Robust Acoustic and Semantic Modeling in a Telephone-based Spoken Dialog System

Kinfe Tadesse Mengistu (M. Sc.)

Der Fakultät für Elektrotechnik und Informationstechnik

der Otto-von-Guericke Universität Magdeburg

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

am 27 April 2009 vorgelegte Dissertation

Declaration

I hereby declare that this dissertation is my autonomous work and that, to the best of my knowledge, it contains no material previously published or written by another person, except where due reference is made in the thesis. Any contribution made to the research by others, with whom I have worked is explicitly acknowledged.

This work has not been submitted as a dissertation to any other German or foreign institution and has not been previously published as a whole.

April 27, 2009 Magdeburg

To my wife Yeshi and my daughter Eden for their patience, love and encouragement!

Acknowledgements

Words are inadequate to express my gratitude to the Almighty God for all He has done to me in my life. I would like to express my heartfelt gratitude to Him in the first place.

I owe a debt of gratitude to my supervisor Prof. Andreas Wendemuth for critically reviewing this manuscript and enriching it with his constructive criticisms. He has been much more than just a supervisor. Without him, neither the start nor the completion of this research work would have been possible. Many thanks are due to him!

I am greatly indebted to Mathias Mamsch for his helpful ideas during the inception of this project and Mirko Hannemann who had a remarkable contribution in extending the semantic model we propose in this thesis with various features. Thanks also go to Tobias Baum who wrote the scripts for database access and dialog management under my supervision.

I sincerely acknowledge all members of the Cognitive Systems group for being very kind, friendly and cooperative in all possible ways during my journey to this end.

My earnest gratitude goes to my wife Yeshi who sacrificed her personal aspirations in pursuit of mine. She has always been there all the time when I needed someone to share my difficulties. Special thanks are due to my daughter Eden for her love and refreshing smile. Among the nice people I have known in Magdeburg, the kindness that Mrs. Opl has shown us is worth appreciation. I would also like to acknowledge the moral support and encouragement of my family, relatives and friends.

Finally, my sincere thanks go to the federal state of Saxony Anhalt for its financial support during most part of this work.

Abstract

In this thesis, we investigate various robustness issues of speech recognition and spoken language understanding models in a telephone-based spoken dialog system. We also show the feasibility of building a robust, multi-domain telephone-based spoken dialog system framework that can be seamlessly used for multiple application domains in different languages while using domain-dependent resources.

In automatic speech recognition, some level of robustness can be achieved by using domain-specific recognition resources. In addition, the existence of considerable acoustic similarity within speakers of the same gender, accent, and age-group suggests that the use of user-group dependent acoustic models can give improved recognition performance. In this thesis, we group users based on gender and accent to exploit the shared vocal characteristics of speakers in the same group. We show that a tremendous performance boost can be obtained by efficiently tailoring gender-dependent acoustic models trained on native US-English speech data to the particular vocal characteristics of German-accented English speakers. We also demonstrate the effectiveness of cross-language accent adaptation where native German enrollment data is used to adapt native US-English acoustic models to the German accent. To use group-dependent acoustic models, one has to discern the group (i.e., gender and/or accent) of a speaker from a spoken utterance. Most current approaches to accent recognition use accented speech data to train an accent recognizer. In this thesis, we demonstrate a high-performance accent recognizer that can be trained on a merger of native speech data of two or more languages. We also build a gender recognizer using cepstral features and effectively use it to load the acoustic model corresponding to the recognized gender.

In the domain of spoken language understanding, we introduce a new approach to hierarchical semantic modeling that enriches a recognized utterance with semantic information at various levels of detail. The model is essentially built by grouping semantically and hierarchically related low-level concepts into higher level structures using prior domain knowledge and training examples. The proposed model possesses a number of features; namely, it offers a remarkable ambiguity resolution ability, high predictive power and produces a structured, semantically rich information that is convenient for dialog management. Moreover, it is robust in that it successfully deals with utterances containing unseen observations, and a significant percentage of out-of-vocabulary words can be correctly labeled using the surrounding context. Besides, the model allows us to safely ignore semantically irrelevant speech recognition errors. The model is also suited to properly handle noisy input containing false starts, filled pauses, hesitations, etc. More importantly, the model can be readily trained on completely unlabeled data with relatively less human supervision. The required additional human effort to design the proposed hierarchical model is much less than the laborious and error-prone semantic annotation of a training data set or hand-crafting a semantic grammar as no particular linguistic expertise is required. Furthermore, the resulting hierarchical model outperforms the flat-concept model and has been successfully used in our demonstration system.

