given in Williams (1999). In other words, confidence measurement (CM) is an absolute measure of hypothesis quality and it can be applied to different levels of speech units as a basis for deciding the acceptance or rejection of specific hypotheses. The performance of speech recognizers is subject to deterioration primarily in the following two situations:

- Speech recognizers running under *mismatched conditions*, i.e. conditions vastly different from those for which they were prepared during training; for example, if trained using less-suitable acoustic training data for live situations such as background noise, channel distortion and reverberation.

- In the case of unexpected speech inputs, also known as *out-of-vocabulary* (OOV) words, the recognizer is forced to produce incorrect hypotheses from its vocabulary by nature.

In this section CM techniques are classified with respect to different points of view. There are few overview papers available in the corresponding literature which categorize CM techniques into various groups such as Williams (1999) and Jiang (2005). In this dissertation, confidence measures are categorized according to three different points of view, namely, *1)* depending on the speech unit to which they are applied, *2)* according to the underlying computation methods and *3)* with respect to their utilization in different applications associated with the recognition process. According to the speech unit to which CM is applied, we can differentiate among confidence measurement techniques based on subword units (e.g. phonemes), word hypotheses, semantic concepts or entire utterances. Especially the usage of semantic concepts with the aid of posterior scores has been shown to be useful for speech dialogs as described in Section 3.5 on page 46. Depending on the underlying computation methods, CM techniques can be classified roughly into two categories:

- *Confidence predictor features* based on acoustic and language model information collected during decoding. Generally such features are merged to a single probabilistic confidence decision often made by neural network (NN) classifiers as will be shown later in Section 3.2 on page 29.

- *Posterior probability based* confidence measures computed either during decoding (as proposed e.g. in Kamppari & Hansen, 2000) or in a post-processing step on N-best lists or word graphs (see Wessel *et al.*, 2001). Due its prevalence among today's CM techniques, a more detailed description of the computation of posterior scores as implemented in the scope of this work is presented in Section 3.4.1 on page 34.

Another possible classification of confidence measurement techniques can be made based on the field of applications utilizing different CM techniques. A wide range of CM approaches is used in research and practice in order to support the following areas:

- *Rescoring and pruning:* As mentioned earlier in Section 2.5 on page 24, direct implementation of Bayes' optimal decoding is generally not feasible in practice and therefore current ASR systems apply several potentially limiting assumptions for reasons of feasibility. Analysis of N-best hypotheses or word graphs using CM in a post-processing step has the potential to improve performance of the recognizer. Similarly, using CM techniques in pruning during the search process, for example, allows further optimization of time consumption. In the scope of this work a confidence-guided dynamic pruning technique was developed which is described in detail in Chapter 4 on page 59.

- *Rejection* techniques: This group of applications consists of tasks which are often referred to as *out-of-vocabulary detection*, *keyword spotting* or *utterance verification*. Common to all these tasks is the identification of hypothesis errors based on different CM techniques that incorporate various *in-vocabulary* and *out-of-vocabulary* assumptions regarding recognizer input. A short description of commonalities and differences along with a selection of corresponding literature is presented in Section 3.6.2 on page 49.

- *Adaptation* methods of acoustic models are destined to improve recognition performance under certain conditions which could not be considered adequately (or not at all) during training of the recognizer. Unsupervised adaptation is possible with the aid of appropriate CMs as shown in Section 3.6.3 on page 51.

- *Dialog management* strategies are intended to lend more intelligence to speech-based human-machine interaction. This is unthinkable without appropriate confidence scores which serve as the basis of decisions made during user interactions. A more practically-oriented discussion of this subject can be found in Chapter 5 on page 79.

This work elaborates on each of the classification items above to provide a comprehensive description of state-of-the-art CM techniques. The following section begins with an overview of predictor features used for the computation of confidence measurement.

3.2 Confidence Predictor Features

The vast majority of publications on confidence measures deals with ingenious combinations of confidence predictor features which are collected during decoding. Features are denoted as predictor features if they are appropriate to distinguish clearly between correct and incorrect hypotheses based on their probabilistic distribution. Jiang (2005) gives a relatively exhaustive list of predictor features which can be originated from acoustic or language models at word, utterance or semantic levels. A slightly enhanced list of predictor features is given as follows:

- *normalized likelihood score:* acoustic score of speech units (e.g. phoneme, word) divided by the number of frames that they span

- *N-best list related features*

    ◦ *N-best count:* the percentage of times a hypothesis appears at a similar position in the N-best list

- ◦ *N-best homogeneity:* the ratio of the score of paths containing the hypothesized word to the total score of the N-best list (see Zhang & Rudnicky, 2001)

- *word graph related features*
  - ◦ *hypothesis density:* the number of arcs on the word graph that span the time segment of a hypothesis (see Kemp & Schaaf, 1997)
  - ◦ number of similar hypotheses with similar locations in the word-graph
  - ◦ number of similar paths containing the same hypothesis in the word-graph (see Sun *et al.*, 2003)
  - ◦ ratio of all paths passing through a specific hypothesis to all possible paths

- *acoustic stability:* the measure of hypothesis occurrence at the same position in a list of hypotheses generated for different weightings between language model scores and acoustic model scores (see Kemp & Schaaf, 1997)

- *language model related features*
  - ◦ *back-off behavior* of n-gram language models along different lengths of word context (see Uhrik & Ward, 1997)
  - ◦ *language model score:* the log-probability for each word in a sequence as computed from a back-off language model in its history (see San-Segundo *et al.*, 2001)

- *parsing related features*
  - ◦ position of parsed words within a semantic slot
  - ◦ whether or not a word is parsed by grammar, as part of a slot (see Zhang & Rudnicky, 2001)

- *log likelihood ratio related features*, e.g. the ratio of the log likelihood of the best hypothesis to other hypotheses (see Bouwman & Boves, 2001)

- *duration-related features* based on HMM state, vowel, phoneme or word duration

Unfortunately, most of the predictor features are not optimal in separating correct hypotheses from incorrect ones. Therefore it is common practice to merge a certain combination of these predictor features into a single probabilistic confidence score in order to be able to make a unique decision regarding the correctness of recognition hypotheses. The fusion of predictor features is generally done by NNs which have to be trained for optimal decision of a specific task. However, CM performance can be improved only when combined features are statistically independent. Overlap between features is often quite large as reported in Kemp & Schaaf (1997) and in this case the resulting CM quality is largely determined by the performance of the best feature.

Wessel *et al.* (1999) present a comparison of several CM techniques based on N-best lists and word graphs and conclude that posterior word probabilities clearly outperform alternative measures such as acoustic stability or hypothesis density. Furthermore it is shown that posterior scores estimated on word graph perform better than those estimated on N-best lists. Falavogna *et al.* (2002) confirm these results after comparing posterior scores computed on word graph with several acoustic based predictor features on three different speech corpora.

3.3 Fusion of Predictor Features

Within the scope of this work several evaluation studies have been made regarding the impact of speech rate, especially fast speech, on recognition performance (see Fabian *et al.*, 2001). As shown in this section, speech rate related measures have been found to be quite useful for CM. As reported in Fabian *et al.* (2001) the performance of speech recognition systems can deteriorate considerably with variations in speech rate. This is often caused by mismatched conditions when input data during recognition are not sufficiently represented within the training corpus of the acoustic models. Fabian *et al.* (2001) report significant differences in word error rates depending on the speech rate categories slow, average and fast speech as is shown clearly in Table 3.1 obtained by the evaluation corpus Verbmobil '96. As we can see in the table, there is a dramatic increase in word error rate (WER) for fast speech in comparison with other speech rates; WER for fast speech is almost double that for slow speech. The classification of speech material into groups of slow, average or fast speech is based on vowel rate distribution as described in detail in Pfau (2000).

Speech Rate	slow	average	fast
WER [%]	24.2	33.8	44.4

Table 3.1: Dependency of word error rate (WER) on speech rate; i.e. for slow, average and fast speech.

The differences in recognition performance depending on speech rate are caused

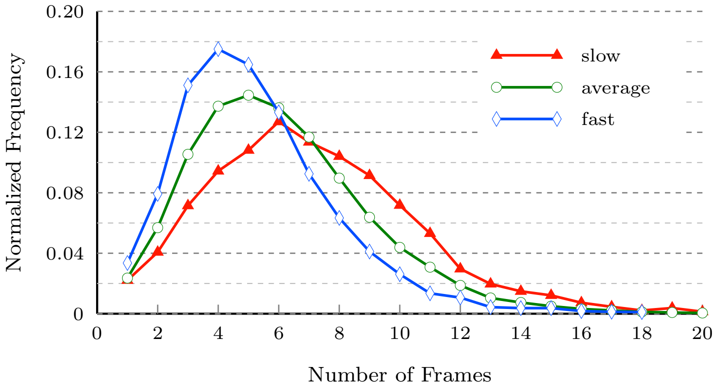

Figure 3.1: Histogram of vowel length for the different speech rates slow, average and fast speech, for the vowel /a/ in the training set Verbmobil '96.

by large deviations in temporal and spectral characteristics of speech samples with different speech rates, i.e. vowel length variations or displacement in position of formant frequencies (see Kuwabara, 1997; Martinez *et al.*, 1997). To illustrate such speech rate dependency, Figure 3.1 shows the distribution of phoneme lengths for slow, average and fast speech rates for the German phoneme /a/ determined on the Verbmobil '96 corpus; and as we can see, there are significant differences in mean values of those distributions.

Based on these findings Weber (2002) reports evaluation results achieved by the fusion of several duration-related predictor features with features derived from the acoustic score by means of a multi-layer perceptron. Duration based features are computed depending on speech rates and relative vowel or phoneme lengths compared to statistics generated on the training corpus. Weber (2002) concludes that the performance of duration based features falls below that of acoustic score related features. However, the fusion of both groups achieved better performance than the acoustic features alone. Among the duration based features, those which performed the best deal with frequency and amount of statistical deviation of phoneme lengths in a word hypothesis.

Similarly to the above, Goronzy *et al.* (2000) propose different measures based on phoneme durations in order to take mismatch situations between training and testing data into account. Statistics of phone durations obtained on training data serve as the basis for comparison with durations found by the recognizer during recognition. Comparison results of different duration-based features are used by a neural network as input for confidence score computation. Goronzy *et al.* (2000) conclude that even though duration-based features were not found to be as good in reducing confidence error rate (CER) as features related to acoustic scores, they have a great advantage over score-based features: duration-related features are independent of a specific recognizer and therefore retraining of the NN classifier is not necessary if the recognizer is changed, whereas the usage of score-based features would require retraining in that cases.

3.4 Posterior Probability as Confidence Measure

Posterior probability of a word hypothesis can be used directly as a measure of confidence as proposed in several papers such in Weintraub *et al.* (1997). However, its efficient computation can be a challenge, especially in dependency on the underlying recognizer architecture (e.g. NN or HMM). For practical reasons, ASR systems normally omit the computation of the observation probability $p(X)$, the normalization term in the formula of the posterior probability Equation 2.2 (page 9), because it is constant across different hypotheses and so negligible for maximum decision. Hypothesis scores computed in that way, however, are inadequate as confidence measure. In contrast, the posterior probability including the normalization term $p(X)$ is well suited to measure hypothesis reliability but precise estimation of the observation probability $p(X)$ can be quite difficult. The mathematical formulation of the observation probability is as follows:

$$p(X) = \sum_{\forall\, H \in \mathcal{H}} p(H)\, p(X|H), \tag{3.1}$$

where H denotes a specific hypothesis for X, $p(H)$ is the hypothesis probability and $p(X|H)$ is the probability of observing X by assuming that H is the underlying hypothesis. The summation must be made over all H in the entire set of possible hypotheses $\mathcal{H}$. In order to make the computation of Equation 3.1 feasible for practice, approximating methods are normally applied. Kamppari & Hansen (2000), for example, propose the utilization of a *catch-all model* for the computation of $p(X)$ within an HMM environment. This method provides reasonable performance and it is also used in the scope of this work for the confidence-guided dynamic pruning approach as described detailed in Chapter 4 on page 59.

As shown in Wessel *et al.* (2001), posterior probability can also be computed efficiently based on word graphs in a post-processing step and thus independently from the decoding implementation method. Word graphs are widely used in speech recognition systems in order to represent resulting hypotheses in a very compact way. They can also be utilized to compute the posterior probability for each hypothesized word in the word graph. Mangu *et al.* (1999) and Wessel *et al.* (2001) have shown that the posterior probability computed in this way can be used directly as a measure of hypotheses' confidence. Its quality can be further improved by taking time alignment information of similar word hypotheses into consideration. It is also consistently proven that the quality of CM as posterior probability outperforms alternative methods such as acoustic stability and hypothesis density.

For this reason, one main focus of this work is to analyze different CM techniques based on the computation of posterior probabilities of the hypothesized words on word graphs. Two different computation methods are compared: on the one hand, *simple accumulation* of the posterior probabilities is carried out for similar word hypotheses which overlap in time, as proposed by Wessel *et al.* (2001). On the other hand, a more complex method is applied as part of the computation of the so-called *consensus hypothesis*, as proposed by Mangu *et al.* (1999). In order to determine the performance of these algorithms, investigations were carried out on two evaluation corpora for different word graph densities.

This section recapitulates the computation of the word posterior probability and its extension for both techniques, *simple accumulation* and *consensus hypothesis*. Evaluation results are also presented subsequently regarding the impact of word graph density on the quality of these CM techniques.

3.4.1 Computation of Posterior Probability on Word Graph

As shown in Section 2.4 on page 22, a word graph may consist of nodes connected with arcs and contains time alignment information as well as acoustic and language model scores for each hypothesized word. Word graphs used in this section have the Mealy format, which represents words by arcs and by arc weights corresponding to acoustic and language model scores (log probabilities). The time alignment information for each word is given by starting time τ and ending time t. The word graph also contains two special nodes: the START node, which corresponds to the beginning of the utterance at $\tau = 1$, and the END node, which stands for the end of the utterance at $t = T$ as shown in Figure 3.2 on page 39. The posterior probability of a specific word, $p([w;\tau,t])$, can be computed by summing up the posterior probabilities of all sentence hypotheses containing this specific word at the given position, as described in Wessel *et al.* (2001), which means the total probability of all complete paths; any path from START node to END node in the word graph that passes through the arc with word $[w;\tau,t]$ conditioned on the sequence of feature vector $x_1^T = x_1, x_2, \ldots, x_T$ can be expressed as follows:

$$p\left([w;\tau,t]|x_1^T\right) = \sum_{\forall\, w_1^M \in \mathcal{G}} \frac{\prod_{m=1}^{M} p\left(x_{\tau_m}^{t_m}|w_m\right) p\left(w_m|w_1^{m-1}\right)}{p\left(x_1^T\right)}, \tag{3.2}$$

where $\mathcal{G}$ is the set of all paths in the graph from START node to END node passing through the word $[w;\tau,t]$. Such a path can be thought of as a sequence of M word hypotheses with given time boundaries τ_m and t_m, which can be also expressed as

$$w_1^M = [w_1;\tau_1,t_1], [w_2;\tau_2,t_2], \ldots, [w_M;\tau_M,t_M],$$

with starting time $\tau_1 = 1$, ending time $t_M = T$ and where $t_{n-1} = \tau_n$ for all $n = 2, \ldots, M$. In the equation above, $p(x_{\tau_m}^{t_m}|w_m)$ denotes the acoustic model probability for the observation sequence within the time boundaries τ_m and t_m conditioned on word w_m, $p(w_m|w_1^{m-1})$ is the language model probability computed for the history

w_1^{m-1} of word w_m along the path and in the denominator $p(x_1^T)$ is the observation probability.

Note that the posterior probabilities of all word graph edges at a specific point in time $t^{'}$, which also can be seen as a cut through the word graph, must always add up to 1:

$$\sum_{\substack{[w;\tau,t]: \\ \tau \leq t^{'} \leq t}} p\left([w;\tau,t]|x_1^T\right) = 1 \qquad \forall\ t^{'} \in [1,\ldots,T]. \tag{3.3}$$

As shown in Wessel *et al.* (2001), the computation of the word posterior probability can be carried out very efficiently using the *forward-backward algorithm* on the word graph. It allows separate computation of the *forward probability* and *backward probability* of a word hypothesis $[w;\tau,t]$ which can be than combined to determine the word posterior probability.

The forward probability of a specific word hypothesis, $p_{_{fw}}([w;\tau,t])$, is the total probability, i.e. the sum of the posterior probabilities of all partial paths in the word graph which start from the START node and end in word hypothesis $[w;\tau,t]$:

$$p_{_{fw}}\left([w;\tau,t]\right) = \sum_{\forall\ w_1^n \in \mathcal{F}} \prod_{i=1}^{n} p\left(x_{\tau_i}^{t_i}|w_i\right) p\left(w_i|w_1^{i-1}\right), \tag{3.4}$$

where $\mathcal{F}$ is the set of all paths starting at the node with time stamp $\tau_1 = 1$ and ending at the node with $t_n = t$; note: $\mathcal{F}$ can be also expressed as a set of word sequences

$$w_1^n = [w_1;\tau_1,t_1],[w_2;\tau_2,t_2],...,[w_n;\tau_n,t_n],$$

where $[w_n;\tau_n,t_n] = [w;\tau,t]$.

Analogously the backward probability of a specific hypothesis $p_{_{bw}}([w;\tau,t])$, is the total probability of all partial paths which start with the word hypothesis $[w;\tau,t]$ and end at the END node of the word graph. $p_{_{bw}}$ can thus be computed as shown in the following formula:

$$p_{_{bw}}\left([w;\tau,t]\right) = \sum_{\forall\ w_n^M \in \mathcal{B}} \prod_{j=n}^{M} p\left(x_{\tau_j}^{t_j}|w_j\right) p\left(w_j|w_1^{j-1}\right), \tag{3.5}$$

where $\mathcal{B}$ is the set of all paths starting at the node $\tau_n = \tau$ and ending at $t_M = T$; note that $\mathcal{B}$ can also be expressed as the set of word sequences

$$w_n^M = [w_n;\tau_n,t_n],[w_{n+1};\tau_{n+1},t_{n+1}],\ldots,[w_M;\tau_M,t_M],$$

where $[w_n;\tau_n,t_n] = [w;\tau,t]$.

The posterior probability of the word hypothesis $[w;\tau,t]$ can be computed using forward and backward probabilities as shown in the following formula:

$$p\left([w;\tau,t]|x_1^T\right) = \frac{p_{fw}\left([w;\tau,t]\right) p_{bw}\left([w;\tau,t]\right)}{p\left(x_1^T\right) p\left(x_\tau^t|w\right)}, \tag{3.6}$$

where the acoustic probability, $p(x_\tau^t|w)$, in the denominator is necessary because it was included twice by the computation of p_{fw} in Equation 3.4 and also by p_{bw} in Equation 3.5.

In the denominator of Equation 3.6, the probability of the acoustic observation $p\left(x_1^T\right)$ can also be thought of as the total probability of all possible paths in the word graph and can be calculated either as the forward sum of probabilities of all word sequences w_1^M, starting in the START node of the word graph and ending in the END node, or as the backward sum of probabilities of all paths starting with the END node backward to the START node. Correct calculations lead to identical results in both cases.

Implementation Details

For the evaluations whose results are presented later in this work the word posterior probabilities were computed using the *forward-backward algorithm* of the HTK system. This routine works on word graphs produced by HTK but it needed some modifications by the author in order to support the computation of word posterior probabilities and to use them as a confidence measure. This section describes several implementation details in order to make the theoretical background above clearer.

As far as the computation of forward probability is concerned, the word graph is processed from the START node to the END node. In the implementation of p_{fw} acoustic and language model scores are summed up for each arc (word hypothesis) along all possible paths. In case paths merge into a specific node of the word graph, scores are summed up in the logarithmic space. It is important to note that this procedure must be performed in topological order to ensure that when processing of a specific arc occurs, all its predecessor arcs have already been processed. For the computation of p_{fw} the information given by each arc in HTK word graph, i.e. word w, time alignment $[\tau, t]$ acoustic score $a_{[w;\tau,t]}$ and language model score $l_{[w;\tau,t]}$, was extended with a variable for the logarithm of the forward probability $f_{[w;\tau,t]} = \ln\left(p_{fw}([w;\tau,t])\right)$. The computation steps in the forward direction are as follows:

- *initialization* of forward score of each word graph arc $\mathfrak{r}$

$$f_{\mathfrak{r}} = f_{[w_{\mathfrak{r}};\tau_{\mathfrak{r}},t_{\mathfrak{r}}]} = -\infty \quad \text{i.e. log probability 0,} \;\; \forall \;\; \mathfrak{r} \in \mathfrak{R} \tag{3.7}$$

- *propagation* of forward scores through the word graph

$$\forall\ \mathfrak{n} \in \mathfrak{N}_{top} \text{ set of all nodes in topological order}$$

$$\forall\ \mathfrak{r}_s \in \mathfrak{R}_s(\mathfrak{n}) \text{ set of all successor arcs of node } \mathfrak{n}$$

$$f_{\mathfrak{r}_s} = \bigoplus_{\substack{\forall\ \mathfrak{r}_p:\\ \mathfrak{r}_p \in \mathfrak{R}_p(\mathfrak{n})}}\!{}_{\ln} \left(f_{\mathfrak{r}_p} + \alpha a_{\mathfrak{r}_s} + \beta l_{\mathfrak{r}_s}\right) \tag{3.8}$$

In Equation 3.8 $f_{\mathfrak{r}_s}$ denotes the forward score of the successor arc $\mathfrak{r}_s$, $f_{\mathfrak{r}_p}$ is the forward score of the predecessor arc $\mathfrak{r}_p$ and $\mathfrak{R}_p(\mathfrak{n})$ is the set of all predecessors of node $\mathfrak{n}$. $a_{\mathfrak{r}_s}$ and $l_{\mathfrak{r}_s}$ are acoustic and language model scores attached to the successor arc $\mathfrak{r}_s$ scaled by α and β. The *initialization step* with Equation 3.7 (page 36) sets all forward scores $f_{\mathfrak{r}}$ to $-\infty$ in the logarithmic space (in practice a large negative number, e.g. $-1.0E+10$) which is equivalent to zero posterior probability, $p_{f_w} = 0$. $f_{\mathfrak{r}}$ is attached to each arc $\mathfrak{r}$ of the set of all arcs of the word graph $\mathfrak{R}$. The *propagation step* computes the forward score of each arc by utilization of multiple nested loops. The outer loop, $\forall\ \mathfrak{n} \in \mathfrak{N}_{top}$, works on the entire set of word graph nodes $\mathfrak{N}_{\mathfrak{top}}$ sorted in topological order. The inner loop works on the set of successor arcs $\mathfrak{R}_s(\mathfrak{n})$ of node $\mathfrak{n}$. For each node $\mathfrak{n}$ the forward score of each successor arc f_{r_s} is computed by contributing all predecessor arcs $\mathfrak{R}_p$ of the node $\mathfrak{n}$ using Equation 3.8. This computation step is carried out by performing the inner loop for all successor arcs $\forall\ \mathfrak{r}_s \in \mathfrak{R}_s(\mathfrak{n})$.