We demonstrate our approaches on two corpora in two application domains; namely, airline travel planning in English (©2001 Communicator Evaluation) and train information inquiries in German (©ERBA). After carefully building the required recognition resources for each application, we evaluate the performance of the models in real-time use and the usability of the system as a whole with actual test users. In general, the results obtained are promising while rooms for improvement have been identified.

Most of the presented approaches in this thesis have been published in appropriate international media.

Table of Contents

TABLE OF CONTENTS

List of Figures

List of Tables

List of Abbreviations

API	Application Programming Interface
ASR	Automatic Speech Recognition
ATIS	Airline Travel Information System
ATK	Application Toolkit for HTK
BRI	Basic Rate Interface
CAPI	Common ISDN Application Programming Interface
CCXML	Call Control eXtensible Markup Language
CFG	Context Free Grammar
CHRONUS	Conceptual Hidden Representation of Natural Unconstrained Speech
CMN	Cepstral Mean Normalization
DARPA	Defense Advanced Research Projects Agency
DCT	Discrete Cosine Transform
EM	Expectation-Maximization
ERBA	Erlanger Bahn Ansage
FFT	Fast Fourier Transform

FIA	Form Interpretation Algorithm
FSM	Finite State Machine
GMM	Gaussian Mixture Model
GUI	Graphical User Interface
GUS	Genial Understander System
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
HTTP	Hypertext Transfer Protocol
HUM	Hidden Understanding Model
ISDN	Integrated Services Digital Network
KQML	Knowledge Query and Manipulation Language
LMSF	Language Model Scaling Factor
LPC	Linear Predictive Coding
LPCC	Linear Prediction Cepstral Coefficients
MAP	Maximum a Posteriori
MFCC	Mel-Frequency Cepstral Coefficient
MLLR	Maximum Likelihood Linear Regression
OOV	Out-of-Vocabulary
P	Precision
PARADISE	PARAdigm for DIalog System Evaluation
PCM	Pulse Code Modulation
PLP	Perceptual Linear Prediction

PRI	Primary Rate Interface
R	Recall
RASTA-PLP	Relative Spectral Transform - Perceptual Linear Prediction
SASSI	Subjective Assessment of Speech System Interfaces
SI	Speaker-Independent
SISR	Semantic Interpretation for Speech Recognition
SLF	Standard Lattice Format
SLU	Spoken Language Understanding
SRGS	Speech Recognition Grammar Specification
SUNDAIL	Speech UNderstanding in DIALog
TTS	Text-to-Speech
URI	Uniform Resource Identifier
VoiceXML	Voice eXtensible Markup Language
W3C	World Wide Web Consortium
WER	Word Error Rate
WIP	Word Insertion Penalty
XML	eXtensible Markup Language

Chapter 1

Introduction

Speech is an efficient, high-capacity output channel that is mostly the preferred modality of communication in human-to-human interaction. Speech can also serve as a high-capacity input medium in human-machine interaction as most computer users speak much faster than they can type. As a communication medium, speech is the preferred mode of interaction in hands and eyes busy environment (e.g. driving), for communication across distances (e.g. through the telephone), in situations where input terminals or keyboards are unavailable and more importantly to people with visual disabilities. However, speech is not always the best communication modality for human-machine interaction due to the inherently sequential nature of speech. In particular, unlike Web-based graphical user interfaces, there is no way to present more than one piece of information at a time. This imposes significant cognitive demand on users as they have to carefully listen to the list of available options before they can proceed to the next action. Nevertheless, with careful design, speech-enabled applications can provide the required service with reasonable performance in a more natural way in situations where speech is the preferred mode of communication as noted above.

The telephone is one of the most available and the most widely used communication device that plays a vital role for communication at a distance. The ubiquity of speech and the availability of the telephone can be put together in a telephone-based spoken dialog system toward the goal of accessing information at anytime from anywhere. In particular, telephone-based spoken dialog systems can be very economic solutions for handling information inquiry services such as train timetable information, weather information,

airline travel planning, etc. in that they reduce waiting time, extend opening hours and users need not physically be at an Internet-connected PC or at a service station.

A telephone-based spoken dialog system mainly comprises of an interface to a telephone network to deliver calls into the system, an automatic speech recognition engine for recognizing a spoken utterance, a spoken language understanding component to extract the meaning of the recognized utterance, a mechanism for response generation, an audio output module for playing prompts and responses to the caller (e.g. via text-to-speech synthesis), and a dialog manager to orchestrate the various components.

As can be imagined, building a telephone-based spoken dialog system requires knowledge and expertise from a large spectrum of disciplines. Briefly, it involves tasks including signal processing, automatic speech recognition, spoken language understanding, speech synthesis, dialog management, telephony interface development, etc. each of which is a demanding task per se. In this thesis, we investigate various robustness and performance issues for automatic speech recognition and introduce a new approach that partly solves the spoken language understanding problem. In order to demonstrate the performance of the various models that we build in this thesis, we set up a flexible, multi-domain telephone-based spoken interaction system framework using existing standards, tools, application programming interfaces, etc.

The steps taken are briefly described here as they will be fully described in later chapters. The first task involves the identification of suitable tools, standards and APIs to realize the envisaged system. One of the first design choices made was to use the World Wide Web Consortium's (W3C) standard; namely, the Voice eXtensible Markup Language (VoiceXML) to script the dialog between the caller and the system mainly due to its convenience. To complement VoiceXML with advanced telephony control functions we use the Call Control eXtensible Markup Language (CCXML) of the same standard body. To interpret the VoiceXML and CCXML documents, we acquire a third party framework that provides VoiceXML and CCXML interpreters along with open interfaces for the integration of our own components. Then, we built our own components; namely, the telephony interface component, the automatic speech recognition module along with a grammar component and the semantic interpreter. These are then integrated within the VoiceXML framework. Once the system components are built and integrated, the corresponding recognition resources are developed where our scientific contribution lies.

1.1 Statement of the Problem

Many practical spoken dialog systems aim to provide information in a specific application domain. For instance, ATIS[1] (Hemphill et al., 1990), Pegasus (Zue et al., 1994), Mercury (Seneff and Polifroni, 2000), and the multi-site DARPA[2] Communicator systems (Walker et al., 2002) all play the role of a travel agent interacting with a user in the domain of airline travel planning and reservation. This is reasonable, as domain-dependent applications can achieve optimal performance by using domain-specific resources. However, a spoken dialog system framework should not be tied to one application domain but should be used for multiple application domains that use different domain-specific resources. The system then should be able to switch between applications and/or languages at runtime as requested by the user.

The main objective of this thesis is to investigate various robustness issues in a spoken dialog system and to build robust automatic speech recognition and spoken language understanding models. In addition to this, we show the feasibility of building a robust, plug-and-play telephone-based spoken dialog system framework that can be used for multiple application domains in different languages while using domain-specific resources.

By robustness we mean the ability of a system to gracefully deal with unseen, unexpected, and/or degraded input. For instance, channel, accent and environmental mismatch have a serious negative influence in the performance of an automatic speech recognition system. A robust system, therefore, should be one that is able to work with a wide spectrum of users including foreign-accented speakers, a variety of channel and environmental conditions, etc. without a significant loss of performance.

The automatic speech recognition component of a spoken dialog system is one of the most decisive components that determine the usefulness and user acceptance of the whole system. Therefore, building a robust and flexible speech recognizer is a key issue in the development of a telephone-based spoken dialog system. Automatic speech recognition per se is a challenging task and it becomes even more demanding when it has to be performed over the telephone due to the bandwidth limitation of the telephone channel. Telephone uses an 8 kHz audio sampling rate, which may considerably degrade the input speech. Besides, different types of telephone handsets may have varying microphone and transmission quality (Junqua and Haton, 1995) which makes the problem more difficult.

[1] Airline Travel Information System

[2] Defense Advanced Research Projects Agency

Moreover, as the speech may come from an uncontrolled environment, the background noise could degrade the input speech further.

In general, the challenges of a telephone-based automatic speech recognition task include:

- Acoustic variability that results from changes in training and testing environment.
- Intra-speaker variability due to changes in the speaker's physical and emotional state.
- Inter-speaker variability that results from differences in accent, dialect, vocal tract size and shape.
- Channel variability due to different kinds of telephone apparatus with varying microphones and transmission quality.

The best recognition performance in a spoken dialog system can, in principle, be achieved by using speaker-dependent models specifically tailored to the vocal characteristics of each user of the system using a large amount of training data from each user. This is, however, practically infeasible. On the other hand, a reasonable performance can be obtained by using a speaker-independent model trained on a huge corpus that captures a wide spectrum of speakers, environments, channels and application domains. However, such a huge training data is often unavailable. A good compromise to achieve some level of robustness and better performance is to use domain-specific and group-dependent recognition resources.