Let us take a closer look at Equation 3.8: the term $f_{\mathfrak{r}_p} + \alpha a_{\mathfrak{r}_s} + \beta l_{\mathfrak{r}_s}$ describes the product development, the Π term in Equation 3.4 (page 35), along possible paths through a specific word hypothesis (arc) where $a_{\mathfrak{r}_s}$ and $l_{\mathfrak{r}_s}$ are acoustic and language model scores of the specific arc $\mathfrak{r}_s$. The logarithmic sum, $\bigoplus_{\ln}$ in Equation 3.8 is equivalent to the Σ term in Equation 3.4, to summing up the probabilities of different paths. The operator $\bigoplus_{\ln}$ is defined for the addition of scores in the logarithmic space as

$$\ln(x_1 + x_2 + \ldots + x_n) = \bigoplus_{i=1}^{n}{}_{\ln}(x_i).$$

It must be emphasized that the logarithmic sum in Equation 3.8 only takes effect in case of merging edges in the word graph due to the initialization of the forward scores. The above algorithm computing the forward score has the main advantage that once all word graph nodes have been processed the forward scores of all word hypotheses of the entire word graph are available for further computations.

Beside the language model scaling factor β in Equation 3.8, the scaling of the acoustic scores, α, is also proposed in Wessel *et al.* (2001). In computing posterior scores, the choice of proper scaling factors is obviously crucial. This is because additive combination of hypotheses' scores in the logarithmic space is greatly affected by excessively high scores. Using appropriate scaling, however, prevents logarithmic sums from being dominated by only a few word graph hypotheses in the large dynamic range of acoustic scores. Since α holds a major impact on the computation of the forward score, it has to be estimated on a cross-validation corpus, a set distinct from the evaluation corpus, which is described in Section 3.7.2 on page 53. The results of empirical studies presented in Wessel *et al.* (2001) propose settings for the scaling factor of α by 0.05 or 0.06, and for β a value very close to 1. Tests performed in the scope of this work using the implementation with the HTK system confirm that these scaling factors consistently lead to good results.

Analogously to the implementation of the forward score, in order to support the computation of the backward score the set of information given for each hypothesis arc in the word graph was extended with a variable for the backward score $b_{[w;\tau,t]} = \ln\left(p_{bw}([w;\tau,t])\right)$. The backward score is determined by processing the word graph in the backward direction, from the END node to the START node as follows:

- *initialization* of backward score of each word graph arc $\mathfrak{r}$

$$b_{\mathfrak{r}} = b_{[w_{\mathfrak{r}};\tau_{\mathfrak{r}},t_{\mathfrak{r}}]} = -\infty \quad \text{i.e. log probability } 0,\ \ \forall\ \ \mathfrak{r} \in \mathfrak{R} \tag{3.9}$$

- *propagation* of backward scores through the word graph

$$\forall\ \ \mathfrak{n} \in \mathfrak{N}_{rev.\,top}\ \ \text{set of all nodes in reverse topological order}$$

$$\forall\ \ \mathfrak{r}_p \in \mathfrak{R}_p(\mathfrak{n})\ \text{set of all predecessor arcs of node}\ \mathfrak{n}$$

$$b_{\mathfrak{r}_p} = \bigoplus_{\substack{\forall\ \mathfrak{r}_s:\\ \mathfrak{r}_s \in \mathfrak{R}_s(\mathfrak{n})}}{}_{\ln} \left(b_{\mathfrak{r}_s} + \alpha a_{\mathfrak{r}_p} + \beta l_{\mathfrak{r}_p}\right) \tag{3.10}$$

In Equation 3.10 $b_{\mathfrak{r}_p}$ denotes the backward score of the predecessor arc $\mathfrak{r}_p$, $b_{\mathfrak{r}_s}$ is the backward score of the successor arc $\mathfrak{r}_s$ and $\mathfrak{R}_s(\mathfrak{n})$ is the set of all successors of node $\mathfrak{n}$. $a_{\mathfrak{r}_p}$ and $l_{\mathfrak{r}_p}$ are acoustic and language model scores attached to the predecessor arc $\mathfrak{r}_p$ scaled by α and β. Similar to the computation of the forward score above, backward scores are set to $-\infty$ in the *initialization step*. During the *propagation step*, backward scores are computed by utilization of nested loops. In this case, however, the outer loop works in reverse topological order on the entire set of the word graph nodes $\mathfrak{N}_{rev.\,top}$. For each node $\mathfrak{n}$, the backward score of each predecessor arc b_{r_p} is computed using Equation 3.10 by contributing all successor arcs of node $\mathfrak{n}$, $\mathfrak{R}_s$. This computation step is carried out by performing the inner loop for all predecessor arcs $\forall\ \ \mathfrak{r}_p \in \mathfrak{R}_p(\mathfrak{n})$. After processing is completed, the backward score of each word hypothesis is available for further computation of the posterior score.

Figure 3.2 on page 39 shows a simple word graph example where words are represented by arcs and time boundaries are represented by starting and ending

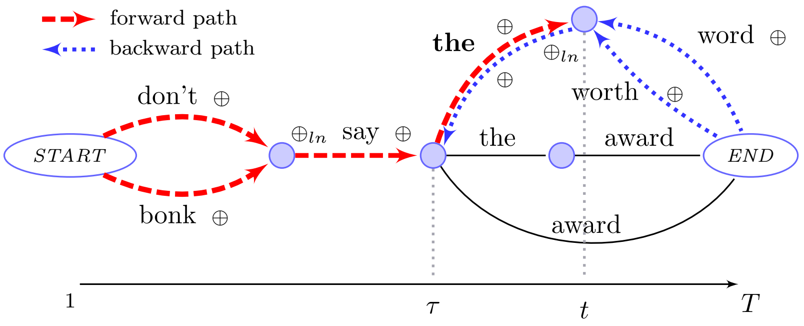

Figure 3.2: Schematic view of the computation of forward and backward scores on an example word graph for a specific word (*the*) with starting time τ and ending time t. Summing up scores along the example paths is marked by operator $\oplus$ whereas path merging using the logarithmic sum is marked by operator $\oplus_{ln}$.

nodes for each arc $[\tau, t]$, where the START node of the graph corresponds to the beginning of the utterance ($\tau = 1$) and the END node to the end of the utterance ($t = T$). Figure 3.2 can be used to explain the computation of forward and backward scores more clearly. The figure shows a schematic view of the propagation of forward and backward scores along word graph arcs for an example word hypothesis *"the"* (boldface in the word graph). All partial paths are denoted which are used for computation in forward and backward directions. Dashed lines mark those partial paths which are relevant for the computation of the forward score. As we can see in the forward direction, scores are summed up along two different possible paths starting at the START node and ending in the arc of the specific hypothesis (word *"the"*). To make computation steps more clear the operator $\oplus$ marks summation of acoustic and language model scores along paths as the term $f_{\mathfrak{r}_p} + \alpha a_{\mathfrak{r}_s} + \beta l_{\mathfrak{r}_s}$ in Equation 3.8 (page 37). Additionally, operators $\oplus_{ln}$ are placed where the logarithmic sum takes effect, namely at merging paths. Paths involved by the computation of backward scores are plotted with dotted lines.

The complete trees of forward and backward graphs of the example word graph from Figure 3.2 are shown in Figure 3.3 on page 40 in order to visualize the implementation of the algorithms as defined in Equation 3.8 (page 37) and Equation 3.10 (page 38). As we can see in Figure 3.3, the computation of the forward scores begins at the START node of the graph and sums up all scores of all alternative forward paths until the END node; backward scores are computed similarly in the backward direction.

The score for the observation probability $p\left(x_1^T\right)$ in Equation 3.2 (page 34) corresponds to the total sum of all probabilities of all possible paths through the word graph. In the logarithmic space it can be computed either by the logarithmic sum, $\bigoplus_{\ln}$, of all forward scores of all arcs which end at the END node or by summing up

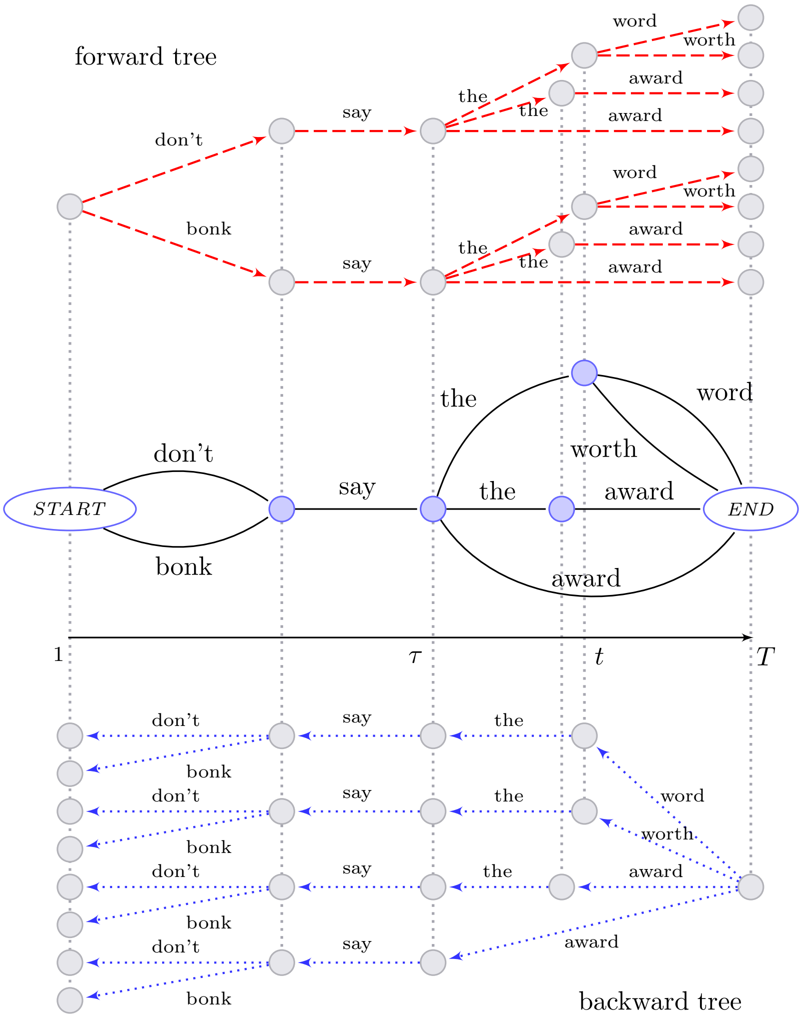

Figure 3.3: Schematic view of complete forward and backward trees passed through the entire word graph during computation of posterior scores using the forward-backward algorithm.

all backward scores in the logarithmic space attached to arcs starting at the START node:

$$o \;=\; \underset{\substack{\forall\;\mathfrak{r}_p:\\ \mathfrak{r}_p \in \mathfrak{R}_p(\mathfrak{n}_{\text{END}})}}{\bigoplus\nolimits_{\ln}} \left(f_{\mathfrak{r}_p}\right) \;=\; \underset{\substack{\forall\;\mathfrak{r}_s:\\ \mathfrak{r}_s \in \mathfrak{R}_s(\mathfrak{n}_{\text{START}})}}{\bigoplus\nolimits_{\ln}} \left(b_{\mathfrak{r}_s}\right), \tag{3.11}$$

where o is the score of the acoustic observations $o = \ln p(x_1^T)$, $\mathfrak{R}_p(\mathfrak{n}_{\text{END}})$ is the set of arcs which end at the END node, i.e. predecessors of END. Analogously, $\mathfrak{R}_s(\mathfrak{n}_{\text{START}})$ is the set of arcs which start at the START node, i.e. successors of START. Both terms in Equation 3.11 should lead to identical observation scores if calculations proceeded correctly in previous steps. This is why comparison of those results is often used to check correctness in practice.

After forward, backward and observation scores are determined, they can be combined to form the posterior score, the log posterior probability, which is then used directly as the confidence score $C([w;\tau,t])$ of a specific word hypothesis w belonging to arc $\mathfrak{r}_w$ of the word graph:

$$C([w;\tau,t]) = f_{\mathfrak{r}_w} + b_{\mathfrak{r}_w} - (\alpha a_{\mathfrak{r}_w} + \beta l_{\mathfrak{r}_w}) - o, \tag{3.12}$$

where the subtraction of the term $(\alpha a_{\mathfrak{r}_w} + \beta l_{\mathfrak{r}_w})$ is necessary for algorithmic reasons because acoustic and language model scores of a word hypothesis were included twice in the forward score in Equation 3.8 (page 37) and also during computation of the backward score in Equation 3.10 (page 38).

3.4.2 Considering Time Alignment with Simple Accumulation

The experimental results presented in Wessel *et al.* (2001) show that the performance of the confidence measure calculated in the previous section can be significantly improved by summing up the posterior probabilities of all hypotheses of the same word with *overlapping time intervals*. This is because word graphs usually contain several hypotheses, other than the best hypothesis, which have slightly different time alignment of the same word. The usage of fixed starting and ending time as in the previous section, however, does not allow consideration of those hypotheses in the computation of the posterior probability, and the word probability is split among them. Figure 3.4 on page 42 shows schematically seven word graph arcs of the same word, which belong to different sentence hypotheses but have overlapping time intervals. This example is an excerpt of a word graph as typically produced by HTK speech recognition system. For the computation of the best word hypothesis' confidence score (arc $\mathfrak{r}_2$, boldface in Figure 3.4) by considering intersection in time boundaries of similar word hypotheses, we must sum up posterior probabilities of arcs which have overlapping time intervals (e.g. arc $\mathfrak{r}_3$ in Figure 3.4). This is equivalent to the logarithmic sum of the confidence scores in the log space:

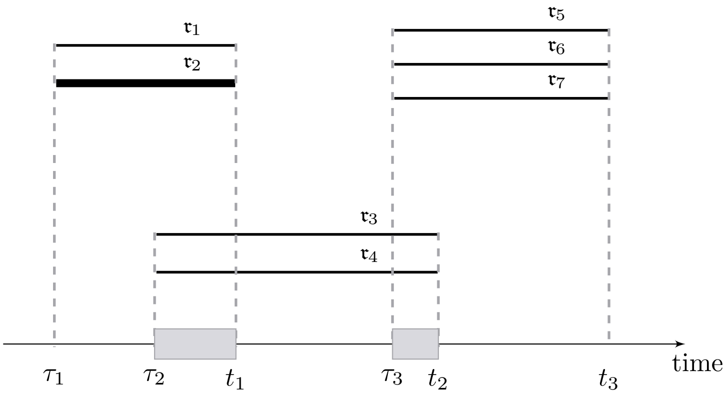

Figure 3.4: Overlapping time intervals, $\tau_2 - t_1$ and $\tau_3 - t_2$, for different arcs $\mathfrak{r}_i$ of a word graph as typically produced by HTK ASR system. Boldface denotes arcs belonging to the best recognition result.

$$C_{sec}\left([w;\tau,t]\right) = \bigoplus_{\substack{\ln \\ \forall\, [w;\tau',t']: \\ \{\tau,\ldots,t\}\cap\{\tau',\ldots,t'\}\neq\emptyset}} C\left([w;\tau',t']\right), \tag{3.13}$$

or for the example shown in Figure 3.4:

$$C_{sec}\left([w_2;\tau_2,t_2]\right) = \bigoplus_{i=1}^{4}{}_{\ln} C_{\mathfrak{r}_i}, \tag{3.14}$$

where $C_{\mathfrak{r}_i}$ is the confidence score of arc $\mathfrak{r}_i$ according to the definition in Equation 3.12 (page 41).

It is to be mentioned that since C_{sec} does not necessarily fulfill the condition of posterior probability as formulated in Equation 3.3 (page 35) and does not sum up to unity in the normal space of probabilities, it can lead to posterior scores $C_{sec} > 0$ in the logarithmic space. It does, however, perform significantly better on five different evaluation corpora than the score defined in Equation 3.12 (page 41) as reported by Wessel *et al.* (2001). Also evaluations carried out in the scope of this work on two additional test corpora confirm better results in confidence error rate for the definition of CM as in Equation 3.13 than those based on Equation 3.12 (page 41).

Wessel *et al.* (2001) propose two additional ways of summing up posterior probabilities of word hypotheses with slightly different starting and ending time boundaries. On the one hand, the method known as C_{med} accumulates posterior probabilities restricted to only those arcs with the same word hypotheses which intersect the median time frame of the best hypothesis, for which the CM is actually computed. This method also fulfills the original condition of posterior probabilities as given in

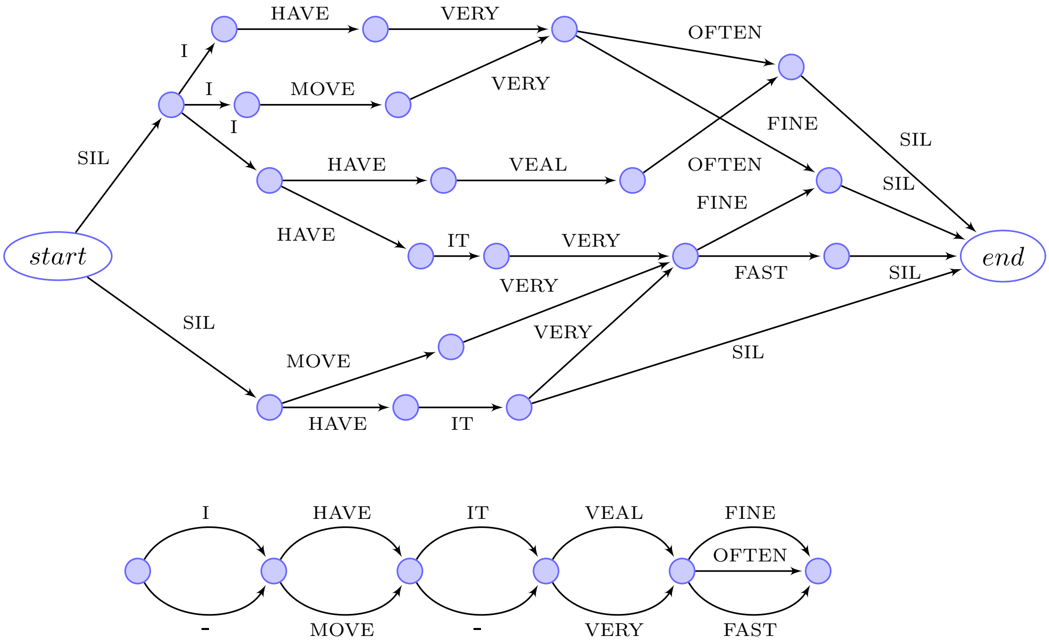

Figure 3.5: Sample word graph and corresponding multiple alignment represented as *confusion network*, presented in Mangu & Brill (1999).

Equation 3.3 (page 35). On the other hand, the method known as C_{max} accumulates posterior probabilities not only for the median time frame but for all time frames which intersect with the best hypothesis, and the maximum of these values is chosen from all sums as the measure of confidence.

3.4.3 Considering Time Alignment as Consensus Hypothesis

Another possibility for the computation of the posterior probability based confidence score on word graphs is described in Mangu *et al.* (2000). The algorithm primarily used for the computation of the so-called *consensus hypothesis* can also be applied to generate posterior probability based confidence scores. Figure 3.5 shows an example word graph with its corresponding multiple alignment. The approach presented in Mangu *et al.* (2000) selects that word at each position in the alignment which has the highest posterior score; the resulting hypothesis is called the consensus hypothesis by Mangu *et al.* (2000). For this method, posterior scores of hypothesized words are computed in the same way as described in Section 3.4.1 on page 34. However, the accumulation of the confidence scores, which makes use of the time alignment information of the word graph, differs from that described in Section 3.4.2 on page 41. The algorithm proposed in Mangu *et al.* (2000) has as its primary goal to compute the consensus hypothesis which minimizes the word error rate of recognition results rather than the sentence error rate. Empirical results in Mangu *et al.* (2000) prove a significant lack in correlation between sentence error rate and word error rate and the difference between optimizing for both. In order to determine the consensus hypothesis, the word graph is converted to a compact format through the following computation steps:

Step 1: Computation of the posterior score of each word hypothesis (arc) of the word graph, as described in Section 3.4.1 on page 34

Step 2: Building equivalence classes, composed of all the arcs with the same word label and identical starting and ending times (see Figure 3.6 on page 45)

Step 3: Merging equivalence classes which contain the same word by computation of time similarity by overlapping time intervals (*intra-word clustering*)

Step 4: Grouping equivalence classes if they correspond to different words with so-called phonetic similarity (*inter-word clustering*)

For analysis carried out in the scope of this work, it is not necessary to perform Step 4, because Step 3 already considers time alignment information of similar word hypotheses and allows accumulation of posterior scores of the word hypotheses with overlapping time intervals which are grouped together in common equivalence classes. Merging equivalence classes in Step 3 is an iterative grouping process. In each iteration step the time similarity between all pairs of classes is computed. The pair of classes which are most similar to each other are then combined into a new equivalence class. As a measure of similarity $\mathcal{S}$ between two equivalence classes, $\mathcal{E}_i$ and $\mathcal{E}_j$, Mangu *et al.* (2000) propose the following definition for the intra-word clustering step:

$$\mathcal{S}(\mathcal{E}_i, \mathcal{E}_j) = \max_{\substack{\mathfrak{r}_i \in \mathcal{E}_i \\ \mathfrak{r}_j \in \mathcal{E}_j}} \mathcal{O}(\mathfrak{r}_i, \mathfrak{r}_j)\, p(\mathfrak{r}_i)\, p(\mathfrak{r}_j), \tag{3.15}$$

where $\mathcal{O}$ stands for time overlap between arc $\mathfrak{r}_i$ and arc $\mathfrak{r}_j$ normalized by the sum of their time duration (from start time to end time of the arc). In Equation 3.15, $\mathcal{O}$ is weighted by the posterior probabilities of corresponding arcs to make the measure of similarity more robust against unlikely word hypotheses.

Iteration steps are repeated until no more classes can be merged. Upon completion, all arcs with overlapping time intervals are merged to one equivalent class. For computation of the confidence measure, posterior scores of all arcs within resulting classes are accumulated as shown in the example in Equation 3.16 (page 45).

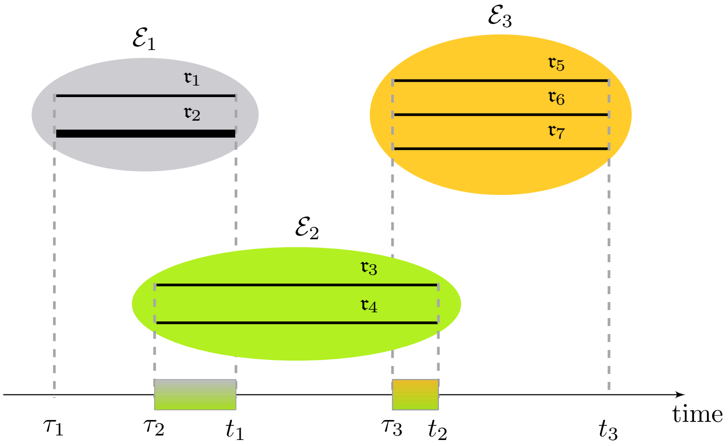

Figure 3.6: Initial equivalence classes, created in the second step of the consensus network algorithm, with inter-class time overlaps $\tau_2 - t_1$ and $\tau_3 - t_2$.