Although very essential, the correct recognition of what is said alone can hardly serve any purpose in a spoken dialog system without the correct comprehension of what is meant. A spoken language understanding (SLU) component in a spoken dialog system is responsible to extract the intention of a user from a recognized utterance. SLU can be easy for narrow application domains where users are restricted in the way they can formulate their requests and the vocabulary size is very small. However, if a spoken dialog system allows a more natural conversation, the task becomes more demanding because spontaneous speech often contains noisy input such as false starts, filled pauses, hesitations, etc. Moreover, the occurrence of words not seen in the training data of the model (i.e.

out-of-vocabulary words) become inevitable. Hence, robustness in a spoken language understanding model involves dealing with these effects of natural language. Accordingly, we introduce a robust semantic model that gracefully deals with noisy input and unseen observations. The model essentially enriches the output of the speech recognizer with semantic and hierarchical information that can later be used to easily infer the intention underlying a spoken utterance in a given dialog state.

To guarantee an effective and caller-friendly remote access to real-time information, the design of a good telephony interface is vitally important. Without a good telephony interface, any speech-enabled application will be of no or limited value to users. Dialog design or the design of the system-caller interaction can also highly influence the caller experience and productivity of the system. If the dialog is not intuitive to callers, users may lose confidence in the system and the system may become futile due to poor dialog design. Therefore, the quality of the dialogs and the dialog management strategies are as important as the quality of the other components making up a spoken dialog system. As a result, due effort needs to be put to design intuitive, natural-like dialogs to allow users to articulate their requests in a certain order in a chosen application domain.

Building systems that allow completely unconstrained and human-like natural language interactions is a very complex task as the current state of automatic speech recognition (ASR) and spoken language understanding technologies are far from what the task requires. Therefore, a system developer has to strike the right balance between the level of flexibility that must be allowed and the recognition performance of the system. One compromise is to allow varying degrees of freedom based on how well the system is working with a user; i.e., using fairly relaxed language models and dialog control strategies under normal circumstances and switching to more constrained grammars and dialog strategy when task completion is at risk.

In summary, we aim to achieve robustness through:

- The use of group-dependent acoustic models based on gender and accent to exploit the shared vocal characteristics of a group of users.
- Effective utilization of prior domain knowledge to build models that can compensate for recognition errors and natural language effects (e.g. by using dialog state-specific grammars (language models) and domain-specific semantic model).
- The use of a new, hierarchical semantic model that possesses various robustness features. The model will be discussed in detail in Chapter 6. The main features of the new semantic model is briefly summarized in the next section.

1.2 Contributions of the Thesis

The thesis covers a wide range of topics including speech recognition, automatic gender and accent identification, spoken language understanding, dialog management, and evaluation of a spoken dialog system. In addition, we set up a multi-domain telephone-based spoken dialog system framework that can use domain-specific and group-dependent recognition resources. The system is based on ©OptimTalk VoiceXML framework in which we integrated our own telephony interface component, speech recognition engine and spoken language understanding unit.

In summary, we consider the key contributions of this thesis to be the following.

1. (Chapters 6, 8): We introduce a new approach to semantic modeling that:

    - Unlike most conventional data-driven approaches to spoken language understanding, requires no semantically labeled training data.
    - Captures hierarchical relationship between concepts in an utterance.
    - Outperforms the conventional flat-concept approach in terms of performance, ambiguity resolution ability, predictive power and information richness of the output.
    - Effectively accounts for observations not seen in the training data of the semantic model.
    - Uses the encoded context to correctly label out-of-vocabulary (OOV) words.
    - Properly handles the effects of spontaneous speech such as hesitations, false starts, filled pauses, etc.
    - Allows us to safely ignore recognition errors in semantically irrelevant words and frequently confused semantically equivalent expressions (e.g. six vs. sixth, eighth vs. eight, yes vs. yeah, etc.).
    - Is easily extensible to include new requirements or business rules.
    - Can produce output at different levels of detail and is convenient for dialog management.

2. (Chapters 2, 5, 8): Most current approaches to accent recognition use a database consisting of accent sensitive phrases spoken by foreign speakers or a combination of accented speech data and acoustic features such as energy, duration, pitch, formant frequencies, etc. In this thesis, we introduce a high-performance automatic accent recognizer that can be trained on a merger of native speech data of two or more accent groups. Experimental results show that accent-related information could be effectively captured from the native language speech of a speaker. We also investigate various cepstral features in search of those that are particularly suitable for accent recognition.

3. (Chapters 2, 5, 8): Pitch is known to be a very strong cue to reliably estimate the gender of an adult speaker from a spoken utterance. However, in telephone speech the pitch information is either very weak or missing due to the band-limiting effect of the telephone channel. Therefore, we use cepstral features to build an automatic gender recognition system that can reliably discern the gender of a speaker from a single-word utterance. We also investigate various cepstral features in search of those that are particularly suitable for gender recognition in a telephone-based spoken dialog system.