3.4.4 Summary

In order to point out differences between alternative computation methods of confidence scores described in Section 3.4.2 and in Section 3.4.3, let us take a closer look at Figure 3.4 and Figure 3.6. As we can see in Figure 3.4 on page 42, the confidence score of arc $\mathfrak{r}_2$ bold-plotted[1], has overlapping time intervals with three different arcs, namely with arcs $\mathfrak{r}_1$, $\mathfrak{r}_3$ and $\mathfrak{r}_4$. As a consequence the confidence score of arc $\mathfrak{r}_2$ is calculated as the sum of posterior scores of the arcs $\mathfrak{r}_{1-4}$ as shown in Equation 3.14 (page 42). On the other hand, according to the consensus hypothesis algorithm described in Section 3.4.3, all initial equivalence classes intersect in time mutually as shown in Figure 3.6. Due to the ordering rules given in Mangu *et al.* (2000), all three classes in Figure 3.6 can be merged during the intra-word clustering step described in Step 3 of the consensus hypothesis algorithm. As a result the confidence score of arc $\mathfrak{r}_2$ can be computed in this case as follows:

$$C_{sec}\left([w_2; \tau_2, t_2]\right) = \bigoplus_{i=1}^{7}{}_{\ln} C_{\mathfrak{r}_i}. \tag{3.16}$$

Obviously, this kind of calculation differs from that defined in Equation 3.14 for the same word graph topology and time overlap situation between word hypotheses. This fact that both methods could produce different confidence scores for the same word hypothesis was the motivation to compare their quality on different word graph topologies. Results of analyses carried out in the scope of this work are described in Section 3.7 on page 52.

[1] Note: $\mathfrak{r}_2$ marks best hypothesis word schematically as part of the best word sequence.

3.5 CM for Semantic Interpretation

Semantic interpretation of speech, also known as natural language understanding (NLU), is gaining more and more attention not only as a research object but also in practical implementation of speech applications; grammar formats supporting semantic interpretation are already standardized as part of W3C Recommendations W3C-SISR (2007). Speech dialog designs are converging more and more closely to the natural form of human communication and increasingly allow the use of natural language utterances for user input. In contrast to strictly prescribed user inputs, e.g. words contained in severely limited instruction sets, natural language utterances allow the user spontaneous interaction. The main advantages are increased user acceptance and the elimination or abbreviation of a learning phase for users to become accustomed to the speech application. The meaning of such complex natural language user inputs is represented by semantic results in order to allow their use in further data processing steps or also in dialog management (see Chapter 5 on page 79 for additional details). Meaning is often defined as a combination of words representing a semantic concept.

For speech applications, the confidence of semantic concepts must be measured by semantic confidences just as confidence of word hypotheses is measured by word confidences as described previously in this chapter. Sarikaya *et al.* (2005), for example, present two methods for modeling semantic information in a sentence. Statistical semantic features obtained by those techniques are incorporated to word- and concept-level posterior probability based confidence measures using a special word alignment technique. The posterior scores are computed on the word graph in the same way as described in Section 3.4.3; the method of converting word graphs to confusion networks (also called as sausages) proposed by Mangu *et al.* (1999). Hacioglu & Ward (2002) present a different technique for incorporating semantic information into confidence score computation which first converts the word graph into a concept graph and then calculates scores on the concept graph. Guillevic *et al.* (2002) propose a method for robust estimation of semantic confidence scores. The main focus of this approach is placed on generating task-independent CMs for dialog systems. For each semantic concept different predictor features are used and merged by a multi-layer perceptron.

Lieb *et al.* (2004) propose a straightforward approach to how posterior scores computed on word graphs can be used directly for confidence measurement of semantic concepts. This technique is applied to a one-stage automatic speech interpretation system, ODINS as described in Thomae *et al.* (2003). In addition to the best semantic result, ODINS generates alternative semantic hypotheses represented by nested word graphs of several hierarchy levels incorporating all necessary knowledge sources in one stage. The CM of a specific semantic concept is computed as the posterior score of the corresponding sublevel word graph belonging to a specific semantic concept.

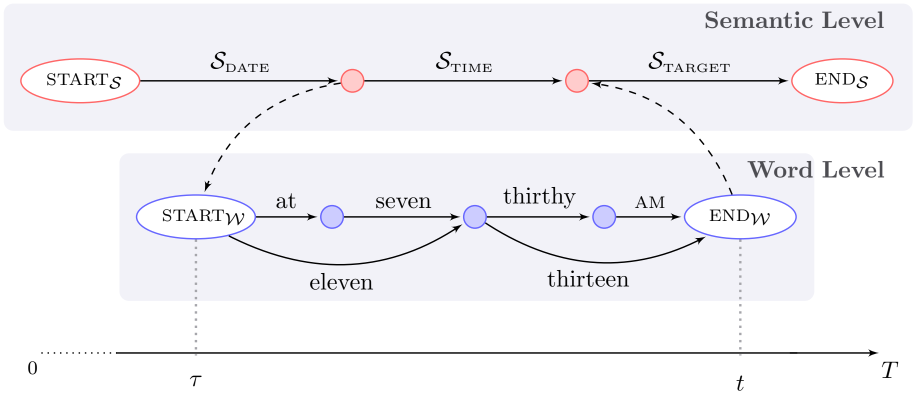

Figure 3.7: Nested graph of semantic concepts with an example sublevel word graph belongs to the concept label $\mathcal{S}_{\text{TIME}}$ with starting time τ and ending time t.

To make this more clear, Figure 3.7 on page 47 shows an example nested semantic graph with one sublevel word graph for the concept $\mathcal{S}_{\text{Time}}$. Each arc at the semantic level corresponds to a specific semantic concept and is connected to the start and end nodes of the corresponding sublevel word graph. For this reason Lieb *et al.* (2004) propose the computation of posterior scores of semantic concepts as the total posterior score of the underlying sublevel word graph. The computation of posterior scores described in Section 3.4.1 on page 34 can be simply applied to the word level of nested graphs computed using the forward-backward algorithm. At this point Equation 3.12 (page 41) can be used to determine the posterior score of each concept arc, e.g. the score for the semantic label $[\mathcal{S}_{\text{TIME}}; \tau, t]$ in Figure 3.7 for the speech fragment between time boundaries τ and t, where τ corresponds to the start node START$_{\mathcal{W}}$ and t to the end node END$_{\mathcal{W}}$ of the sublevel word graph. The observation score o in Equation 3.12 is also computed in this case on the entire graph at the semantic level from the starting node START$_{\mathcal{S}}$ in Figure 3.7 to the ending node END$_{\mathcal{S}}$ incorporating all sublevel word graphs.

As described above in Section 3.4.2 the performance of posterior scores can be significantly improved by summing up scores of alternative hypotheses on word graphs, taking time alignment information of similar hypotheses into account. Lieb *et al.* (2004) show that this method can also be applied successfully to the semantic scores computed on sublevel word graphs. In this case the posterior scores of those sublevel word graphs are summed up in the logarithmic space, which belong to similar concept arcs that intersect in time boundaries with the best hypothesis at the semantic level.

3.6 Utilization of Confidence Measurement

Decision of recognition quality made on the basis of reliable confidence measures can also be helpful for other tasks than the assignment of CM to specific hypotheses, i.e. tasks which generally have the capability to improve recognition performance not only in terms of accuracy but also in reducing time consumption and memory use. A dynamic pruning approach is presented in detail in Section 4 on page 59 which is based on confidence measurement. Section 5 on page 79 deals with the utilization of CM techniques for dialog management strategies in great detail. This section presents a survey of CM utilization techniques such as lattice rescoring and pruning, detection of out-of-vocabulary words (OOV) or unsupervised adaptation.

3.6.1 Rescoring and Pruning

Once a hypothesis' confidence is known it can be also used for rescoring resulting N-best lists. The task of a speech recognizer is, generally, to find the best word sequence for unknown speech data. Particularly when the recognizer runs under conditions that are mismatched, i.e. not sufficiently represented within the training material, such as fast speech, recognition performance can be improved by properly selecting one of the N-best output hypotheses. Through analysis of N-best hypotheses, also called as *N-best rescoring*, the performance of the recognizer can be improved because even the hypothesis which best matches the spoken utterance has not necessarily been scored as the best one during search. N-best rescoring, though, has the potential to spot it in the N-best list during a post-processing step. Furthermore, N-best rescoring is suitable for integrating those knowledge sources into the decision process, which are not available before the end of an utterance or for which integration into the search process would be computationally very expensive.

As shown in Fabian *et al.* (2001) experimentally, the integration of knowledge about the speech rate can be helpful for analysis and selection of N-best hypotheses. The speech rate can vary considerably between different utterances from the same speaker or even within a single utterance. Therefore a method is proposed which compares similarities between speech rates of spoken utterances and speech rates of hypotheses in the resulting N-best list. As part of a post-processing step, that hypothesis is selected from the N-best list whose speech rate is most similar to the original speech rate of the utterance. The speech rates are detected simply as phoneme or vowel rates either on hypotheses' contents, or to obtain the original speech rate of the utterance by using a simple phoneme recognizer.

Word graphs typically contain arcs of very low posterior scores which are negligible for computing total posterior score of a specific hypothesis. The *pruning* of such arcs can increase efficiency of computation tasks carried out on word graphs. Mangu *et al.* (2000) propose a pruning technique which removes all arcs from the word graph whose posteriors fall below a specific threshold which can be determined empirically. Similarly Lieb (2006) proposes a pruning approach which works on nested word graphs of multiple hierarchy, semantic and word, in the ODINS semantic decoder. This method is a straightforward implementation of the posterior based word graph pruning method proposed in Sixtus & Ortmanns (1999), where the pruning criterion of a specific word graph arc is the distance of its posterior score with respect to the average posterior score of word graph arcs belonging to the best path.

3.6.2 Rejection Techniques

As already mentioned earlier, the primary objective of rejection techniques is to assess the quality of recognition hypotheses under different expectations regarding user inputs and limitations in the recognizer's vocabulary. Out-of-vocabulary (OOV) detection, for example, has the main task of reliably spotting those recognizer inputs which are not part of its vocabulary. In contrast, *keyword potting* is intended to handle unconstrained recognizer inputs by rejecting all inputs other than a small number of task-specific keywords (non-keyword rejection). *Utterance verification*, on the other hand, deals with the rejection of incorrect hypotheses in general without distinguishing between reasons why they are incorrect.

OOV Detection

There are different ways to detect OOV words in speech recognition tasks, for example by modeling OOV words with a set of HMMs, also known as *filler models*, or using confidence thresholds for OOV decision. Incorporating CM in OOV detection generally has the potential to improve detection performance. Especially the use of posterior score (as discussed e.g. in Young, 1994b; Mengusoglu & Ris, 2005) or normalized log likelihood scores brings significant improvement in OOV detection as reported in Sun *et al.* (2003). Those methods generally apply word confidence by determining a threshold for OOV empirically on training sets. If resulting hypotheses fall below a specific threshold, those hypotheses are considered as OOV words. Another possibility is to combine different CM approaches into a final probability decision using an NN classifier.

In Ketabdar *et al.* (2007) a different, more sophisticated method is proposed which uses a two-channel ASR technique to detect OOV words. The main idea behind this technique is to discover unexpected words by comparing *out-of-context* and *in-context* hypothesis posteriors generated for identical input utterances. In-context posteriors are determined by a recognition channel using prior contextual knowledge sources such as pronunciation dictionary and word probabilities provided by the language model. On the other hand, out-of-context posteriors are computed through utilization of a simple phoneme recognizer without any prior lexical information. Posteriors of both channels are then compared by the *Kullback-Leibler* *divergence*[2] and an OOV is detected if in-context posteriors significantly deviate from out-of-context posteriors measured on phoneme level by passing through the entire utterance.

Keyword Spotting

Keyword spotting works on unconstrained speech inputs when specific keywords are to be separated from other, non-keywords. Due to its unconstrained nature, there is no language model available to describe input sentences, especially to mark semantic relations of keywords related to other words. Earlier works, such as Rose (1992), propose keyword spotting by modeling non-keywords explicitly by means of a so-called *garbage model*, also known as *sink* or *filler models*, in order to use them to compete with keyword models during decoding of unconstrained speech input. There is abundant literature dealing with this specific task; to mention a few examples: Boite *et al.* (1993) introduces a speaker independent approach for recognition over telephone lines using explicit garbage modeling. Junkawitsch *et al.* (1997) propose a CM-based method where keywords are detected via confidence maximization. It uses two different CM approaches: the negative logarithm of keywords' posterior probability and the likelihood ratio between the keyword model and an *anti-model*.

Utterance Verification

As stated above, the term *utterance verification* (UV) is understood as the process of detecting and rejecting the least reliable hypotheses according to the motto "no recognition is better than misrecognition" as illustrated by an example in Bouwman & Boves (2001). This is especially true for commercial speech applications which have to manage a wide range of dialects in user inputs while simultaneously avoiding misrecognition, often for reasons of security or data protection. Rose *et al.* (1998), for example, condition the word probability estimated by the language model upon acoustic confidences of the elements of its n-gram history. For utterance verification, the log likelihood ratio (LLR) as a measure of confidence has been shown to be generally useful as reported in Lee (1997) and Charlet *et al.* (2001). Evaluation results by combining LLR with other predictor features like speech rate factors or a lexical stress measure are shown in Bouwman & Boves (2001). The likelihood ratio testing (LRT) approach is shown in Jiang (2005), which provides good theoretical formulation of the utilization of confidence measurement for utterance verification. As stated in Jiang (2005), the LRT algorithm was originally motivated by the speaker verification problem but it could be also applied successfully to UV.

[2] The Kullback-Leibler divergence is also known as information divergence or relative entropy and is a measure of the difference between two probability distributions (see Kullback & Leibler, 1951).

3.6.3 Adaptation Methods

Speech applications developed for a wide range of user groups, e.g. in the banking or telecommunication sector, should provide acceptable performance in recognition accuracy from the outset. Therefore they are generally trained on huge amounts of training material gathered from a large number of different speakers who vary in gender, age and dialect. Further improvements for specific users are only possible by applying adaptation techniques with speech data collected from those users while they interact with applications. Supervised adaptation methods, such as reading huge amounts of predefined text, often cannot be used because they are non ergonomic, awkward and therefore not user friendly.

An interesting area of CM utilizations is that of unsupervised or semi-supervised adaptation techniques. Decisions regarding the quality of recognition results based on CM allow more reliable speech segments to be selected for adaptation in order to improve the performance of recognition models automatically or semi-automatically as reported in Kemp & Waibel (1999); Wallhoff *et al.* (2000); Charlet (2001); Goronzy *et al.* (2000) and by many others. Kemp & Waibel (1999), for example, evaluate the effect of the lattice-based posterior score as CM (referred as *gamma* CM in Kemp & Schaaf, 1997) against *perfect CM*[3]. It uses an interesting approach for unsupervised training: first, multiple recognizers are trained based on different subsets of the training material selected using different CM thresholds. Then the recognition results are combined weighted by word confidence.

In Wallhoff *et al.* (2000) a comparison is made between efficiency of supervised and unsupervised adaptations using maximum likelihood linear regression and frame-based discriminative training techniques. The CM is computed based on N-best list word density as proposed in Willet *et al.* (1998).

Charlet (2001) presents an incremental unsupervised adaptation method based on ranking of adaptation data according to their confidence scores and shows that utilization of CM for unsupervised adaptation brings significant improvement depending on the adaptation rate[4]. Charlet (2001) uses as CM the difference between the log-likelihood of the first and second candidates in an N-best decoding approach normalized by the length of the utterance, a measure proposed as quite effective in Willet *et al.* (1998).

As mentioned earlier in Section 3.3 on page 31, Goronzy *et al.* (2000) propose the usage of NN classifier to incorporate several confidence predictor features based on phoneme duration and acoustic score for computing CM. In Goronzy *et al.* (2000) it is also shown how the CM can be applied to a semi-supervised speaker adaptation technique while only those utterances are used for adaptation which are accepted by the confidence measure. Zhang *et al.* (2005) present an approach for semi-supervised

[3] The term *perfect CM* refers to hypotheses' tagging, correct or false, based on known transcription of the recognition result.

[4] The adaptation rate measures how important adaptation data are considered with respect to prior data.

training using several confidence measures such as LM-backoff-mode and posterior probability of different levels, i.e. utterance, word and frame level.

3.7 Impact of Lattice Density, Evaluation Results

As already mentioned in the introduction, reduction of the computation time, consumed by the decoding process of ASR systems is an important and topical issue in order to optimize runtime behavior, especially for embedded speech enabled systems. Here good results can be achieved, for example, by applying optimization techniques to the search process and also by producing word graphs with low densities. The latter saves time by retaining only a few alternatives during the Viterbi search which are then sufficient for building a word graph of low density. But the question remains how reduction in *word graph density*[5] (WGD) influences the quality of confidence measures computed on word graphs in a post-processing step.

This section shows the results of analyses carried out by investigating the influence of word graph density on the quality of posterior probability based confidence measures as described in Section 3.4.2 and in Section 3.4.3. In order to allow evaluation of both CM techniques, the HTK system was enhanced with these methods by the author. Carrying out the implementation as presented in Section 3.4.1 on page 36, the forward-backward algorithms working on the word graph were extended by both accumulation techniques: the *simple accumulation* of similar words with intersections in time alignment and the accumulation method based on the *consensus hypothesis* algorithm. Investigations were performed on two different speech corpora, on Verbmobil '96 and on NaDia (see Section 3.7.2 on page 53). Prior to the results, evaluation metrics used for analysis are discussed.

3.7.1 Evaluation Quantities

The *confidence error rate* (CER) is defined as the number of incorrectly tagged hypotheses divided by the total number of recognized words, as defined in Wessel *et al.* (2001). Tagging of hypotheses as *correct* or *false* is made according to whether hypotheses' confidence scores exceed a certain threshold or not. Those hypotheses whose posterior scores fall below the threshold are simply tagged as incorrect whereas all others are tagged as correct. Optimal setting of the confidence threshold was determined by minimizing the CER on a cross-validation corpus which must be clearly distinct from the evaluation corpus as shown in Section 3.7.2. This tagging strategy can lead to different types of classification errors of incorrectly tagged hypotheses: *false acceptance* (FA) and *false rejection* (FR).

The trade-off between FA and FR rates are depicted in form of *receiver operating characteristic* curves, also called as ROC curves, which are described in great detail

[5] The word graph density is defined as the total number of the word graph links divided by the number of spoken words.

in Appendix A on page 99. The ROC curves in Figure A.2 on page 102 are generated while varying the confidence threshold for given hypothesis confidence score distributions. For each threshold the error rates FA and FR are computed and pictured in a diagram. The false acceptance rate is defined as the percentage of incorrectly recognized words tagged as *correct* (accepted) because their confidence scores are higher than a certain confidence threshold. Analogously, the false rejection rate is the percentage of correctly recognized words tagged as *false* (rejected) by the threshold (see also in Falavogna *et al.*, 2002). The ROC curve allows an exploration of what happens to FA and FR while varying the position of the confidence threshold. If the confidence threshold is moved toward from higher to lower values, the number of false rejections will decrease as shown in Figure A.2 on page 102. Finally, it reaches a region where there is a remarkable decrease in false acceptance; the ROC curve flatten out if the confidence threshold is moved to very low values.

The baseline CER, which is independent of the word graph density, is computed on untagged recognition result sentences as the number of insertions and substitutions divided by the total number of recognized words. CER is similar to the definition of word error rate but it does not consider word deletions because the number of deleted words cannot be influenced anymore by the tagging threshold; they always remain deleted in hypotheses. Therefore, changes in CER caused by different tagging thresholds are independent of deleted words. In contrast, insertions and substitutions can be tagged as incorrect hypotheses because of their low confidences which fall below a specific tagging threshold. In this case recognition error can be reduced by correct tagging which corresponds to CER reduction. Thus baseline CER also marks the maximum confidence error rate.

3.7.2 Experimental Setup

Evaluations performed in the scope of this work were carried out on the commonly used speech recognition system Hidden Markov Model Toolkit (HTK) release 3.1. Details about its modules and algorithms are given in Young (1994a). To supply the needs of evaluations the code basis of the open source tool kit was modified by the author in order to implement necessary algorithms. For example for the computation of the posterior score the lattice post-processing tool HLRescore and its component HLat, part of the HTK library, were enhanced with appropriate methods. Also the accumulation techniques like the computation of the consensus hypothesis was implemented in HLat. In order to support confidence tagging and appropriate alignments in recognition results for CER computation the tool HResults was modified accordingly.

Verbmobil Speech Corpus

One of the speech corpora was used for evaluation is the German Verbmobil ’96 corpus which is described detailed in Bub & Schwinn (1996). This corpus is split into three data sets: training, cross-validation and test set which are strictly distinct from each other. The test set, also called as evaluation set, contains 343 sentences, i.e. 6428 words. For determining parameters settings empirically, e.g. scaling factors, a distinct cross-validation set was used which contains 599 sentences (i.e. 11577 words). For evaluations on the Verbmobil '96 corpus we used a bigram language model and a dictionary with 5343 entries. The training of acoustic models was performed on intra-word triphones by the use of the common parameter tying approach and additionally by applying the method of *mixture splitting* for further quality improvement of HMMs as described in Lieb (2006). Resulting HMMs consist of about 25000 state-tied Gaussian mixture components. The training corpus contains about 11000 utterances recorded from about 600 speakers. The set of acoustic feature vectors extracted from the speech signal for acoustic modeling consists of 39 components, which are 12 mel frequency cepstral coefficients (MFCCs), the normalized signal energy and the corresponding temporal derivations of first and second order.

NaDia Speech Corpus

The other speech corpus used for evaluations in the scope of this work is the NaDia speech corpus which was collected for the industry-funded research project NaDia ("Natürlichsprachliche Dialogführung für die Nutzung komplexer Informationsdienste im Automobil"). For this purpose spontaneously spoken utterances of 30 speakers were recorded in a Wizard-of-Oz simulation for an airport information domain. Also this corpus is split up in three distinct sets. The test data set contains 233 sentences, i.e. 1150 words, of 3 speakers and the cross-validation set consists of 320 utterances (1183 words) of 3 speakers. Analogously to the Verbmobil evaluation material, for the analyses on this corpus a bigram language model and intra-word triphone acoustic models were used. In this case the acoustic models had about 25000 state-tied mixtures and the dictionary had 640 entries. The acoustic models were initially trained on the Verbmobil training material and than adapted on the NaDia training set which contains 1446 utterances of 17 speakers (see Lieb, 2006).

3.7.3 Results of Simple Accumulation Method

Table 3.2 on page 55 shows confidence error rates detected for both evaluation corpora, Verbmobil and NaDia, where confidence scores of word hypotheses were computed by the simple accumulation method as described in Section 3.4.2. In order to produce different word graph densities, the recognition task over all utterances of a specific evaluation corpus was performed repeatedly using the recognition tool HVite which is part of the HTK system. Several recognition tasks were performed with different configuration settings varying the number of alternative hypotheses kept during Viterbi decoding. As a result word graphs of different densities were generated by the speech recognizer.

Verbmobil			**NaDia**		
WGD	**CER** [%]	**rel. CER** [%]	**WGD**	**CER** [%]	**rel. CER** [%]
17.7	19.7	27.8	30.9	10.9	29.2
206.5	16.7	38.8	383.5	10.2	33.8
736	16	41.4	1170.4	10.3	33.1

Table 3.2: Confidence error rates (CER) for different word graph densities (WGDs) for the evaluation corpora Verbmobil and NaDia, generated by the *simple accumulation* method. Rel. CER is computed relative to the baseline CER of 27.3 % for the Verbmobil corpus and of 15.4 % for NaDia corpus.

As we can see in Table 3.2, the word graph density obviously has an impact on confidence error rate for both evaluation corpora; lower CERs correspond to higher WGDs especially in case of the Verbmobil corpus. In other words, if the WGD becomes very low, there is a significant leak in performance of the posterior score. As a consequence, one can say that in practice one should be aware of generating word graphs for the usage of posterior scores depending on speech corpora.

As per the previous definition of the baseline CER, for the Verbmobil corpus it amounts to 27.3 %. According to Table 3.2 there is a relative CER reduction between 27.8 % and 41.4 % corresponding to the absolute values 19.7 % and 16 %. As far as the NaDia corpus is concerned, the baseline confidence error rate amounts to 15.4 %. The results in Table 3.2 show relative CER reductions between 29.2 % and 33.8 % depending on word graph density and corresponding to the absolute CERs of 10.9 % and 10.2 %. The ROC curves in Figure 3.8 on page 56 underline the observation that the performance of the confidence measure depends on the WGD. The diagrams show significantly better *equal error rates*[6] for higher word graph densities than for lower ones for both evaluation corpora.

3.7.4 Results of Consensus Hypothesis Method

Similarly to the above, Table 3.3 shows confidence error rates obtained for both evaluation corpora, Verbmobil and NaDia, computing confidence scores by means of the consensus hypothesis algorithm as described in Section 3.4.3. The CER baselines are the same as in the previous section for both speech corpora. Evaluations on the Verbmobil corpus result in relative CER reduction between 28.9 % and 40.7 % corresponding to the absolute CER values of 19.4 % and 16.2 % as shown in Table 3.3. For the NaDia corpus there is a relative CER reduction range from 27.3 % to max-

[6] The equal error rate is defined as the equal false rejection and false acceptance rate as described in Appendix A on page 99.

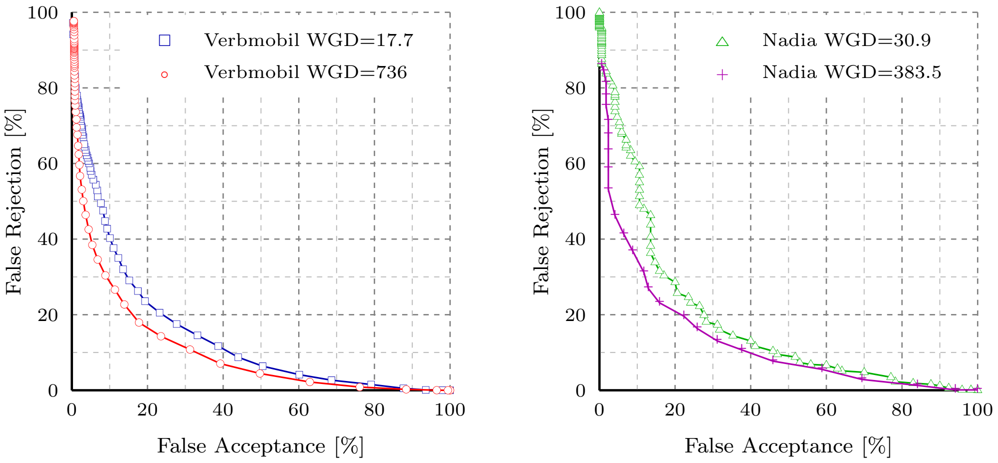

Figure 3.8: ROC curves for the evaluation corpora Verbmobil and NaDia, for different word graph densities (WGDs) using the *simple accumulation* method.

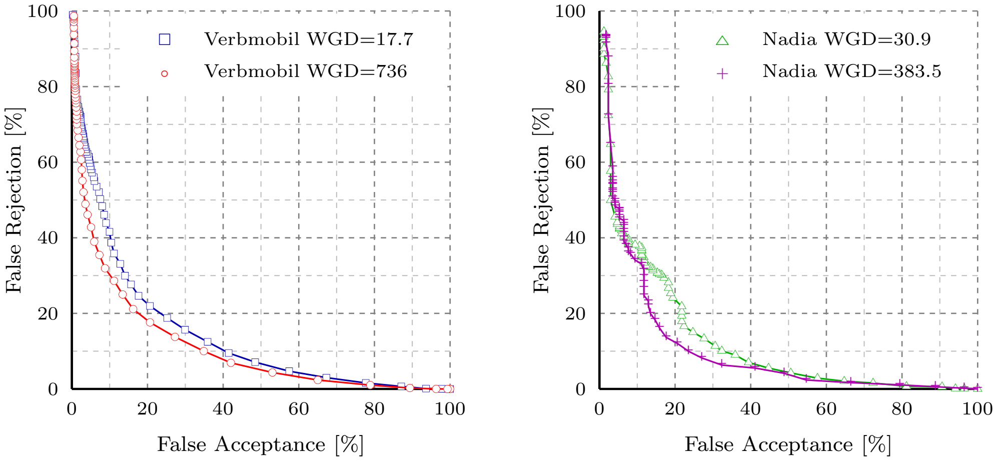

Figure 3.9: ROC curves for the evaluation corpora Verbmobil and NaDia, for different word graph densities (WGDs) using the *consensus hypothesis* method.

Verbmobil			NaDia		
WGD	**CER** [%]	**rel. CER** [%]	**WGD**	**CER** [%]	**rel. CER** [%]
17.7	19.4	28.9	30.9	11.2	27,3
206.5	16.7	38.8	383.5	10.3	33.1
736	16.2	40.7	1170.4	10.1	34.4

Table 3.3: Confidence error rates (CER) results for different word graph densities (WGDs) and for the evaluation corpora, Verbmobil and NaDia generated by the *consensus hypothesis* method. Rel. CER is computed relative to the baseline CER of 27.3 % for the Verbmobil corpus and of 15.4 % for NaDia corpus.

imum 34.4 % corresponding to absolute values of 11.2 % and 10.1 % in Table 3.3. Figure 3.9 on page 56 shows results comparable to Figure 3.8 on page 56 and again obvious dependence on word graph density for both ROC curves.

3.7.5 Summary

In conclusion we can say that both accumulation methods of posterior scores perform similarly and achieve very good results as far as the reduction in confidence error rate is concerned. However, evaluation results in this section also show a clear dependence on word graph density for both test corpora. Obviously, higher WGDs are more suitable for the accumulation methods and deliver lower CERs than WGDs of low density. Regarding the computation time consumption of the two methods, the evaluation tasks show that the time consumption of the consensus hypothesis method is many times higher than that of the simple accumulation method. This is because the *intra-word clustering* step (see Section 3.4.3) tends to consume a huge amount of time especially in case of higher word graph densities. It seems that even on modern PCs, it is not feasible to compute confidence scores in a way that would fulfill real-time requirements, due to the computation complexity of $O(T^3)$ (where T is the length of the utterance). Therefore, the best choice for practice would be to apply the simple accumulation method for confidence score calculation in order to obtain both the best confidence quality, by working on word graphs of high densities, and acceptable runtime behavior. The results above could also be adopted into practice as a way of adjusting the optimal word graph density depending on the speech application in order to minimize confidence error rates.

3.8 Limitations

In spite of all the research efforts put into CM development over recent decades, it still remains a challenge to apply these CM approaches to practical applications successfully. We still encounter serious problems and severe performance degradation almost every time speech recognition systems are integrated into real spoken-dialog applications following a laboratory development phase. Such difficulties stem primarily from mismatched situations between assumptions made in the lab environment and real world conditions like unexpected user behavior, inconsistent acoustic channel conditions e.g. distortion, different kinds of background noise or the low transmission quality of mobile phones.

Even the best ASR systems currently available on the commercial market are not in a position to provide high quality CMs which are robust enough to make reliable decisions in all dialog situations. Especially in large vocabulary tasks, CMs often fail to provide a solid basis for detecting OOV words as unexpected user input; at most, they allow the implementation of rudimentary rejection techniques. In spite of these known issues, Chapter 5 on page 79 shows in detail how different CM techniques can be utilized successfully nonetheless for decisionmaking in dialog systems.

Chapter 4

Confidence-Guided Pruning

Making speech recognition more efficient in computation time is still an important and topical issue, in particular for embedded speech recognizers with limited memory capacity and CPU power. More and more speech applications will be deployed in embedded systems which often have only a limited computation capacity. In order to meet users' expectations we need to ensure acceptable runtime behavior by minimizing system response delays. Improved pruning algorithms for automatic speech recognition lead directly to a more efficient recognition process.

Herein lies the motivation to analyze commonly used speech recognition algorithms in order to optimize their efficiency in computation time. The most time consuming part of the recognition process is the search process. Depending on the complexity of the search network, managing alternative hypotheses for each time frame can be prohibitive in terms of processing time and memory resources. The Viterbi search space size of HMM-based automatic speech recognition systems usually increases non-linearly with the vocabulary size and this is why different pruning strategies have been already proposed to reduce the time consumption of the recognition process as described in Section 2.3.2 on page 22.

Improved pruning efficiency accelerates the search process and leads to a more time-efficient speech recognition system. Proven confidence measures based on *posterior score* (C_P) or *normalized log likelihood score* (C_{NLL}) allow an assessment of the classification correctness at phone or word level during the search process as described in Williams & Renals (1997); Kamppari & Hansen (2000); Fabian *et al.* (2003). Especially in recent years, several pruning algorithms have been introduced concerning confidence measurement as a guide for pruning techniques (among others, Ortmanns *et al.*, 1997; Renals & Hochberg, 1999; Liu *et al.*, 2001; Abdou & Scordilis, 2003). In Abdou & Scordilis (2003) a complex look-ahead technique is presented to manage HMM-specific thresholds of posterior confidence scores in order to support the pruning procedure. This however, could result in an enormous management effort in the case of the thousands of triphones which are often used in current ASR systems. The posterior score based look-ahead approach proposed in Ortmanns *et al.* (1997) operates on neural network (NN) and cannot be deployed to an HMM framework easily. All of these pruning techniques generally use constant pruning thresholds over the entire search procedure.

In the course of this work a new dynamic pruning technique was developed which optimizes the well-known probability-based pruning (beam width) by utilization of confidence measurement. In this work normalized hypothesis scores are used to guide the beam width of the pruning process dynamically, frame by frame, over the entire utterance. Compared with classical pruning techniques, like fixed beam pruning and histogram rank pruning, significantly better results can be achieved regarding the time consumption of the recognizer. In this chapter a novel pruning approach is introduced, which controls the beam width B_{set} of HMM-based Viterbi search process framewise. The decision as to the appropriate threshold at each time frame is based on the utilization of normalized log likelihood confidence measures (see also in Fabian *et al.*, 2005).

4.1 The Confidence Measure

The confidence-guided pruning approach is a combination of the widely used classical probability-based beam pruning technique and runtime confidence measurement. As described in Section 2.3.2 on page 22, probability-based pruning uses a constant threshold B_{set} as the beam width of the Viterbi search process at each time frame of the whole utterance. Both B_{set} of the probability-based pruning and N_{max} of the rank-based approach are predefined thresholds which have to be justified during cross validation tests. In order to improve efficiency, however, these thresholds could be adjusted dynamically to fit time-variant requirements by taking variable search quality into consideration utilizing an appropriate confidence measure. As a result, beam width $B(t)$ is set dynamically at each time frame t according to the confidence estimation.

As discussed in Chapter 3, the best basic metric for confidence measurement is the posterior probability; its mathematical formulation for the class c_i is:

$$C_P(c_i|x) = p(c_i|x) = \frac{p(x|c_i)\,p(c_i)}{p(x)}. \tag{4.1}$$

C_P can be thought of as the ratio of a proposed probability $p(x|c_i)\,p(c_i)$ of a model c_i and the observation probability $p(x)$. The proposed probability is the product of the acoustic model probability and the prior model probability and it reflects how well the class c_i fits the observation x. The observation probability $p(x)$ can be also called the *catch-all probability* since it describes how well the acoustic models account for the acoustic observation in general.

4.1.1 Normalized Log Likelihood Score

This work uses a CM which was developed based on a slight variation of C_P and is called the *normalized log likelihood score*, C_{NLL}. It is defined as the logarithm of C_P normalized by the prior class probability $p(c_i)$ and can be computed using the following formula:

$$C_{NLL}(c_i|x) = \ln\left(\frac{p(x|c_i)}{p(x)}\right). \tag{4.2}$$

The observation probability in the denominator can be expressed as follows:

$$p(x) = \sum_{j=1}^{N_c} p(x|c_j)\, p(c_j), \tag{4.3}$$

where N_c is the total number of classes, $p(x|c_j)$ is the acoustic model probability given the class c_j and $p(c_j)$ is the prior probability of the class c_j. Using this formula, C_{NLL} can be also formed as follows:

$$C_{NLL}(c_i|x) = \ln\left(\frac{p(x|c_i)}{\sum_{j=1}^{N_c} p(x|c_j)\, p(c_j)}\right), \tag{4.4}$$

or as:

$$C_{NLL}(c_i|x) = \ln\left(p(x|c_i)\right) - \ln\left(\sum_{j=1}^{N_c} p(x|c_j)\, p(c_j)\right). \tag{4.5}$$

As far as C_P is concerned, its values range between 0 and 1. Where lower values, close to 0, indicate low confidence since this means that there are models, other than c_i, which correspond to the acoustic observation x as closely ore more closely than c_i itself. Higher values, close to 1, on the other hand, indicate high confidence because in such cases the proposed model c_i best fits the acoustic observation x.

Regarding the normalized log likelihood score C_{NLL}, its value range differs from that of C_P since the prior probability $p(c_i)$ in the numerator of Equation 4.1 was removed and therefore the value range of the log operation becomes $[-\infty, ln(p(c_i))]$. C_{NLL} is expressed in the logarithmic space and can be viewed as a zero-centered confidence score where positive scores indicate high confidence and negative scores low, which means that the more positive C_{NLL}, the higher the indicated confidence.

Earlier works (e.g. Kamppari & Hansen, 2000) provide adequate results on the good quality of C_{NLL} and classify it as a reliable confidence measurement. This fact was the main motivation to utilize C_{NLL} of the hypotheses in order to develop a confidence-based pruning technique and apply it to the Viterbi beam search. Although it is possible to compute C_{NLL} of each hypothesis for each time frame of the search procedure using Equation 4.5, the resulting confidence values cannot be used

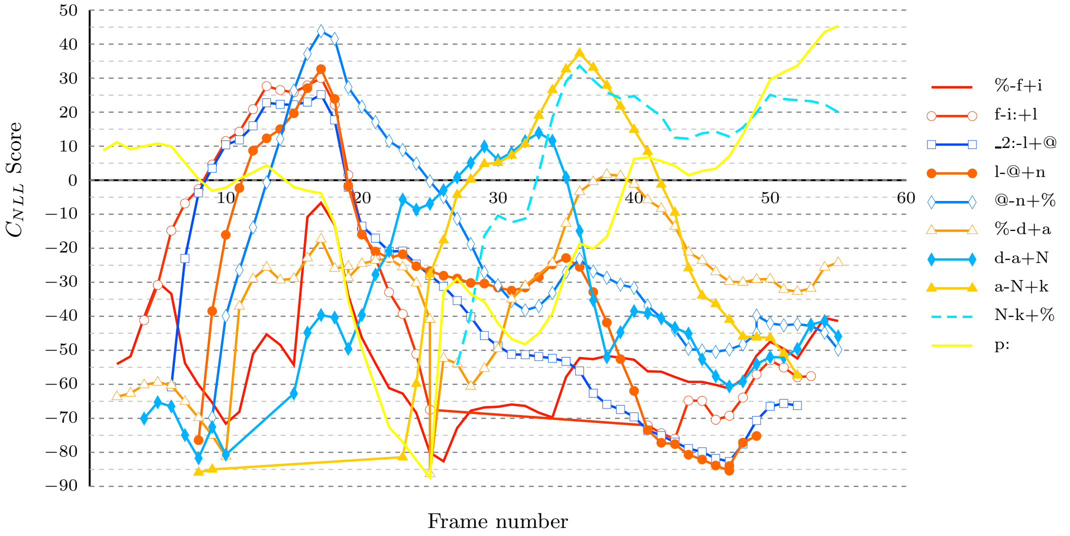

Figure 4.1: Course of triphones' C_{NLL} confidence scores during the utterance 'Vielen Dank!' (English: 'Thank you!').

directly to improve the effectivness of the pruning. This is because the confidence of a specific hypothesis varies widely over time. As shown in Figure 4.1 on page 62, C_{NLL} values of different triphones change during an utterance depending on which part of the acoustic information best fits a specific hypothesis containing the most appropriate triphone. A particular hypothesis could be pruned at a specific time frame because of its low confidence, even if this hypothesis would become the best at the end of the utterance.

In order to overcome such local time-variant effects, most known pruning techniques work on accumulated quantities. The simple classical probability-based beam width, for example, uses accumulated hypothesis scores as the basis for the pruning decision. Similarly, confidence-guided pruning also uses a confidence measurement of accumulated values of hypotheses which need to be computed step by step during the search process. Therefore the accumulated normalized log likelihood score C_{acc} is defined based on Equation 4.5 as the difference between the accumulated hypothesis likelihood score and the accumulated observation's score computed for each time frame t of the utterance:

$$C_{acc} = \sum_{t=1}^{T} \ln\left(p(x_t|c_t)\right) - \sum_{t=1}^{T} \ln\left(\sum_{j=1}^{N_c} p(x|c_j)\, p(c_j)\right). \tag{4.6}$$

As far as the pruning decision of a hypothesis is concerned, for each time frame the confidence score of each active hypothesis needs to be classified as good or bad based on an appropriate confidence threshold. Unfortunately, the confidence score, computed by Equation 4.6, does not allow specification of such a pruning threshold because its value increases continuously from frame to frame due to the steadily increasing difference between the accumulated best hypothesis score

$$\sum_{t=1}^{T} ln\left(p(x_t|c)\right)$$

and the accumulated observation probability

$$\sum_{t=1}^{T} \ln\left(\sum_{j=1}^{N_c} p(x|c_j)\,p(c_j)\right).$$

This is because the score of the best hypothesis is higher than the observation score for each time frame; Equation 4.6 accumulates the differences and doing so it does not allow assignment of a specific confidence threshold for the pruning decision.

For reasons mentioned above, a modified normalization is used for the computation of C_{acc} score, namely the combined maximum of the accumulated $ln\left(p(x_t)\right)$ and the best word end likelihood $W_{T,best}$:

$$C'_{acc} = \sum_{t=1}^{T} \ln\left(p(x_t|c_t)\right) - \max(\sum_{t=1}^{T} \ln\left(p(x_t)\right)|W_{best,t}). \tag{4.7}$$

Equation 4.7 allows to generate a normalization quantity which can be used for each time step to compute the confidence score of hypotheses. Figure 4.2 on page 64 shows a diagram as an example of normalized hypothesis score C'_{acc} of the best hypothesis plotted at each time frame. As we can see in the diagram the curve of the normalized score (dashed line) depends on the time frame. Especially high local maximums appear in correlation to pauses in the utterance.

In the classical pruning case, such as probability-based beam width, an appropriate constant threshold needs to be defined on a specific training set. This constant threshold should allow effective pruning of superfluous hypotheses and on the other hand it should also allow the best hypothesis to be retained from the beginning of the utterance until the end. Such a constant beam width would correspond to a horizontal line in Figure 4.2 on page 64 at a specific score level of e.g. 200, meaning a constant distance of score 200 from the best hypothesis score. In contrast to this the C'_{acc} dynamic approach allows the usage of a constant threshold B_{set} relative to the normalized score. As shown in Figure 4.2 on page 64, at each time frame only those hypotheses are kept whose scores are no less than a certain threshold from the score of the best hypothesis, computed as:

$$B(t) = B_{set} + C'_{acc}(t). \tag{4.8}$$

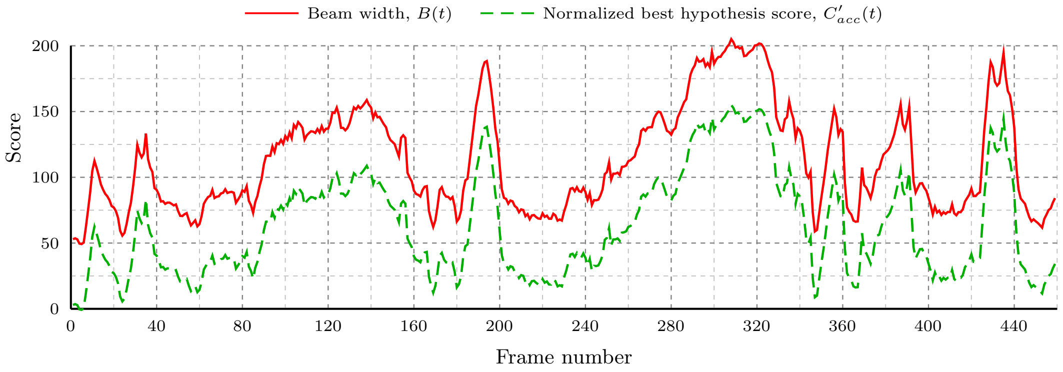

Figure 4.2: Example for the course of C'_{acc} and beam width $B(t)$ during the appointment negotiation utterance (Verbmobil speech corpus) 'Ja genau, lassen wir gleich die letzte Woche im März, prima!' (English: 'That's right, let's keep the last week in March, great!'). $B(t)$ is computed with $B_{set} = 50$ as per Equation 4.8.

Further optimization of the confidence-guided (CG) pruning approach can be achieved if the constant threshold B_{set}, which is set relative to the normalized score, is not a constant value but is computed dynamically. For this purpose the pruning threshold is varied depending on the current value of the normalized score C'_{acc} itself which indicates the observation quality of acoustic models. Low scores indicate poor certainty of the momentary best hypothesis; therefore the beam width should be increased in order to reduce the risk of pruning relevant hypotheses. Greater C'_{acc} scores, on the other hand, indicate good confidence of the best hypothesis and therefore the dynamic beam width should be decreased in order to improve efficiency.

This kind of dynamic lift ΔB (dotted line in Figure 4.3 on page 65) compresses $B(t)$ from Figure 4.2 somewhat; the result is plotted as solid line $B_{dyn}(t)$ in Figure 4.3. To implement the dynamic lift $\Delta B(t)$ a modified *sigmoid function* is used to allow control of the beam width between appropriate upper and lower thresholds:

$$\Delta B(t) = T_{upp} - \frac{T_{low}}{1 + e^{(\alpha - C'_{acc,t})/\beta}}. \tag{4.9}$$

The parameters α and β in Equation 4.9 can be determined using a cross evaluation corpus. A reasonable setting for the experiments presented in this work was $\alpha = \beta = 20$. The results of this dynamic approach for different T_{upp} and T_{low} are shown in Section 4.2. The dynamic threshold for the pruning decision is computed in this case as follows:

$$B_{dyn}(t) = \Delta B(t) + C'_{acc}(t). \tag{4.10}$$

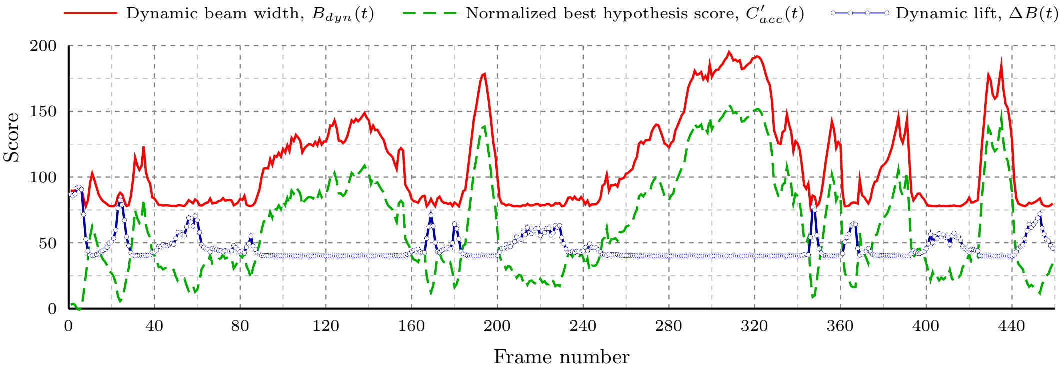

Figure 4.3: Example for dynamic beam width $B_{dyn}(t)$ during the appointment negotiation utterance (Verbmobil speech corpus) 'Ja genau, lassen wir gleich die letzte Woche im März, prima!' (English: 'That's right, let's keep the last week in March, great!'). Also the courses of C'_{acc} and the dynamic lift $\Delta B(t)$ are plotted.

In order to summarize the above, the computational steps of the *confidence-guided dynamic* (CGD) pruning threshold are as follows:

Step 1: Computation of C'_{acc} using Equation 4.7

Step 2: Calculation of the dynamic lift $\Delta B(t)$ using Equation 4.9

Step 3: Computation of the dynamic pruning threshold $B_{dyn}(t)$ as defined in Equation 4.10 for each time frame

The dynamic pruning threshold $B_{dyn}(t)$ computed in Step 3 is used directly as the pruning threshold for the Viterbi search process. Figure 4.4 on page 66 shows the schematic block diagram of the CGD pruning approach in order to make the implementation details more clear. CGD pruning computes beam width $B_{dyn}(t)$ of pruning dynamically in accordance with the confidence assessment of the best hypothesis. The Δ estimator is responsible for computing the dynamic lift $\Delta B(t)$ for each time frame based on the confidence score of the best hypothesis $C'_{acc}(t)$.

The main challenge in computing C'_{acc} in HMM-based systems is to attain a correct assessment of the observation probability, $p(x)$. This is because HMM-based systems generally do not have dedicated models for this purpose. The computation of $p(x)$, the catch-all score, requires the calculation of the emission of all HMMs which could cost large amounts of time. In Kamppari & Hansen (2000) a technique is proposed for managing this problem by reducing the catch-all model's size in terms of the number of Gaussian components.

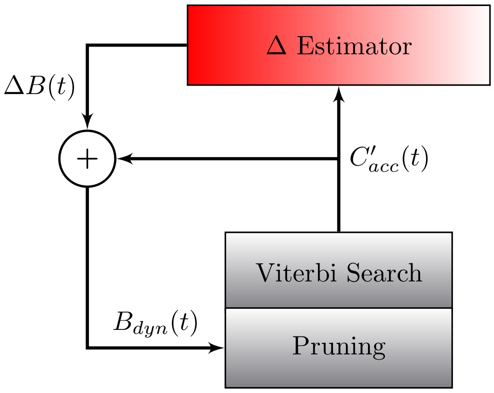

Figure 4.4: Schematic overview of the confidence-guided dynamic (CGD) pruning method.

4.1.2 Observation Probability in HMM Networks

As described in Chapter 2, HMM-based speech recognition systems generally omit the computation of the normalization quantity $p(x)$ in Equation 4.2 (page 61), the observation probability, in order to reduce computation time and memory usage during the search process. This is reasonable if only the best hypothesis needs to be found among all competitors. The computation of the confidence measurement C_{NLL}, however, requires the observation probability as normalization quantity and therefore it must be computed for the confidence-guided pruning approach.

The main objective of pruning in the first place is to reduce computation effort; therefore, it can be efficient only if the additional computation effort needed for pruning is negligible compared to the resulting savings effect in the search process. Therefore highly efficient methods for computing of the observation probability are very important to the realization of the CGD pruning.

Given an acoustic observation x the observation probability can be expressed as follows:

$$p(x) = \sum_{j=1}^{N_c} p(x|c_j)\, p(c_j), \tag{4.11}$$

where N_c is the total number of classes, $p(x|c_j)$ is the acoustic model probability given the class c_j and $p(c_j)$ is the prior probability of the class c_j. HMM-based speech recognition systems generally do not have dedicated models for the computation of the observation probability $p(x)$, and the state prior probability $p(c_j)$ is also unknown since only prior probabilities of higher word unit levels, such as words, are used to compute the global word sequence probability $p(W|X)$, see Equation 2.3 (page 9).

State Prior Probability Estimation

In the scope of this work, the class prior probability $p(c_j)$ in Equation 4.11 is estimated empirically by performing the following computation steps:

Step 1: The time consumption of each class/HMM was determined by running the Viterbi search on the Verbmobil '96 training data in order to determine the best alignment down to the state level. Consumption time was measured in the number of frames while a specific state was active in the total data set. This procedure was carried out using the HVite recognizer, part of the HTK toolkit. Since HVite does not provide this information by default, it was slightly modified by the author to count state activation time during alignment.

Step 2: The prior probability is then estimated for each class by the fraction of the time t_{c_j} while the HMM c_j was active over the total training data set duration:

$$p(c_j) = \frac{t_{c_j}}{\sum_{i=1}^{N_c} t_i}, \tag{4.12}$$

where t_i is the activation time for class i and N_c is the total number of classes.

Catch-all Model Generation

The computation of the catch-all score based on Equation 4.11 requires the calculation of the likelihoods of all model classes N_c at each time frame. Using this formula directly would be very time costly and therefore would not allow implementation of an efficient pruning technique. This is because the main saving effect of pruning is to exclude unneeded hypotheses from the search and consequently to omit the computation of likelihoods of superfluous models.

To resolve this conflict and to compute the catch-all score very efficiently, one global *catch-all model* of reduced size is used (see Kamppari & Hansen, 2000). The goal of building such a low complexity model is to find exclusively those Gaussians which have a non-negligible effect on the computation of the observation probability.

The probability $p(x|c_j)$ in Equation 4.11 can be expressed in more detail with additive mixture components as follows:

$$p(x|c_j) = \sum_{k=1}^{N_G(j)} w_{k,j}\, g_{k,j}(x), \tag{4.13}$$

where $N_G(j)$ is the number of Gaussians used to represent the model for class c_j, $w_{k,j}$ is the weight for the kth Gaussian of the jth class and $g_{k,j}$ is the Gaussian's probability for the observation x. The additive description above implies that the number of Gaussians in the catch-all model, N_{GCA}, can be expressed as the sum of the number of Gaussians modeling all classes:

$$N_{GCA} = \sum_{j=1}^{N_c} N_G(j), \tag{4.14}$$

where N_c is the number of classes.

Therefore, the global catch-all model is initialized with the pool of Gaussians of all HMM states. The HMMs used for the experiments carried out for this work consists of about 25000 Gaussian mixture components. The computation of $p(x|c_j)$ for each time frame using this highly complex catch-all model would be time costly and therefore very inefficient. In Kamppari & Hansen (2000) a method was proposed which reduces the catch-all model size by approximating a smaller number of Gaussians. The process of reduction of catch-all model size is an iterative bottom-up clustering process. In each iteration step, two Gaussians which are most similar to each other are found and then combined into a new one. As the measure of similarity, the weighted *Battacharyya distance* is used, which is generally defined as follows:

$$D_{Batt} = -\ln \int \sqrt{p_1(x)\, p_2(x)}\, dx. \tag{4.15}$$

The Battacharyya distance is a measure of overlap between two probability distributions and its values range between 0 and ∞ corresponding to full and no overlap. The specific implementation of D_{Batt} for Gaussians is as follows:

$$\begin{aligned} D_{Batt} = {} & \frac{1}{8}\,(\mu_1 - \mu_2)^T \left(\frac{\Sigma_1 + \Sigma_2}{2}\right)^{-1} (\mu_1 - \mu_2) \\ & + \frac{1}{2} \ln \left(\left| \frac{\Sigma_1 + \Sigma_2}{2} \right| \cdot |\Sigma_1|^{-1/2}\, |\Sigma_2|^{-1/2} \right), \end{aligned} \tag{4.16}$$

where μ_1 and μ_2 are the means of the Gaussians and Σ_1 and Σ_2 the covariance matrices. D_{Batt} is scaled to compress the acoustic space so that the entire acoustic space is covered with acceptable resolution using weights of the Gaussians w_1 and w_2:

$$D_{scale} = \sqrt{\frac{w_1^2 + w_2^2}{2 w_1 w_2}}. \tag{4.17}$$

This scaling prevents a single high covariance Gaussian from absorbing neighboring Gaussians while outliers remain unabsorbed. In HMM systems, these weights can be computed based on the weights of the mixture distribution functions. If w_1 and

w_2 are similar then Equation 4.17 becomes 1. In case of high difference however, $w_1 \gg w_2$ or $w_1 \ll w_2$, the value of D_{scale} goes to ∞. The scaled distance measure between Gaussians becomes:

$$D_{BS} = D_{Batt} D_{scale}. \tag{4.18}$$

After the distance between each pair of Gaussians is computed the pair with the lowest distance, minimum D_{BS}, is combined to a new Gaussian based on Equation 4.19, Equation 4.20 and Equation 4.21. The weights of the most similar Gaussians are summed up to determine the weight for the new Gaussian:

$$w_{new} = w_1 + w_2. \tag{4.19}$$

The mean value of the new Gaussian, μ_{new}, is the weighted sum of the mean values of the parents for each dimension $1 < d < D$ and is computed as follows:

$$\mu_{new,d} = \frac{w_1}{w_1 + w_2}\mu_{1d} + \frac{w_2}{w_1 + w_2}\mu_{2d}. \tag{4.20}$$

The variance of each dimension d of the new Gaussian, Σ_{new}, becomes the weighted sum of the parent Gaussians adjusted with the mean values:

$$\begin{aligned} \Sigma_{new,d} = &\frac{w_1}{w_1 + w_2}\Sigma_{1d} + \frac{w_2}{w_1 + w_2}\Sigma_{2d} \\ &+ \frac{w_1}{w_1 + w_2}\frac{w_2}{w_1 + w_2}\left(\mu_{1d} - \mu_{2d}\right)^2 . \end{aligned} \tag{4.21}$$

After a new combined Gaussian is computed it is added to the pool of Gaussians of the catch-all model and the Gaussians from which the new one was created are removed. This iteration is repeated as long as required to achieve the desired compression ratio of the acoustic space.

Due to this steps of model compression theoretical performance in acoustic modeling is given up since Equation 4.1 (page 60) is slightly altered. C_P is no longer constrained to be in the range $[0, 1]$. Instead, it falls in the range $[0, \gg 1]$ and therefore in order to be able to use the altered C_P as a basis for confidence decision some mapping mechanism, e.g. non-linear transformation, is needed for scaling. As far as C_{NLL} is concerned, its score is in the log domain, where even large variations in value range are automatically scaled to a reasonable range. Therefore, this work utilizes the normalized log likelihood score as the basis of the confidence-based pruning technique for the confidence measure in order to avoid the use of an additional scaling mechanism.

As presented in Kamppari & Hansen (2000) and Abdou & Scordilis (2003) the catch-all model of reduced complexity allows acceptable estimation of the observation probability, $p(x)$, even with a compression ratio of about 95 %. Figure 4.5 shows

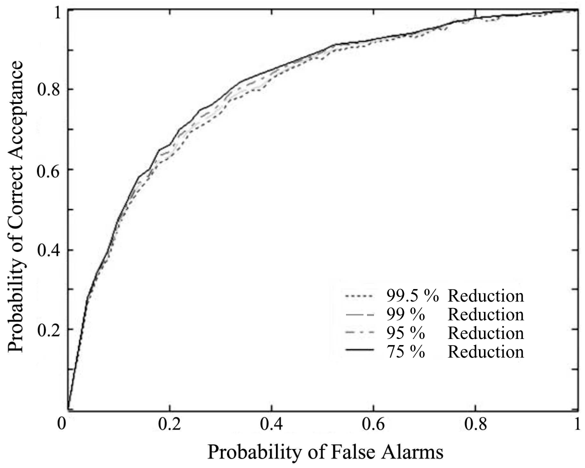

Figure 4.5: Relative ROC of the performance of reduced-complexity catch-all models as presented in Abdou & Scordilis (2004).

ROC curves, the trade-off between incorrectly and correctly accepted phoneme hypotheses, for different catch-all model compression ratios. Only a slight performance degradation is expressed even by a reduction in the complexity of the catch-all model by more than 90 %. Based on these findings the evaluations for this work were also carried out with a catch-all model of the compression ratio of 95 %. The complexity of the acoustic model could thus be reduced from about 25000 Gaussians to about 1000. As far as the implementation details are concerned, the computational effort of estimating $p(x)$ using the catch-all model of highly reduced complexity is negligible compared to that of using all Gaussians of all models. Therefore this catch-all approach is applicable for the confidence-based pruning technique since its savings for the search effort are vastly greater than the additional computational costs as proven by the results presented in the next section.

4.2 Experiments and Results

The goal of the experiments presented in this section was to evaluate the capability of the confidence-guided dynamic pruning approach to accelerate the search process of an ASR system. For this purpose several tests were carried out on the Verbmobil '96 evaluation data using different pruning techniques and parameters. The results are presented in Figure 4.6 on page 71 and Table 4.1 on page 72. All tests were performed on all utterances of the evaluation corpus Verbmobil '96 which is described in detail in Section 3.7.2 on page 53.

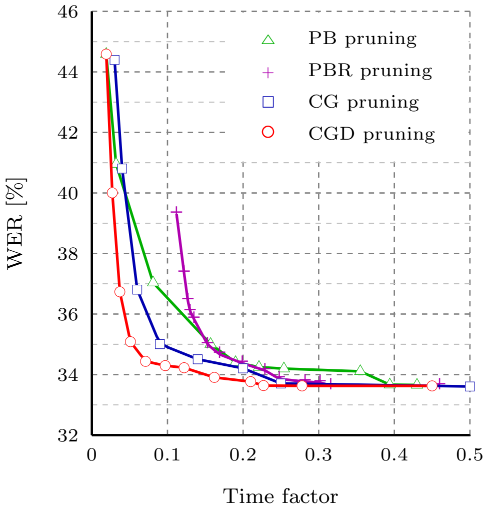

Figure 4.6: Word error rates (WERs) of different pruning techniques, classical and confidence-guided pruning, depending on time factor[1]: probability-based (PB) beam width, probability-based rank (PBR), confidence-guided (CG) and confidence-guided dynamic (CGD) pruning.

Figure 4.6 shows *word error rates* (WER) depending on the *time factor*[1] and allows the comparison between results of confidence-guided pruning techniques with classical pruning methods. Four different pruning techniques were evaluated, they are as follows:

- *probability-based* (PB) beam width pruning is the classical approach setting the beam width of the Viterbi search based on best hypothesis score and a specific threshold
- *probability based rank* (PBR) pruning limits the maximum number of active hypotheses, such as histogram rank pruning, in combination width PB pruning
- *confidence-guided* (CG) pruning based on Equation 4.8
- *confidence-guided dynamic* (CGD) pruning based on Equation 4.10

The curve of PB pruning in Figure 4.6 was determined by computing the WER for the evaluation corpus using different B_{set} values in a range of [80-250]. Greater B_{set} values lead to lower WER but a higher time factor. The combination of beam width and rank pruning, PBR pruning, was evaluated by keeping B_{set} at 210 and varying N_{set}, the maximum number of active hypotheses is allowed, in the range of [500-9000]. The curve of CG pruning was found using static beam width B_{set}

[1] The time factor is defined as the ratio of ASR time consumption with a particular pruning parameter setting to ASR time consumption without any pruning.

Pruning method; parameters	WER [%]	Time factor
PB; $B_{set} = 250$	33.63	0.43
PB; $B_{set} = 150$	34.40	0.19
PBR; $B_{set} = 210, N_{set} = 9000$	33.63	0.32
PBR; $B_{set} = 210, N_{set} = 2000$	34.37	0.20
CG; $B_{set} = 200$	33.63	0.50
CG; $B_{set} = 90$	34.50	0.14
CGD; $T_{upp} = 110, T_{low} = 40$	33.63	0.23
CGD; $T_{upp} = 110, T_{low} = 70$	34.43	0.07

Table 4.1: Word error rates (WERs) and the corresponding time factors[1] (see page 71) of CG and CGD pruning techniques in comparison with different classical pruning methods.

relative to the normalized score of the best hypothesis in a range of [55-200]. The curve of CGD pruning was plotted using $T_{upp} = 110$ and different T_{low} in a range of [20-70] (see Equation 4.9 for details).

To conclude, the CGD pruning approach outperforms all other methods significantly as shown in Figure 4.6. The time factor of the ASR using CGD pruning could be decreased to 0.23 without increasing WER. Furthermore, if an increase of about 1 % in WER is acceptable, CGD pruning achieves a time factor of 0.07 which corresponds to acceleration of the ASR by about 14 times (reciprocal time factor). In comparison, the classical constant beam width pruning achieves with the same WER increase of 1 % a time factor of 0.19 (acceleration by 5 times). Further details of the evaluation results show that remarkable improvement could be achieved in decoding speed of the ASR system as presented in Table 4.1.

4.3 Comparison with the Adaptive Technique

In this section two dynamic pruning algorithms are compared, the *confidence-guided dynamic* (CGD) pruning and the *adaptive control dynamic* (ACD) pruning method. Both algorithms set the pruning threshold for the Viterbi beam search process dynamically for each time frame depending on search space properties. Earlier in this chapter the CGD pruning method was presented which uses confidence measurement to minimize the computation time effort of the Viterbi search process by reducing the search space to an acceptable level (i.e. the number of active hypotheses). The decision of the appropriate threshold at each time frame is based on the utilization of confidence measurement. The other dynamic pruning approach ACD was presented

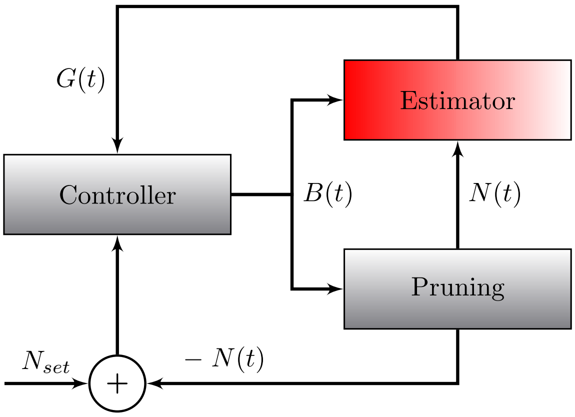

Figure 4.7: Schematic view of the adaptive control dynamic (ACD) pruning approach.

in Van Hamme & Van Aelten (1996) and Zhang & Du (2004). This pruning method uses adaptive control techniques to steer the pruning threshold dynamically. As will be shown in this section, both dynamic pruning techniques are applicable in reducing the time consumption of the recognizer whereas the novel confidence-guided pruning approach clearly outperforms the adaptive control technique (see also in Fabian & Ruske, 2006).

4.3.1 Adaptive Control Dynamic Pruning

One possibility to steer the dynamic beam width frame by frame is to take the advantages of adaptive control algorithms into account (see Van Hamme & Van Aelten, 1996). ACD pruning is a technique which changes the pruning threshold B_{set} of the probability based pruning for the Viterbi search process at runtime to compensate the variations in the search environment and to achieve the preset threshold of maximum number of hypotheses N_{set}.

Figure 4.7 shows the topology of the ACD pruning approach. This method uses a *feedback-control mechanism* that contains adjustable coefficients. The ACD pruning system consists of an *inner loop* and an *outer loop*. The inner loop contains an ordinary feedback loop and the plant (the controlled system). In Figure 4.7 these are the feedback *controller* and the *pruning* process. The parameters of the controller are adjusted by the outer feedback loop, the recursive parameter *estimator*, which is able to tune the parameters automatically to achieve the desired behavior of the system (see Astrom & Wittenmark, 1995). In Figure 4.7 the input parameter of the pruning is the beam width $B(t)$ and its output is the number of active hypotheses $N(t)$ for each time frame. N_{set} is the expected number of the hypotheses, a preset value and the goal of the described adaptive mechanism is to drive $N(t)$ close to the preset value N_{set}. For this reason the beam width of the pruning $B(t)$ is varied dynamically for each time frame.

The pruning process is a non-linear time-variant dynamic system but for simplicity it can be modeled by a 0th order linear system with slowly varying gain using the following simple differential equation (see Zhang & Du, 2004):

$$N(t) = G(t)B(t), \tag{4.22}$$

where $G(t)$ is the time-variant gain reflecting the relation between the beam width $B(t)$ and the number of active hypotheses $N(t)$. The controller is an integrator which can be described with the following equation:

$$B(t+1) = B(t) + \alpha(N_{set} - N(t))/G(t), \tag{4.23}$$

where α is the parameter of the controller which can adjust the response speed of the feedback loop. The time-variant gain $G(t)$ in Equation 4.23 can be estimated based on the past L observations of the pruning threshold $B(t)$ using *least squares estimation* with the following formula:

$$G(t) = \frac{\sum_{i=1}^{L} N(t-i)B(t-i)}{\sum_{i=1}^{L} B^2(t-i)}. \tag{4.24}$$

For the dynamic pruning approach, reasonable parameter values are $L = 5$ and $\alpha = 0.2$ as proposed in Van Hamme & Van Aelten (1996). The computation steps of the pruning process based on this adaptive controller are as follows:

Step 1: Estimation of the gain $G(t)$ of the pruning process, Equation 4.24

Step 2: Computation of the pruning threshold $B(t)$ with Equation 4.23

Since Equation 4.24 and Equation 4.23 must be calculated only once per frame, their computation costs should not compromise pruning efficiency. To catch side effects of the controller, especially at the beginning of an utterance, the computed pruning threshold should be limited by maximum and minimum values (see Van Hamme & Van Aelten, 1996).

4.3.2 CGD versus ACD Pruning

The main advantage of both CGD and ACD pruning techniques is the framewise computation of the pruning threshold for the search process. That way both of them are able to take time-variant characteristics of the search process into account. This is clearly shown in Figure 4.8 on page 75 with an example sentence of the Verbmobil evaluation set. The diagram shows the dynamic pruning threshold of ACD pruning depending on the frames of the example utterance. In this diagram the horizontal line at $y = 0$ (i.e. the x-axis) represents the best hypothesis scores for each frame. The pruning threshold is plotted relative to that as y-value frame by frame. The

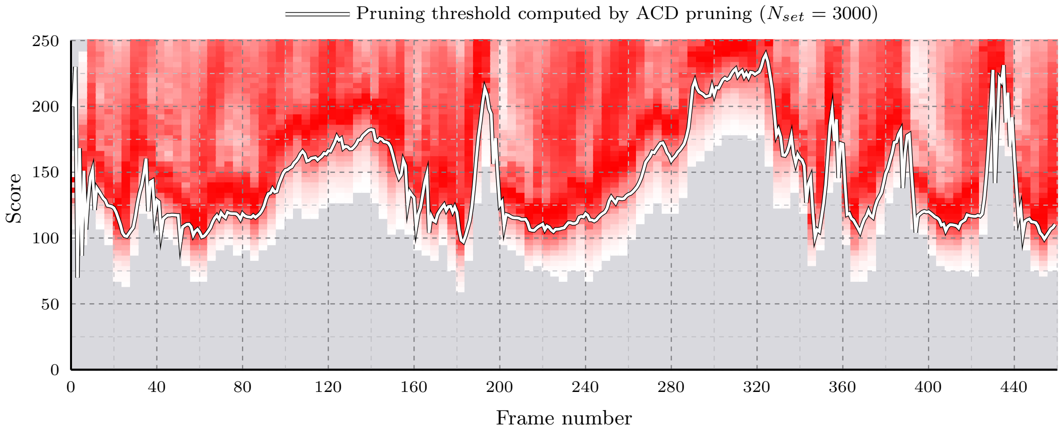

Figure 4.8: Histogram of active hypotheses and an example for dynamic pruning threshold during the appointment negotiation utterance 'Ja genau, lassen wir gleich die letzte Woche im März, prima!' (English: 'That's right, let's keep the last week in March, great!').

pruning threshold curve for CGD pruning almost resembles the plotted curve of ACD pruning so it is omitted for the sake of clarity.

There is one main difference between CGD and ACD pruning: at the beginning of the $B(t)$ curve of ACD pruning a kind of transient oscillation can be observed, as clearly presented by Figure 4.8. This is caused by the integrator due to the insufficient number of observation values for the computation of the plant's gain in Equation 4.24 (page 74). At this point CGD pruning holds a clear advantage because its computation is based on confidence measurement and therefore it does not exhibit this negative effect at the beginning of the utterance.

In addition to the pruning threshold curve, the histogram of the number of the active hypotheses is also plotted in Figure 4.8 as the color-coded z-axis in the background. This histogram was computed over equidistant score intervals of 4 in the range of [0-250] from the x-axis. The color transition from white to red color (in gray scale, from white to black) illustrates the number of active hypotheses in the range from 1 to ∞ for each time frame and score interval. Gray color beneath the color gradient represents no active hypothesis for the specified intervals.

In the histogram plot of Figure 4.8 we can see that the number of active hypotheses generally increases with increasing distance to the best hypothesis (x-axis). When a preset constant pruning threshold was used (i.e. horizontal line parallel to the x-axis e.g. by score = 250) there were many hypotheses of poor quality which could not be pruned because they fell below the constant threshold, the maximum score distance allowed from the best hypothesis score. In contrast to the constant threshold, the time dependent dynamic pruning thresholds of CGD or ACD pruning methods is clearly able to cut off more hypotheses. This is possible because the dynamic threshold demarcates the edge of the histogram transition to the increasing

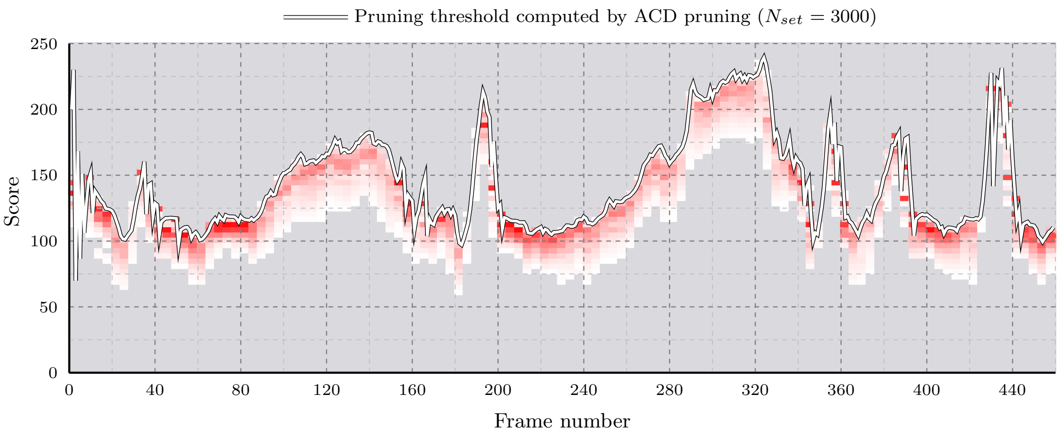

Figure 4.9: Histogram of active hypotheses after cut off by dynamic pruning threshold for the same example utterance as Figure 4.8 on page 75 using ACD pruning technique.

number of active hypotheses framewise at different score distances from the x-axis. As a result the number of active hypotheses can be reduced dramatically as shown in Figure 4.9 in comparison with Figure 4.8, which means the ASR can be speeded up and on the other hand, enormous memory savings can be realized.

4.3.3 Comparison Results

The goal of the experiments presented in this section was to compare the CGD and ACD pruning techniques. For this purpose several tests were performed on the Verbmobil '96 test data (see Section 3.7.2 on page 53) using different pruning techniques. The results are presented in Figure 4.10 on page 77 and in Table 4.2 on page 78. The investigation was focused on the comparison between the CGD pruning technique and the adaptive control approach, however, Figure 4.10 allows comparison of results of the following pruning methods (similarly to Section 4.2):

- *probability-based* (PB) beam width pruning is the classical approach setting the beam width of the Viterbi search based on best hypothesis score and a specific threshold
- *probability based rank* (PBR) pruning limits the maximum number of active hypotheses, such as histogram rank pruning, in combination width PB pruning
- *confidence-guided dynamic* (CGD) pruning computed based on Equation 4.10
- *adaptive control dynamic* (ACD) pruning applied as described in Section 4.3.1

As far as the results of the PB pruning technique are concerned, the curve in Figure 4.10 was determined by computing the WER for the evaluation corpus using different pruning values B_{set} in a range of [80-250]. The greater pruning threshold

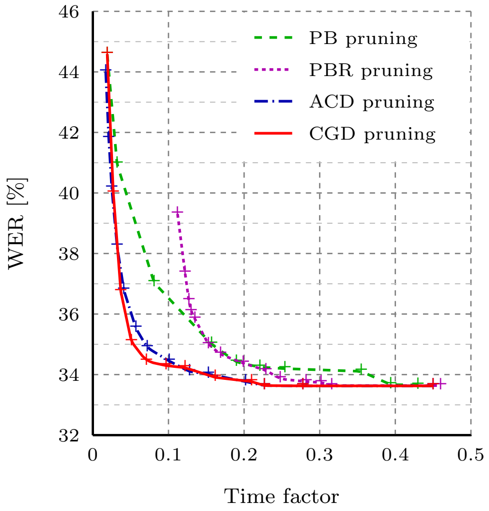

Figure 4.10: Word error rates (WERs) of different pruning techniques, classical, adaptive and confidence-guided dynamic pruning, depending on time factor[1] (see page 71): probability-based (PB) beam width, probability-based rank (PBR), adaptive control dynamic (ACD), and confidence-guided dynamic (CGD) pruning.

value has a lower WER but a higher time factor. The combination of probability-based and rank pruning PBR was evaluated by keeping B_{set} at 210 and varying N_{set} in the range of [500-9000]. Regarding the dynamic approaches, the resulting WER curve of the CGD pruning method was found using $T_{upp} = 110$ and different T_{low} in a range of [20-70]. In order to obtain results for the ACD pruning method, N_{set} was varied in the range of [300-10000].

Figure 4.10 shows that both dynamic pruning techniques outperform the static methods significantly. The time factor of the ASR could be decreased to 0.23 without increasing WER by using CGD or ACD pruning. Furthermore if an increase in WER of less than 1 % is acceptable, ACD pruning achieves a time factor of 0.1 which corresponds to the acceleration of the ASR by 10 times (reciprocal time factor). Or, compared with the best PB pruning result, ACD pruning makes the ASR 1.9 times faster. The best result was achieved by the CGD pruning approach, namely a time factor of 0.07 which corresponds to ASR acceleration of about 14 times, or 2.7 times compared with the best PB pruning result. Further details of the evaluation tests are shown in Table 4.2 on page 78.

One question remained: how is it possible to achieve better results using ACD pruning technique with the preset value of $N_{set} = 3000$ than with the classical rank-based approach with $N_{set} = 2000$? The explanation is that ACD pruning controls the beam width of the search process to avoid exceeding the maximum number of active hypotheses for each time frame. In contrast, the rank-based pruning needs two passes: first the Viterbi search step is performed and only afterward is the number

Pruning method; parameters	**WER [%]**	**Time factor**
PB; $B_{set} = 250$	33.63	0.43
PB; $B_{set} = 150$	34.40	0.19
PBR; $B_{set} = 210, N_{set} = 9000$	33.63	0.32
PBR; $B_{set} = 210, N_{set} = 2000$	34.37	0.20
ACD; $N_{set} = 8000$	33.63	0.23
ACD; $N_{set} = 3000$	34.44	0.10
CGD; $T_{upp} = 110, T_{low} = 40$	33.63	0.23
CGD; $T_{upp} = 110, T_{low} = 70$	34.43	0.07

Table 4.2: Word error rates (WERs) and the corresponding time factors[1] (see page 71) of ACD and CGD pruning techniques in comparison with different classical pruning methods.

of the active hypotheses reduced to the preset value for the next search step. As a result ACD pruning indeed achieves an average N_{set} of 3000, but the ASR using the classical rank-based pruning approach often has to handle 3 or 4 times more hypotheses, which leads to increased computation time effort.

4.4 Summary

The comparison of the two dynamic pruning methods CGD and ACD pruning has shown that both of them are applicable to reduce the computation time of the speech recognizer. CGD as well as ACD pruning approaches perform significantly better than classical pruning techniques. As a result, a significant improvement in decoding speed of the ASR system could be achieved. The best results of the confidence-guided dynamic pruning approach outperforms not only classical pruning techniques but also ACD pruning.

Chapter 5

Human-Machine Dialog Control

Previous chapters of this work deal with ASR core technology and different confidence measurement techniques which are implemented at the core level of speech recognition, providing an assessment of the confidence of recognition results. This chapter's focus is to show how confidence measures can be utilized at the dialog level in speech-based human-machine interaction. The realization of appropriate *dialog control strategies* is quite a current topic within many industrial branches across the field of voice-enabled applications. On the other hand, dialog control approaches are well connected to the field of confidence measurement; therefore, this work could not be considered complete without discussing current strategies and implementation techniques.

The course of interaction in speech-based communication between human and machine needs to be steered — sometimes in a more restricted manner, sometimes less formal, depending on the capabilities of the specific voice application and on the experiences of the target user group. In this context, steering means making decisions regarding the instantaneous dialog flow, i.e. processing or skipping dialog states or entire branches, based on information collected over the course of the dialog so far. The quality of steering, on the other hand, depends heavily on the correctness and reliability of ASR results. In most cases confidence scores serve as the basis for decisionmaking but the questions are always: *1)* how reliable confidence scores really are in certain situations and *2)* which additional knowledge sources could be used in order to improve reliability of decisions at dialog level.

To start with, let us take a closer look at two main levels of the decision hierarchy of speech-based applications which merit consideration in the context of this chapter:

- At the *state level* of dialogs, decisions regarding acceptance or rejection of particular recognition results are made based on a specific confidence threshold and on confidence scores of hypotheses provided directly by the speech recognizer. In this case local decisions are needed in order to control reprompt strategies, always with the goal of gathering from the user exactly the information for which the dialog state was designed.

- At the *application level*, the entire dialog flow and its history must be considered. The main intelligence of the application is implemented at this level, providing global decisions from the application's point of view. For example, in case of repeatedly poor ASR results or too many questions at the state level, the best global decision may be to surrender the conversation to a human operator rather than frustrating users by causing misunderstandings.

In this chapter the primary elements of user interaction are outlined together with their underlying dialog architectures. Possible pitfalls to speech-based interaction are pointed out and classified. The major goal of this chapter is to describe the use of confidence measurement in the field of dialog control. Instead of focusing on particular applications, the chapter deals with global concepts and problem definitions which emerged from the author's research activities and industrial experiences in human-machine speech communication over the past decade.

5.1 Speech-Based Interaction

The section provides a general discussion of the fundamentals of human-machine interaction in voice enabled applications. Figure 5.1 on page 81 shows a schematic overview of dialog control architecture exemplarily with its main modules and their communication channels. Here, each module is described briefly according to its role in supporting interactive speech-based communication.

Dialog Manager (DM)
DM is the main controller and processing unit. It synchronizes all resources of the underlying interactive voice response (IVR) framework and connects several knowledge source, e.g. databases. It makes real-time decisions on the basis of available outcomes of the dialog history, e.g. content of ASR results and their confidence scores.

Application Library (AL)
AL consists of different dialog flow descriptions. In each application description the main intelligence of the dialog is implemented with the aid of a specific dialog description language, e.g. standard VoiceXML. To interact with the user, the dialog manager executes the application description according to user inputs.

Automatic Speech Recognition (ASR)
The ASR module can generate simply-recognized phrases but can also contain natural language understanding (NLU) units in order to produce semantic concepts

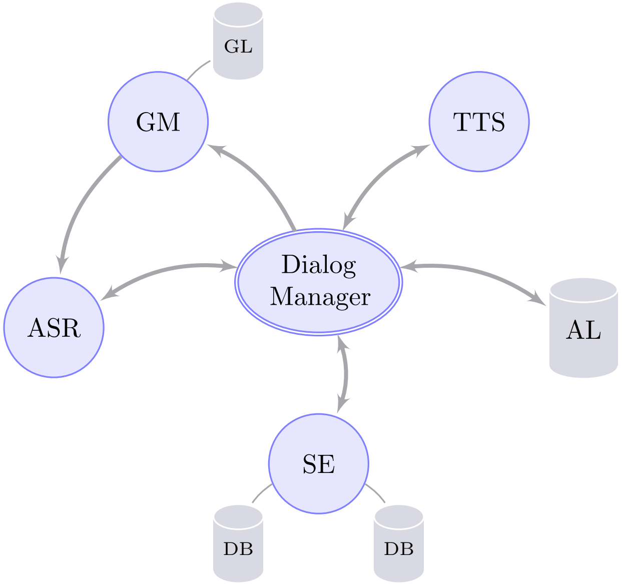

Figure 5.1: Schematic view of an example dialog management architecture with its main modules and their interaction connections. These main modules are Dialog Manager, Text-To-Speech (TTS) module, Application Library (AL), Search Engine (SE) connected to several databases (DBs), Automatic Speech Recognition (ASR) module, Grammar Manager (GM) with Grammar Library (GL).

that interpret the essence of the user's utterance. Results at concept level are then used by the dialog manager for making decisions independently of the exact wording provided by the user. NLU allows robust application design since mapping between concept hypotheses and their possible wording variants is made by appropriate grammars used for ASR.

Grammar Manager (GM)
The GM supplies the dialog with grammars (language models) held in the grammar library (GL) which are suited to the application-specific speech recognition tasks. Regarding their generation phase, grammars can be static or dynamic. A *static grammar* is prepared entirely during the set up phase of the application because its content is independent of the exact dialog flow. *Dynamic grammars*, on the other hand, cannot be generated until all information needed for their composition is provided by the application; the content of a dynamic grammar depends partly or entirely on information gathered during the dialog. Example grammars which are defined by standard formats (see W3C-Grammar, 2004) are shown in Appendix B on page 105.

Search Engine (SE)
The search engine is responsible for complex search processes in voice search ap-

plications, especially in the field of automated management of bank accounts, or directory assistance systems dealing with huge amounts of information stored in databases (DBs). Such dialogs need reliable data management mechanisms in order to make necessary information available to the dialog manager without any delays caused by the search process.

Text-To-Speech (TTS)
The TTS module is responsible for transforming textual information produced by the application into a speech signal which can be provided to the user as part of the human-machine interaction. Speech generated by the application is often a mixture of prerecorded audio files containing human voices and audio data provided by the TTS module on a real-time basis.

The bidirectional audio link between user and application, i.e. the voice channel, is omitted in Figure 5.1 in order to maintain clarity by focusing on those modules which are relevant to this work. The voice channel, e.g. telephone line, is responsible for the physical transfer of all acoustic information needed for communication between user and IVR platform. Nowadays, there is an increasing tendency toward replacement of classical telephone lines, analog or digital, with Internet-based technology known as voice over IP (VoIP).

Automatic speech detection is also an important module whose reliability greatly influences recognition accuracy as well as CM quality. Speech detection is usually performed by the voice activity detector (VAD), which is part of the ASR system but often independent from the speech recognition engine itself. Speech decoding is performed only on those audio data which the VAD considers to be speech. Under certain circumstances the VAD may fail and the decoder runs mistakenly on non-speech audio data. This occurs most typically in noisy environments or on low quality transmission channels if the user does not say anything but the level of spurious sounds triggers the VAD. As a consequence, the audio data provided for speech decoding by mistake can be interpreted as speech and result in production of false hypotheses. In such cases the hypothesis confidence score should allow detection of speech recognition failure afterwards in order to provide efficient error recovery in dialog management.

The dialog management framework described above based on Figure 5.1 allows speech-based human-machine interaction by interworking of all needed resources. The dialog manager reserves all necessary resources, e.g. ASR or TTS, for a specific application, which is selected from the application library, before communication between human and machine can start. Generally the application starts with a welcome prompt in order indicate its presence to the user, followed by the initial question. Throughout the dialog, the user's answers are recognized by the ASR module and necessary data sources are used to reach the goal of the application, e.g. to manage the user's bank account successfully. The following sections describe additional details of this communication process together with the efficient use of confidence measurement techniques.

5.2 "Sorry, I Still Didn't Catch That"

There are several potential failure sources which may decrease the quality of user interaction in such a highly complex framework as that shown in Figure 5.1. The classification of failure causes provided in this section focuses on problems which may occur during user interaction and which lead to deviation from the optimal dialog flow. Problems are elucidated from the perspective of inadequate interplay between application logic and user behavior.

In order to point out problems in speech-based applications in practice, possible failure sources are examined on the basis of sample dialogs from the field of telephone-based voice search applications, i.e. a user query for a specific telephone number. In the first example, the dialog system interprets user utterances through natural language understanding (NLU) and reacts appropriately as shown in a short dialog example below.

Dialog A:

System: This is an automated inquiry service. How can I help you?
User: I need the telephone number for Technische Universität München.
System: Are you looking for Technische Universität München in München?
User: Yes.
System: The number is . . . Thanks for calling, good bye.

This is an ideal case of natural user interaction and it may work only for specific tasks, e.g. if the NLU grammar contains only a limited number of variants for certain semantic units, but it probably does not work for huge amounts of data handled by voice-enabled systems. In practice, the dialog flow for the same query as above is rather feasible as the next example shows.

Dialog B:

System: This is an automated inquiry service. Please, say the city.
User: München.
System: I've got München, is that the right city?
User: Yes.
System: Which listing?
User: Technische Universität München.

System:	Are you looking for Technische Universität München in München?
User:	Yes.
System:	The number is . . . Thanks for calling, good bye.

A directory assistance system which handles users' inquiries in similar fashion to Dialog B above is presented in detail in Yu *et al.* (2007). As stated in Yu *et al.* (2007) a relatively large proportion of dialog failures is caused by the underlying ASR system. Requirements for ASR systems depend highly on dialog strategies used in a certain application; here, ASR accuracy is influenced especially strongly by the degree of naturalness allowed for user utterances. In the ideal case, from the user's point of view, the user is allowed to communicate with the application by means of a dialog just as humans naturally interact, as Dialog A shows above exemplarily.

The decision about acceptance or rejection of a specific ASR result is made based on its confidence score and on a certain confidence threshold. For practical reasons, industrial automated speech recognition systems provide confidence scores in a fixed value range, e.g. $[0, 100]$ where the score 100 stands for highest hypothesis confidence. Depending on the hypothesis score, speech applications decide whether ASR results can be used as valid user input or not by means of predefined thresholds. In Dialog A above, which allows NLU, the semantic confidence scores of concepts in the user utterance *"I need the telephone number for Technische Universität München."* must be very high since no further confirmation step is performed. Obviously, the application considers the user's utterance as recognized with certainty.

In contrast, Dialog B shows a confirmation step of the user input *"München."*. Confirmations in general are intended to increase certainty in processing user inputs. In a *dual-threshold strategy*, for example, three decision results are possible depending on two different confidence thresholds: the *confirmation threshold* T_c (the upper confidence threshold) and the *rejection threshold* T_r (the lower threshold), where $T_c > T_r$. Based on these confidence thresholds, possible decision results can be:

- if the hypothesis confidence score is above T_c, the user input is accepted without any further confirmation since recognition result is considered to be confident

- if the confidence score falls below the confirmation threshold T_c but above the rejection threshold T_r, the decision is that further confirmation of user input is needed, e.g. reprompting; and finally

- if the hypothesis score falls below T_r, the user input is rejected, i.e. not understood

At first glance, when the two example call flows Dialog A and Dialog B are compared, it is readily apparent that the second dialog takes much longer than the first NLU dialog. This is because in the second case the application asks the user separately for all attributes needed for DB search. In spite of the obvious loss of naturalness of the underlying dialog design, the second dialog is a more realistic implementation due to existing limitations in current ASR technology. The explanation is that performance of CM depends significantly on vocabulary size or grammar complexity. In the example above, all German cities need to be recognized and so the vocabulary must contain several tens of thousands of entries which are often quite similar. The successful management of this recognition task and the generation of reliable confidence score is a demanding requirement for state-of-the-art ASR technology. Dialog A on page 83, however, combines this requirement with recognition of German cities embedded in arbitrary, natural formulations by the user. In this case confidence score must be computed at a higher level, i.e. for the semantic concept of the city. The resulting CM quality of such complex recognition tasks, however, often leaves a lot to be desired. It is also a more realistic scenario that such a conversation contains confirmation steps in case the application is not sure of the exact content of the user utterance.

The accuracy of confidence-based decisions depends not only on the use of appropriate confidence thresholds but also on the reliability of the hypothesis confidence score itself. The quality of confidence scores, on the other hand, is highly affected by the content of ASR grammars which are used for the recognition tasks. Especially the degree of coverage of possible user inputs is an important requirement for the construction of adequate grammars. The following dialog example shows failures in human-machine user interaction which can be avoided by the use of appropriate grammars:

Dialog C:

System: This is an automated inquiry service. Please say the city.
User: München.
System: Did you say Münster?
User: No, München.
System: Sorry, I didn't catch that. Please say yes or no.
User: No.
System: Please, say the city again.
User: München.
System: I've got München, is that the right city?
User: Correct.
System: Sorry, I didn't catch that. Is München the right city?
User: Yeah.

System: Sorry, I still didn't catch that.
It seems that we have some technical difficulties.
Please, try again later. Good bye.

Failures in the example interaction above are due to missing formulation variants of user inputs in the grammar such as for the speech input *"No, München."*, where user voluntarily provides the right city again but the system is not able to handle this complex answer by design. Similarly, the grammar may not contain the words *correct* or *yeah* if the system expects only *yes/no* as user answers to the confirmation question. Nevertheless, from the application's point of view, the dialog flow is handled correctly since the application detects bad recognition results of unexpected phrases correctly on the basis of appropriate confidence thresholds. It avoids continued work on misunderstood recognition results. Grammars need to consider out-of-vocabulary (OOV) and out-of-grammar (OOG) rates, pronunciation variants, misspellings, hesitations by design.

5.3 Importance of Confidence Measurement

By taking all aspects and levels of human-machine speech interaction into account the quality of confidence measurement directly involves the following items:

- dialog design to meet application target
- user acceptance
- error recovery strategies
- grammars used for recognition tasks, OOV, OOG rates

Speech dialogs should always be designed to meet the target of the application using all necessary resources of interaction with the user, especially appropriate CM for dialog flow control. Hypothesis confidence score generated in the ASR module depends heavily on grammar content, i.e. on grammar size and on the complexity of wordings users are allowed to utter. Especially in voice search applications, grammar size is often the limiting factor in reaching high quality and reliability. Therefore, effective reduction in grammar size is always a topical issue in practice. Error recovery strategies, such as asking the user again for specific information that is still missing, providing help information and reprompting, or passing the conversation along to a human operator in a timely manner, are triggered by confidence thresholds. The more reliably speech applications serve users' needs, the more they are accepted by the users. Unnecessary confirmation steps as well as misunderstandings during dialog are leading cause of user frustration.

Several papers have been published on the evaluation of different dialog management strategies. Möller *et al.* (2007), for example, distinguish between five failure categories depending on the level in the hierarchy of information processing, namely *a)* goal-level error, *b)* task-level error, *c)* command-level error, *d)* concept-level error and *e)* recognition-level error. All these categories describe mismatching of user expectations and system capabilities at different levels of data processing caused by insufficient application design.

Confidence thresholds for decisionmaking must always be determined empirically in expensive evaluation phases using appropriate test sets. Colibro *et al.* (2005) propose a method which is intended to reduce dependency of confidence thresholds on different languages, grammars and vocabularies. If application developers are allowed to use universally valid thresholds for different applications, they can save cost intensive evaluation phases by expecting comparable accuracy in rejection rates. The CM approach presented in Colibro *et al.* (2005), called normalized differential confidence measure (C_{NDC}), is based on posterior probability computation on acoustic state level. Bohus & Rudnicky (2005) present a data-driven approach for determining the relative costs of errors, e.g. misunderstandings and false rejections, and use these costs to optimize rejection thresholds for spoken dialog systems. In other words, the method presented in Bohus & Rudnicky (2005) determine the optimal rejection threshold for a given trade-off between false rejection and false acceptance automatically on evaluation data collected from live applications.

Several CM strategies have also been devised: Cavedon *et al.* (2005), for example, combine confidence scores with contextual features of multiple sources to rate possible dialog handling strategies. Confidence scores are generated during decoding by ASR or natural language parser as semantic confidence. San-Segundo *et al.* (2001) show that predictor features for CM derived from language models perform significantly better in spoken dialog systems than decoder-based acoustic features. Li & Huerta (2007) propose a technique which allows confidence prediction to improve early decisions for adequate dialog strategies based on knowledge of context-dependent confidences associated with previous turns in a dialog. Li & Huerta (2007) compare several methods along this context, for example, direct linear prediction, histogram based linear prediction or maximum entropy model based classification. The first method predicts confidence scores for a specific context as the linear combination of the past observed confidence scores and discrete events, computed as follows:

$$C_t = \sum_i \alpha_{t,i,x} X_i + \sum_j \alpha_{t,i,y} Y_j + \beta_t, \tag{5.1}$$

where X denotes the sequence of discrete events, e.g. rejection or no input; Y stands for a sequence of confidence scores and α, and β are prediction coefficients. The use of Equation 5.1 allows continuous prediction of confidence depending on time t as the dialog evolves in real time. Using confidence prediction it is possible to monitor system performance automatically and intervene in time if predicted outputs do not meet expectations.

As stated earlier in Section 3.6.3 on page 51, unsupervised adaptation is available with the aid of appropriate CMs in order to improve quality in modeling speech characteristics for specific tasks. This feature is already available in practice and there are speech applications that utilize unsupervised adaptation. However, this feature holds a major risk for real applications because if automatic (unsupervised) adaptation is activated, considerable degradation in ASR performance cannot be ruled out and the application may become hampered by serious usability problems.

5.3.1 Approaches to Dialog Flow Control

The degree of confidence of applications toward ASR results also depends on voice user interface (VUI) design, which is responsible for defining roles of interaction and hence which dialog situations are expected and allowed. When confronted with unexpected user behavior, the application cannot handle it adequately. The repertoire of VUI design is growing continuously with the development of underlying technical possibilities and is slowly converging to support human-like interaction. As already stated in the introduction to this work in Chapter 1, marginal conditions of the application's area of operation are to be clarified during the dialog design phase. The entire scope of the application is to be covered with an appropriate dialog structure in order to provide access to all necessary user tasks via the speech interface.

The degree of naturalness, for example, is a very important design aspect because user acceptance and ergonomics demand as much naturalness as possible in order to approach dialogs that approximate human interaction. On the other hand, however, current ASR technology does not allow the use of natural language understanding in every situation of human-machine interaction. Expectations for naturalness are also highly dependent on the application's user group and on user age and experience. Inexperienced users require guidance through the application, with only a restricted set of input possibilities allowed in order to avoid confusion. *Mixed initiative* dialog forms are able to fulfill expectations of both experienced and inexperienced users by providing more flexibility in permissible user inputs. Mixed initiative dialogs expect both complex and simple user inputs. In the case of simple user utterances, the dialog asks for further details which are still missing in the user's input but are necessary for the application to process the dialog. Experienced users, on the other hand, are allowed to phrase all necessary information in one complex utterance.

Generally speaking, humans are not necessarily keen on communicating with machines; therefore, in order to avoid frustrating user experiences and to maintain the focus on the application target, the dialogs for specific tasks should never be prolonged. Confirmation steps are also needed for robustness and error handling, but it must always be carefully considered which type of confirmation is best suited to the specific task. *Implicit confirmation*, for example, allows a more natural dialog flow but is not as robust as its counterpart, *explicit confirmation*, which only allows much more limited possibilities in users' answers. An example of explicit confirmation is shown in Dialog B on page 83, where the system explicitly requests the user to confirm the city with yes or no. If the user confirms with yes, the dialog has a solid basis for further data processing since it can be quite certain in confidence scores produced for a very small yes/no vocabulary. Implicit confirmation, on the other hand, echoes back the information recognized in the previous ASR task without requesting any confirmation from the user. Implicit confirmation would be more challenging as shown in the following example subdialogs.

Dialog D:

System: This is an automated inquiry service. Please, say the city.
User: München.
System: I've got München, which listing?
User: Technische Universität München.

The next subdialog shows an other example of implicit confirmation for the case if the system failed to recognize the right city.

Dialog E:

System: This is an automated inquiry service. Please, say the city.
User: München.
System: I've got Münster, which listing?
User: No, I said München.

Although implicit confirmation does not request the user to confirm explicitly, it must allow negative user answers in case of failures. The examples above, Dialog D and Dialog E, show one positive and one negative user answer implied by good and bad system results. Good VUI design can handle both cases with the use of appropriate grammars. In comparison with explicit confirmation, however, it is clear that ASR tasks on phrases which are strictly limited to yes or no produce much more reliable confidence scores than complex ASR tasks performed in case of implicit confirmation.

5.3.2 The Use of Additional Knowledge Sources

It is a well known fact that hypothesis accuracy and CM quality are highly influenced by grammar size used for recognition tasks. The smaller the grammar, the better the quality of ASR results that can usually be achieved. Therefore dialog design, especially in voice search applications, focuses on the use of additional knowledge sources in order to reduce grammar size whenever possible. In voice search applications, e.g. directory assistance, it is possible to combine confidence scores, which are computed in the ASR module, with additional knowledge sources such as the result of the database search.

Depending on the way information is extracted from the database, it can be used to reduce grammar complexity. To describe this technique, an example voice search application is shown in Figure 5.2 on page 91. The dialog shown there asks for search attributes separately, as *"Please, say the city."* or *"Which listing?"*. As shown in Figure 5.2 after the recognition of the first attribute, the city is used to narrow down the number of possible listings the user may ask for which can only be found in that unique city. Otherwise, similar to Dialog A on page 83, the ASR must be able to recognize all possible listings in the entire country, i.e. several million. As far as the quality of grammars is concerned in the field of voice search applications, it is a matter of great importance to preprocess database content prior to using them for grammars. This is because the textual representations are not optimized for speech recognition but contain, for example, spelling errors or context dependent abbreviations (see Yu *et al.*, 2007). Correctness at grammar level is essential to good results in speech recognition and confidence score computation.

A common technique in the field of voice search applications is to incorporate search results directly into speech recognition of the user utterance formerly recorded in previous recognition steps. This is called *rerecognition* and is performed in the background by IVR systems without involving the user. State-of-the-art IVR technology is able to perform multiple recognition and search tasks without any noticeable delays in interaction with the user. The main advantage of rerecognition is to reduce the size of ASR grammars. This is possible by incorporating database content through dynamic grammar generation for a specific recognition task while using ASR results hypothesized in previous recognition steps. Figure 5.2 on page 91 shows that after listings are found in the database, new grammars are generated and an additional rerecognition step is performed using that grammar. Changes in grammar content can be more variants in wording for specific listings or detecting OOV with much higher reliability which was not possible in the previous recognition step because of much higher grammar complexity. Through the technique of rerecognition, the size of the grammar can be reduced efficiently and the consequence is significant improvement in recognition accuracy and in reliability of hypothesis confidence score.

Parallel data processing is another field of combining multiple knowledge sources in order to improve CM quality. A technique which makes use of parallel computation of confidence scores is presented in Lopez & Mateo (2005). Each recognition task is responsible for a particular topic in the dialog and the confidence scores generated by those recognition tasks are derived from low-level knowledge sources such

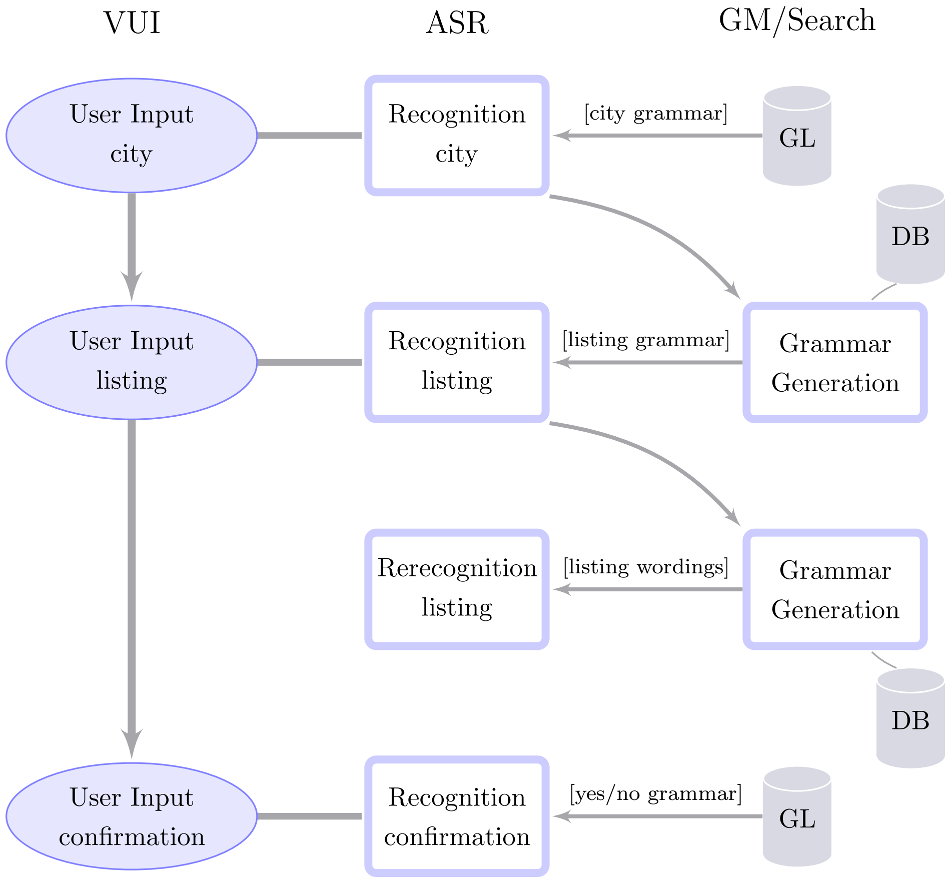

Figure 5.2: Block diagram of an example voice search application with different levels of data processing: voice user interface (VUI), automatic speech recognition (ASR), grammar management (GM) and search level. VUI is the level of user interaction and speech recognition of user utterances is carried out in the ASR level. Appropriate static grammars for ASR tasks, e.g. city grammar or yes/no grammar, are provided by the grammar library (GL). Dynamic grammars, on the other hand, such as listing grammar or listing wordings, are generated in real time based on ASR results hypothesized in previous recognition steps and database (DB) search results. Rerecognition of listings at the ASR level is performed on recorded user input using alternative wordings provided in a (usually) small dynamic grammar (of listing wordings).

as acoustic and linguistic information. Confidence scores of each recognition task are then merged by a neural network classifier, i.e. multi-layer perceptron, to determine a global decision of rejection or acceptance for the dialog manager. Similarly, Vanhoucke (2005) presents an approach of confidence-based decision using multiple speech recognizers in a multi-pass framework where a combined confidence score is computed by incorporating results of the second pass of the decoding together with a set of features derived from the first pass. The final hypothesis confidence score is determined by the combined output of two recognizers based on a set of penalties e.g. on the semantic agreement between the two passes.

Several open issues need to be solved in order to enhance the ergonomics and quality of voice-enabled interactive dialogs. Future speech applications should react to users more appropriately by detecting communication failure as soon as it occurs and by using more sophisticated error recovery strategies. Machines should be able to manage dialogs by incorporating several nuances of communication with users, e.g. the user's age or current mood. Those nuances should determine the *persona* used by the application, which should adopt patterns of behavior appropriate to the current user requirements. Users, on the other hand, also need to adapt their expectations of machines' real capabilities. Excessive user expectations invariable lead to high failure rates and a frustrating user experience.

Chapter 6

Conclusion and Outlook

The main goal of the dissertation was to demonstrate how important reliable confidence measurement (CM) techniques are to modern speech-based human-machine communication in general. Beginning with basic computational aspects and classification details at the core level of automatic speech recognition (ASR), several utilization techniques of CM were outlined in order to enhance ASR computation approaches and also to apply CM to higher level dialog control strategies. The thesis focused on the description of CM techniques in the field of HMM-based automatic speech recognition. Widely-implemented and well-proven CM approaches were presented and it was shown how they can be used successfully for speech applications in practice. The dissertation presented a survey of the abundant literature on different CM techniques which were developed through extensive research activities during recent decades.

This work classified known CM techniques from different points of view, i.e. based on the speech units to which CMs are applied, on the underlying computation methods and with respect to fields of utilization in speech applications. Underlying computation methods of confidence predictor features were categorized based on acoustic and/or language model information collected during decoding. The usage of confidence predictor features was also shown by merging a certain combination of predictor features into a single probabilistic confidence score via neural networks in order to be able to make a unique confidence decision. Several CM utilization techniques were described in the fields of hypothesis rescoring and pruning during Viterbi search, rejection techniques, adaptation methods and dialog management strategies. As a conclusion it can be stated that CM performance of merged predictor features can be improved only when combined features are statistically independent. Overlap between features is often quite large; therefore, the resulting CM quality is predominantly determined by the performance of the best feature.

Due to its outstanding performance as confidence measure, a more comprehensive description of the computation details of the hypothesis posterior probability was provided. Methods were described which are performed during decoding and as a post-processing step on word graphs. To sum up evaluation results for this area, it seems that as far as the computation of CM on the acoustic level is concerned, best performance is already achieved using posterior probability directly as a measure of hypothesis confidence. Improved quality can be achieved by taking time alignment information of the word graph into account. Significant further improvement in CM can be achieved only through use of additional knowledge sources, e.g. semantic information at a higher level of language understanding.

The major goal of the confidence-guided (CG) pruning technique, which was developed in the scope of this work, is the utilization of confidence measurement during the search process. The normalized log likelihood (NLL) scores of active hypotheses were found to be a good measure of confidence which is computed at each time frame in real time to serve as the basis for pruning decision. The NLL score of the best hypothesis is used together with an appropriate constant threshold to define the beam width of the Viterbi search in the NLL space. In this process, the main challenge is efficient estimation of observation probability, the normalization term in the NLL formula. This is because HMM-based speech recognition environments usually do not have any applicable models for direct estimation of the observation probability. To solve this problem, a catch-all model was generated from the entire set of acoustic models of reduced size using an iterative bottom-up clustering process. It was shown that CG pruning achieves significantly better results than classical pruning techniques. The time consumption of the recognizer was decreased significantly without loss in speech recognition accuracy. Such improved efficiency in speech decoding is generally important for embedded recognizers with limited memory capacity and CPU power.

In addition to CG pruning with constant pruning threshold, a dynamic variant (CGD pruning) was also presented in this work. CGD pruning makes use of a dynamic threshold computed by taking the course of best hypothesis confidence into consideration. That way the pruning threshold is decreased in case of high confidence, i.e. high NLL score of the best hypothesis, and increased in case of low confidence for each time frame. It was shown that this dynamic approach achieves further improvement in pruning performance compared to CG pruning. In comparison with an alternative pruning technique, which uses approaches of the field of adaptive control, CGD pruning again shows better results. Above all, it provides more stability in steering the dynamic course of the pruning threshold, especially at the beginning of the utterance where the adaptive control approach is negatively affected by transient oscillations.

Utilization of confidence measures at the dialog level was also demonstrated on global concepts of dialog management with its underlying architectures in speech-based human-machine interaction. The thesis discussed current strategies and implementation techniques for the use of CM in live operational environments. Possible sources of problems in speech-based interaction were pointed out and classified.

It was shown that the degree of naturalness is a highly important design aspect for speech applications because user acceptance and ergonomics demand as much naturalness as possible in order to approach dialogs resembling those of human interactions. Especially in the field of application development, CM techniques play a principal role and hold vast potential for future improvements, in particular at higher levels of speech processing, i.e. as reliable semantic confidence scores. Improvements in error recovery of interactive voice-enabled applications could be achieved by incorporating reliable prediction features of the user's emotion/mood during the dialog. If it detects a bad mood or anger, the machine could pass the conversation along to a human operator in time to ameliorate bad user experience.

Success and acceptance of voice-enabled applications also depend on user expectations. Both sides, voice interface design and users, are currently standing at the beginning of a learning curve. Speech applications definitely require significant improvements but users on the other hand also need to learn what they can realistically expect from a specific application. If users' expectations are too high, communication failures or misunderstandings are practically inevitable.

In recent years, tremendous strides have been made in further increasing the performance of computers. There is a clear tendency to build standard computers with multiple CPU cores and huge amounts of memory, which means that enhanced computer architecture is becoming common in the short term. This fact also introduces new possibilities in the field of speech recognition, especially its utilization in practice. One needs to think differently in order to take full advantage of all the possibilities provided by parallel CPU architectures. But I am sure that simultaneous recognition tasks will become standard practice in order to realize many new ideas dealing with multiple knowledge sources processed in parallel, whose multiple results are then merged in a final computation step. Especially in the field of confidence measurement, novel ideas that were heretofore inconceivable due to lack of practicable computer architecture will become feasible to further the advancement along the long road of speech recognition development. Due to their importance for speech applications, I expect that CM techniques will remain one of the focuses of future research activities in spite of the remarkable improvements already achieved in the quality and applicability in estimation of reliable confidence measures for human-machine communication.

Appendices

Appendix A

The ROC Curve

Receiver Operating Characteristic or simple *ROC curves* were developed originally as a byproduct of research in the field of radio signals in the 1950's. They are, however, remarkably useful in any kind of decisionmaking and also in ASR systems. In the field of speech recognition ROC curves are widely used to plot error rates against each other in order to find appropriate confidence thresholds for the classification of the recognition result.

Once the recognition of a speech utterance is completed, the next important task is to make a decision about its correctness compared with the content of the speech input. In practice, the speech content is unknown and therefore the decision about the correctness of the ASR result can only be made based on its confidence score and on a specific confidence threshold. If the confidence score falls below the threshold, the recognition result is considered as incorrectly recognized (rejected); otherwise as correctly recognized (accepted). This simple decision rule can lead to four possible cases which take the correctness of the decision itself into consideration as shown in Table A.1:

ASR Result	Incorrect	Correct
Rejected	**CR**	**FR**
Accepted	**FA**	**CA**

Table A.1: Confusion matrix of decision made by confidence threshold (rejected/accepted) against recognition status (incorrect/correct).

- *Correct rejection* (CR) refers to the case if the recognition result is rejected because its confidence score falls below the threshold and the recognition result is also different from the speech input.

- *False rejection* (FR) signifies, in contrast to CR, the incorrect decision about the rejection of the recognition result even though it is consistent with the speech input.

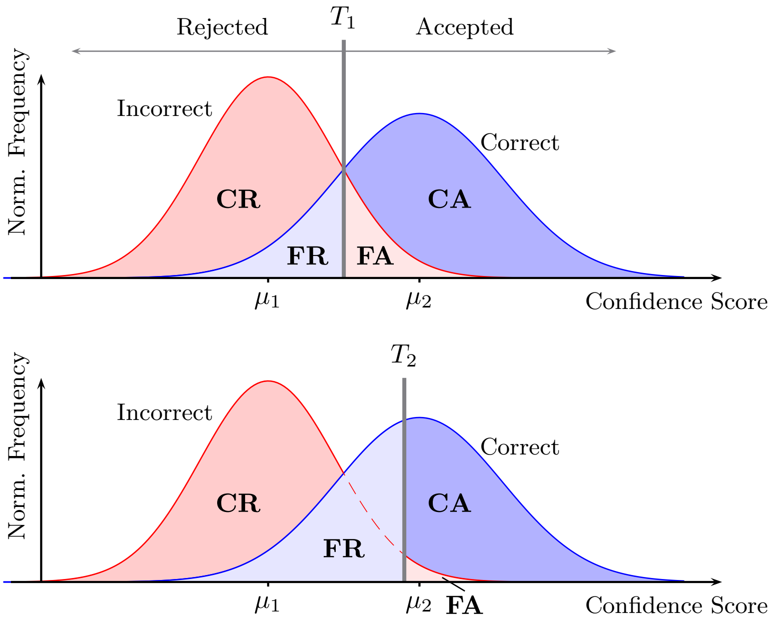

Figure A.1: Distribution of correct and incorrect ASR results depending on their confidence scores modeled by Gaussian distribution functions. The vertical lines T_1 and T_2 mark confidence threshold examples. The colored areas CR, FR, FA and CA of the confusion matrix in Table A.1 are also shown for these two examples of different confidence thresholds.

- *False acceptance* (FA) means that the recognition result is accepted because its confidence score meets or exceeds the threshold, even though the ASR result and the speech input do not match.
- *Correct acceptance* (CA) signifies, in contrast to FA, the correct decision for the acceptance of the recognition result which is also consistent with the speech input.

These four possible cases are pictured in greater detail in Figure A.1. For these plots the assumption was made that the distribution of confidence scores which can be described with Gaussian distribution functions for both correct and incorrect recognition results. The Gaussians have their mean values at different confidence scores μ_1 and μ_2, and they also have different deviations. Correct recognition results have a higher mean confidence score than incorrect results.

The vertical lines T_1 and T_2 are examples of possible confidence thresholds. The ratio between the areas CR, CA, FR and FA varies with the position of T on the x-axis. The overlap between the distributions consists of false rejected (FR) and false accepted (FA) regions. These regions correspond to those recognition results for which the confidence threshold is not appropriate for decisionmaking and would therefore lead to misclassifications. In other words, the overlap between the distribution curves makes a clear classification of the recognition results as accepted or rejected impossible at a specific confidence threshold T.

In Figure A.1 T_1 is located exactly midway between the mean values μ_1 and μ_2. As a result, mistakes in deciding correctness of the recognition result, FR and FA, are almost the same as for the example distribution functions; but only approximately so due to the different deviations of the distribution functions. The case $\textit{FA Rate} = \textit{FR Rate}$ is often referred to as *equal error rate* (EER).

T_2 however, marks a higher confidence threshold than T_1 and it allows less recognition results to be accepted in general; this means lower CA and FA rates. As a result this stricter decision rule reduces the error rate in accepting incorrect results (FA) but on the other hand it improves the rate of decision failures in rejecting correct results (FR).

Answering the question “Which confidence threshold fits a specific speech application best?” is always about finding the appropriate trade-off between FR and FA depending on operational requirements:

- *Robustness* against a noisy environment, for example, is very important if the speech application operates in the telecommunication sector dealing with mobile calls.

- *Security* aspects have high priority for dialogs that manage bank or stock accounts.

- *Voice user interface* generally has the primary goal of achieving the highest possible user acceptance in human-machine interaction.

These requirements often lead to stricter decision rules and as a consequence, to higher confidence thresholds in order to avoid any misunderstanding during the dialog or to prevent a negative outcome. The ROC curve is a very useful instrument for finding the best solution for such operational requirements. There are two types of ROC curves which are widely used:

- Plot of a *detection rate* against an *error rate*; in other words, in the context of confidence decision in speech recognition, the plot of CA, on the y-axis, against FA, on the x-axis, or analogously the plot of CR against FR.

- Plot of one error rate against another error rate which is also called a *detection error trade-off* (DET) curve. In our context this means the plot of FR on the y-axis, against FA on the x-axis, or vice versa.

Figure A.2 on page 102 shows an example ROC curve. On the vertical axis *FR Rate* represents the number of false rejections in relation to the total number of correctly recognized results:

$$\textit{FR Rate} = \frac{N(\textit{rejected} \mid \textit{correct})}{N(\textit{correct})}. \tag{A.1}$$

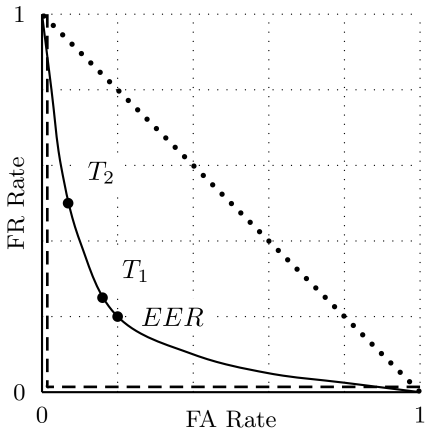

Figure A.2: ROC examples; the confidence thresholds T_1 and T_2 in Figure A.1 on page 100 are also marked here schematically. The point of EER is also indicated as well as the decision cases *ideal* and *guessing* as dashed and dotted lines.

On the horizontal axis *FA Rate* represents the number of false acceptances in relation to the total number of incorrect results:

$$FA\ Rate = \frac{N(accepted \mid incorrect)}{N(incorrect)}. \tag{A.2}$$

The ROC curves in Figure A.2 are generated while varying the confidence threshold for given confidence score distributions. For each threshold the error rates of FR and FA are computed and pictured in a diagram. The ROC curve allows an exploration of what happens to FR and FA while varying the position of the confidence threshold. Figure A.2 contains example ROC curves for three different situations of distribution overlaps. The solid line shows the situation schematically as described in Figure A.1 on page 100. If the confidence threshold is moved toward from higher to lower values ($T_2 \rightarrow T_1$), the number of false rejections will decrease rather rapidly at first, i.e. the ROC curve moves down steeply. Finally, it reaches a region where there is a remarkable decrease in false acceptance; the ROC curve flattens out if the confidence threshold is moved to very low values.

The smaller the overlap between the distribution functions, the more steeply the ROC curve moves down or flattens out. The ideal case is a curve that adheres to the y and x axes, shown as a dashed line in Figure A.2. On the other hand the closer the curve is to the diagonal, the bigger the overlap and as a consequence the less discriminative the decision can be based on a confidence threshold — similar to guessing. The dotted line in Figure A.2 represents this guessing case when the Gaussians from Figure A.1 on page 100 had a total overlap ($\mu_1 = \mu_2$) and also the same deviations.

Stated more precisely, the correct way of characterizing the discriminative capability of the distribution functions is to look at the area below the ROC curve. The closer the area is to 0.5 (diagonal case), the worse the discriminative capability (guessing); the closer it is to 0 (ideal case), the better the discriminative capability.

Appendix B

Grammar Examples

Industrial standards allow the specification of ASR grammars (language models), in two forms, augmented BNF (ABNF) form or XML form (see W3C-Grammar, 2004). ABNF has its roots in the Backus-Naur form (BNF), a formal mathematical description language originally developed to describe the syntax of the programming language Algol 60. ABNF is a plain text representation, whereas the XML grammar form uses elements of the Extensible Markup Language. Both representations are semantically mappable and allow automatic transformation between the two forms.

The example below presents both standard formats ABNF and XML for a simple grammar that accepts a 4-digit number, e.g. for PIN recognition. The ABNF format is as follows:

```
#ABNF 1.0 ISO-8859-1;
language en-US;
mode voice;

$digit = zero | one | two | three | four | five | six |
         seven | eight | nine;
public $number= $digit <4>;
```

The corresponding XML grammar is:

```
<?xml version="1.0"?>
<grammar mode="voice"version="1.0"encoding="ISO-8859-1
         xml:lang="en-US"xmlns="http://www.w3.org/2001/06/grammar">

<rule id="number"scope="public">
   <item repeat="4"><ruleref uri="#digit"/></item>
</rule>
```

```
<rule id="digit">
  <one-of>
     <item>   zero    </item>
     <item>   one     </item>
     <item>   two     </item>
     <item>   three   </item>
     <item>   four    </item>
     <item>   five    </item>
     <item>   six     </item>
     <item>   seven   </item>
     <item>   eight   </item>
     <item>   nine    </item>
  </one-of>
</rule>
</grammar>
```

Bibliography

Abdou, S., & Scordilis, M.S. 2003. An Efficient, Fast Matching Approach Using Posterior Probability Estimates in Speech Recognition. *Pages 1161–1164 of: Proceedings of the 8th European Conference on Speech Communication and Technology (EUROSPEECH)*, Geneve, Switzerland.

Abdou, S., & Scordilis, M.S. 2004. Beam Search Pruning in Speech Recognition Using a Posterior Probability Based Confidence Measure. *Journal of Speech Communication*, 42(3-4), Pages 409–428.

Astrom, K.J., & Wittenmark, B. 1995. *Adaptive Control.* Second edn. Prentice Hall. ISBN 978-0201558661.

Bohus, D., & Rudnicky, A.I. 2005. A Principled Approach for Rejection Threshold Optimization in Spoken Dialog Systems. *Pages 2781–2784 of: Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, Lisbon, Portugal.

Boite, J.M., Bourlard, H., D'hoore, B., & Haesen, M. 1993. A New Approach Towards Keyword Spotting. *Pages 1273–1276 of: Proceedings of the 3th European Conference on Speech Communication and Technology (EUROSPEECH)*, Berlin, Germany.

Bouwman, G., & Boves, L. 2001. Using Information on Lexical Stress for Utterance Verification. *Pages 29–34 of: Proceedings of ITRW on Prosody in ASRU*, St. Thomas, U.S. Virgin Islands.

Bub, T., & Schwinn, J. 1996. VERBMOBIL The Evolution of a Complex Speech-to-Speech Translation System. *Pages 2371–2374 of: Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, Pennsylvania.

Cavedon, L., Purver, M, & Ratiu, F. 2005. Combining Confidence Scores with Contextual Features for Robust Multi-Device Dialogue. *In: Proceedings of the 3rd Australasian Language Technology Workshop (ALTA)*, Sydney, Australia.

CHARLET, D. 2001. Confidence-Measure-Driven Unsupervised Incremental Adaptation for HMM-Based Speech Recognition. *Pages 357–360 of: Proceedings of the 26th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Salt Lake City, Utah.

CHARLET, D., MERCIER, G., & JOUVET, D. 2001. On Combining Confidence Measures for Improved Rejection of Incorrect Data. *Pages 2113–2116 of: Proceedings of the 7th European Conference on Speech Communication and Technology (EUROSPEECH)*, Aalborg, Denmark.

COLIBRO, D., FISSORE, L., VAIR, C., DALMASSO, E., & LAFACE, P. 2005. A Confidence Measure Invariant to Language and Grammar. *Pages 1001–1004 of: Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, Lisbon, Portugal.

FABIAN, T. 1999. *Sprecheradaption in der automatischen Spracherkennung.* Bad Iburg: Der Andere Verlag. ISBN 3980653706.

FABIAN, T., & RUSKE, G. 2006. Comparing Confidence-Guided and Adaptive Dynamic Pruning Techniques for Speech Recognition. *In: Proceedings of the 14th European Signal Processing Conference (EUSIPCO)*, Florence, Italy.

FABIAN, T., & VICSI, K. 1999. Akustische Struktur der ungarischen Vokale aus der Sicht der Leistungsfähigkeit der Spracherkennung. *Akustische Rundschau*, III(1-3), Pages 30–35.

FABIAN, T., PFAU, T., & RUSKE, G. 2001. Analysis of N-Best Output Hypotheses for Fast Speech in Large Vocabulary Continuous Speech Recognition. *Pages 2535–2538 of: Proceedings of the 7th European Conference on Speech Communication and Technology (EUROSPEECH)*, vol. 4, Aalborg, Denmark.

FABIAN, T., LIEB, R., RUSKE, G., & THOMAE, M. 2003. Impact of Word Graph Density on the Quality of Posterior Probability Based Confidence Measures. *Pages 917–920 of: Proceedings of the 8th European Conference on Speech Communication and Technology (EUROSPEECH)*, Geneva, Switzerland.

FABIAN, T., LIEB, R., RUSKE, G., & THOMAE, M. 2005. A Confidence-Guided Dynamic Pruning Approach -Utilization of Confidence Measurement in Speech Recognition-. *Pages 585–588 of: Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, Lisbon, Portugal.

FALAVOGNA, D., GRETTER, R., & RICCARDI, G. 2002. Acoustic and Word Lattice Based Algorithms for Confidence Scores. *Pages 1621–1624 of: Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, Pennsylvania.

Forney, D.G. 1973. The Viterbi Algorithm. *Proceedings of The IEEE*, 61(3), Pages 268–278.

Goronzy, S., Marasek, K., Haag, A., & Kompe, R. 2000. Phone-Duration-Based Confidence Measures for Embedded Applications. *Pages 500–503 of: Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP)*, Beijing, China.

Guillevic, D., Gandrabur, S., & Normandin, Y. 2002. Robust Semantic Confidence Scoring. *Pages 853–856 of: Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, Pennsylvania.

Hacioglu, K., & Ward, W. 2002. A Concept Graph Based Confidence Measure. *Pages 225–228 of: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Orlando, Florida.

Harper, M.P., & Helzerman, R.A. 1995. Extensions to Constraint Dependency Parsing for Spoken Language Processing. *Computer Speech and Language*, 9(3), Pages 187–234.

Hermansky, H. 1990. Perceptual Linear Predictive (PLP) Analysis of Speech. *The Journal of the Acoustic Society of America*, 87, Pages 1738–1752.

Hermansky, H., & Junqua, C. 1988. Optimization of Perceptually-based ASR Front-end. *Pages 219–222 of: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New York, New York.

Hermansky, H., & Morgan, N. 1994. RASTA Processing of Speech. *Speech and Audio Processing, IEEE Transactions*, 2(4), Pages 578–589.

Jiang, H. 2005. Confidence Measures for Speech Recognition: A survey. *Journal of Speech Communication*, 45(4), Pages 455–470.

Johnson, M.T., & Harper, M.P. 1999. Near Minimal Weighted Word Graphs for Post-Processing Speech. *Pages 249–252 of: Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Keystone, Colorado.

Junkawitsch, J., Ruske, G., & Höge, H. 1997. Efficient Methods for Detecting Keywords in Continuous Speech. *Pages 259–262 of: Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece.

Kamppari, S. O., & Hansen, T.J. 2000. Word and Phone Level Acoustic Confidence Scoring. *Pages 1894–1897 of: Proceedings of the 25th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Istanbul, Turkey.

KEMP, T., & SCHAAF, T. 1997. Estimating Confidence Using Word Lattices. *Pages 827–830 of: Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece.

KEMP, T., & WAIBEL, A. 1999. Unsupervised Training of a Speech Recognizer: Recent Experiments. *Pages 2725–2728 of: Proceedings of the 6th European Conference on Speech Communication and Technology (EUROSPEECH)*, Budapest, Hungary.

KETABDAR, H., HANNEMANN, M., & HERMANSKY, H. 2007. Detection of Out-of-Vocabulary Words in Posterior Based ASR. *Pages 1757–1760 of: Proceedings of the 10th European Conference on Speech Communication and Technology (EUROSPEECH)*, Antwerp, Belgium.

KULLBACK, S., & LEIBLER, R.A. 1951. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1), Pages 79–86.

KUWABARA, H. 1997. Acoustic and Perceptual Properties of Phonemes in Continuous Speech as a Function of Speaking Rate. *Pages 1003–1006 of: Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece.

LEE, C.H. 1997. A Unified Statistical Hypothesis Testing Approach to Speaker Verification and Verbal Information Verification. *Pages 63–72 of: Proceedings of COST250*, Rhodes, Greece.

LI, X., & HUERTA, J.M. 2007. How Predictable is ASR Confidence in Dialog Applications? *Pages 1745–1748 of: Proceedings of the 10th European Conference on Speech Communication and Technology (EUROSPEECH)*, Antwerp, Belgium.

LIEB, R. 2006. *Eficient Integration of Hierarchical Knowledge Sources and the Estimation of Semantic Confidences for Automatic Speech Interpretation.* Ph.D. thesis, Technische Universität München, Munich, Germany.

LIEB, R., FABIAN, T., RUSKE, G., & THOMAE, M. 2004. Estimation of Semantic Confidences on Lattice Hierarchies. *Pages 569–572 of: Proceedings of the 8th International Conference on Spoken Language Processing (ICSLP)*, Jeju Island, Korea.

LIU, F., AFIFY, M., JIANG, H., & SIOHAN, O. 2001. A New Verification-Based Fast Match Approach to Large Vocabulary Continuous Speech Recognition. *Pages 851–854 of: Proceedings of the 7th European Conference on Speech Communication and Technology (EUROSPEECH)*, Aalborg, Denmark.

Lopez, D.P.P., & Mateo, C.G. 2005. Application of Confidence Measures for Dialogue Systems through the Use of Parallel Speech Recognizers. *Pages 2785–2788 of: Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, Lisbon, Portugal.

Makhoul, J. 1975. Spectral Linear Prediction: Properties and Applications. *IEEE Transactions ASSP-23*, Pages 283–296.

Mangu, L., & Brill, E. 1999. Lattice Compression in the Consensual Post-Processing Framework. *Pages 246–252 of: In Proceedings of the 3rd World Multiconference on Systemics, Cybernetics and Informatics Joint with the 5th International Conference on Information Systems Analysis and Synthesis*, vol. 5, Orlando, Florida.

Mangu, L., Brill, E., & Stolcke, A. 1999. Finding Consensus Among Words: Lattice-Based Word Error Minimization. *Pages 495–498 of: Proceedings of the 6th European Conference on Speech Communication and Technology (EUROSPEECH)*, Budapest, Hungary.

Mangu, L., Brill, E., & Stolcke, A. 2000. Finding Consensus in Speech Recognition: Word Error Minimization and Other Application of Confusion Networks. *Computer Speech and Language*, 14(4), Pages 373–400.

Martinez, F., Tapias, D., Alvarez, J., & Leon, P. 1997. Characteristics of Slow, Average and Fast Speech and Their Effects in Large Vocabulary Continuous Speech Recognition. *Pages 649–672 of: Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece.

Mengusoglu, E., & Ris, C. 2005. Use of Acoustic Prior Information for Confidence Measure in ASR. *In: Acoustics Research Letters Online*, Acoustical Society of America Digital Library.

Möller, S., Engelbrecht, K.P., & Oulasvirta, A. 2007. Analysis of Communication Failures for Spoken Dialogue Systems. *Pages 134–137 of: Proceedings of the 10th European Conference on Speech Communication and Technology (EUROSPEECH)*, Antwerp, Belgium.

Ortmanns, S., Eiden, A., Ney, H., & Coenen, N. 1997. Look-Ahead Techniques for Fast Beam Search. *Pages 1783–1786 of: Proceedings of the 22nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, Munich, Germany.

Pfau, T. 2000. *Methoden zur Erhöhung der Robustheit automatischer Spracherkennungssysteme gegenüber Variationen der Sprechgeschwindigkeit.* Ph.D. thesis, Technische Universität München, Munich, Germany.

Rabiner, L. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of IEEE*, 77(2), Pages 257–286.

Rabiner, L., & Juang, B.H. 1993. *Fundamentals of Speech Recognition.* New Jersey: Prentice Hall PTR. ISBN 0130151572.

Reichl, W., Harengel, S., Wolfertstetter, F., & Ruske, G. 1996. Neural Networks for Nonlinear Discriminant Analysis in Continuous Speech Recognition. *Verbmobil Report No. 111.*

Renals, S., & Hochberg, M. 1999. Start-Synchronous Search for Large Vocabulary Continuous Speech Recognition. *Speech and Audio Processing, IEEE Transactions*, Pages 542–553.

Rose, R., Yao, H., Roccardi, G., & Wright, J. 1998. Integration of Utterance Verification with Statistical Language Modeling and Spoken Language Understanding. *Pages 237–240 of: Proceedings of the 23nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Seattle, Washington.

Rose, R.C. 1992. Discriminant Wordspotting Techniques for Rejecting Non-vocabulary Utterances in Unconstrained Speech. *Pages 105–108 of: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, San Francisco, California.

Roukos, S. 1995. Language Representation. *Survey of the State of the Art in Human Language Technology*, Pages 28–33. http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html.

Ruske, G. 1994. *Automatische Spracherkennung.* Second edn. München, Wien: Oldenburg. ISBN 3486227947.

San-Segundo, R., Pellom, B., Hacioglu, K., & Ward, W. 2001. Confidence Measures For Spoken Dialogue Systems. *In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Salt Lake City, Utah.

Sarikaya, R., Yuqing, G., Picheny, M., & Erdogan, H. 2005. Semantic Confidence Measurement for Spoken Dialog Systems. *Speech and Audio Processing, IEEE Transactions*, 13(4), Pages 534–545.

Schukat-Talamazzini, E.G. 1995. *Automatische Spracherkennung.* Brausnchweig, Wiesbaden: Vieweg & Sohn Verlagsgeselschaft GmbH. ISBN 3528054921.

Sixtus, A., & Ortmanns, S. 1999. High Quality Word Graphs Using Forward-Backward Pruning. *Pages 593–596 of: Proceedings of the 24nd IEEE International*

Conference on Acoustics, Speech and Signal Processing (ICASSP), Phoenix, Arizona.

Sun, H., Zhang, G., Zheng, F., & Xu, M. 2003. Using Word Confidence Measure for OOV Words Detection in a Spontaneous Spoken Dialog System. *Pages 2713–2716 of: Proceedings of the 8th European Conference on Speech Communication and Technology (EUROSPEECH)*, Geneva, Switzerland.

Thomae, M., Fabian, T., Lieb, R., & Ruske, G. 2003. A One-Stage Decoder for Interpretation of Natural Speech. *Pages 56–64 of: Proceedings of the IEEE International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE)*, Beijing, China.

Tran, B.H., Steinbiss, V., & Ney, H. 1994. Improvement in Beam Search. *Pages 2143–2146 of: Proceedings of the 3rd International Conference on Spoken Language Processing (ICSLP)*, Yokohama, Japan.

Uhrik, C., & Ward, W. 1997. Confidence Metrics Based on N-Gram Language Model Backoff Behaviors. *Pages 2771–2774 of: Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece.

Van Hamme, H., & Van Aelten, F. 1996. An Adaptive-Beam Pruning Technique for Continuous Speech Recognition. *Pages 2083–2086 of: Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, Pennsylvania.

Vanhoucke, V. 2005. Confidence Scoring and Rejection using Multi-Pass Speech Recognition. *Pages 3133–3136 of: Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, Lisbon, Portugal.

W3C-Grammar. 2004. Speech Recognition Grammar Specification Version 1.0. http://www.w3.org/TR/speech-grammar.

W3C-SISR. 2007. Semantic Interpretation for Speech Recognition (SISR) Version 1.0. http://www.w3.org/TR/semantic-interpretation.

Wallhoff, F., Willett, D., & Rigoll, G. 2000. Frame Discriminative and Confidence-driven Adaptation for LVCSR. *Pages 1835–1838 of: Proceedings of the 25th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Istanbul, Turkey.

Weber, F. 2002. *Untersuchung verschiedener Konfidenzmaße für die automatische Spracherkennung*. Diploma thesis, Technische Universität München, Munich, Germany.

Weintraub, M., Beaufays, F., Rivlin, Z., Konig, Y., & Stolcke, A. 1997. Neural-Network Based Measures of Confidence for Word Recognition. *Pages 887–890 of: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Munich, Germany.

Wessel, F., Macherey, K., & Ney, H. 1999. A Comparison of Word Graph and N-best List Based Confidence Measures. *Pages 315–318 of: Proceedings of the 6th European Conference on Speech Communication and Technology (EUROSPEECH)*, Budapest, Hungary.

Wessel, F., Schlüter, R., Macherey, K., & H., Ney. 2001. Confidence Measures for Large Vocabulary Continuous Speech Recognition. *IEEE Transactions on Speech and Audio Processing*, 9(3), Pages 288–298.

Willet, D., Worm, A., Neukirchen, C., & Rigoll, G. 1998. Confidence Measures for HMM-Based Speech Recognition. *Pages 525–528 of: Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP)*, Sidney, Australia.

Williams, D.A.G. 1999. *Knowing What You Don't Know: Roles for Confidence Measure in Automatic Speech Recognition.* Ph.D. thesis, University of Sheffield, Sheffield, UK.

Williams, G., & Renals, S. 1997. Confidence Measure for Hybrid HMM/ANN Speech Recognition. *Pages 1955–1958 of: Proceedings of the 5th European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece.

Young, S.J. 1994a. The HTK Hidden Markov Model Toolkit: Design and Philosophy. *In: Technical Report, Department of Engineering*, Cambridge University (UK).

Young, S.R. 1994b. Look-Ahead Techniques for Fast Beam Search. *Pages 21–24 of: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, Adelaide, Australien.

Yu, D., Ju, Y.C., Wang, Y.Y., Zweig, G., & Acero, A. 2007. Automated Directory Assistance System - from Theory to Practice. *Pages 2709–2712 of: Proceedings of the 10th European Conference on Speech Communication and Technology (EUROSPEECH)*, Antwerp, Belgium.

Zhang, D., & Du, L. 2004. Dynamic Beam Pruning Strategy Using Adaptive Control. *Pages 285–288 of: Proceedings of the 8th International Conference on Spoken Language Processing (ICSLP)*, Jeju Island, Korea.

Zhang, R., & Rudnicky, A. 2001. Word Level Confidence Annotation Using Combinations of Features. *Pages 2105–2108 of: Proceedings of the 7th European Conference on Speech Communication and Technology (EUROSPEECH)*, Aalborg, Denmark.

Zhang, R., Al-Bawab, Z., Chan, A., Chotimongkol, A., Huggins-Daines, D., & Rudnicky, A.I. 2005. Investigations on Ensemble Based Semi-Supervised Acoustic Model Training. *Pages 1677–1680 of: Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH)*, Lisbon, Portugal.

Zwicker. 1982. *Psychoakustik*. Berlin: Springer-Verlag. ISBN 3540114017.

Index