4. (Chapters 2, 5, 8): Though not particularly novel, we also demonstrate a more productive use of within-language and cross-language accent adaptation to tailor an acoustic model trained on native US-English speech data to the vocal characteristics of German-accented English speakers.

5. (Chapters 5, 8): An extensive investigation in search of optimal parameters for the speech recognition models in our telephone-based spoken dialog system is also presented.

1.3 Application of Results

The primary areas of concern in this thesis are robustness issues in speech recognition and spoken language understanding components of a telephone-based spoken dialog system.

One of the outputs of this work is a generic telephone-based spoken dialog system framework that can be used with different compatible recognition resources for various application domains in possibly different languages. The framework can serve as a test-bed for various scientific investigations on speech recognition resources. Besides, the framework can be used in various application domains to provide a real service to users using a more natural mode of interaction. Furthermore, new approaches to semantic modeling and accent recognition are proposed which could be used for various application domains in any language. The performance of the system and the models we built are evaluated with actual test users under real world conditions in two application domains. The framework can also be extended for use in other applications such as speaker identification and verification task.

1.4 Organization of the Thesis

The rest of this thesis is organized as follows. In Chapter 2 we present an organized and integrated summary of literature relevant to the various topics of interest that show what has been done and the significance of our work. We present a historical overview and state-of-the-art approaches to spoken dialog systems, automatic speech recognition, user-group dependent acoustic modeling and spoken language understanding. In Chapter 3 we present a general description of the tools, methods and techniques that we will use in carrying out the research. Chapter 4 provides an overall description of the system; i.e., how the various components that make up the envisaged system are developed and put together. The architecture and a conceptual usage scenario of the system are also presented. In Chapter 5 we present the fundamentals of automatic speech recognition along with the description of the methods we use to exploit group-dependent characteristics to improve speech recognition performance. Chapter 6 describes the new, proposed approach to semantic modeling that partly solves the spoken language understanding problem. In Chapter 7 a brief description of spoken language interactions, dialog initiatives, dialog control strategies and dialog design principles is presented. The approach we used to evaluate our spoken dialog system is also described. The data used, the experiments carried out and the discussions of the results obtained in the various experiments are presented in Chapter 8. Finally, concluding remarks and recommendations are given in Chapter 9.

Chapter 2

Review of State-of-the-Art Research

2.1 Introduction

In this chapter we provide a historical overview of the various topics of interest in spoken dialog systems. We also present analysis of the methods and approaches used in spoken dialog system development, automatic speech recognition, automatic gender and accent recognition, accent adaptation techniques and spoken language understanding as they relate to our work. We also provide a brief description of the motivation and justification for the various tasks undertaken in this endeavor within the context of the state-of-the-art.

2.2 Spoken Dialog Systems

Research in dialog systems, in general, can be traced back to the 1960s. The early systems such as BASEBALL (Green et al., 1963) and LUNAR (Woods et al., 1972) were essentially question answering systems in limited domains and did not have dialog capabilities. In the 1970s systems such as SHRDLU (Winograd, 1972) and GUS (Bobrow et al., 1977) were developed that offered users the opportunity to converse with computer-based systems in order to perform a task or to get information using natural language interfaces. However, the input modality in the earliest dialog systems was typed natural language (McTear, 2004). It is since the late 1980s that spoken dialog systems have emerged as a result of the two large government funded projects; namely, the DARPA program of the

United States and the Esprit SUNDIAL[1] program of Europe (McTear, 2004). The ATIS (Hemphill et al., 1990) and the Communicator (Walker et al., 2002) projects of DARPA were mainly concerned with the domain of airline travel planning while the SUNDIAL project was concerned with flight and train schedules in English, French, German and Italian (Peckham, 1993). After SUNDIAL project, a number of projects in spoken dialog modeling have evolved such as Verbmobil (Wahlster, 1993), RAILTEL (Bennacef et al., 1995), ARISE (Os et al., 1999), DISC (Bernsen and Dybkjær, 1997), and the Philips automatic train timetable information system (Aust et al., 1995).

While ATIS and SUNDIAL projects focus on single domain inquiries and use less flexible dialog strategies, the DARPA Communicator systems are more advanced in that they support mixed initiative conversational interaction and provide meeting coordination and travel planning services.

The core of the Communicator systems is based on MIT's distributed Galaxy II architecture (Seneff et al., 1998) where a number of servers interact with each other through a hub. The Galaxy architecture is mainly composed of an audio server that answers incoming calls, plays prompts and records incoming user input; a speech recognizer to recognize spoken requests; a confidence server to detect and reject misrecognized units at the concept level using acoustic and language model features from the recognizer; a text-to-speech (TTS) synthesis engine; a language generator; a language understanding component; a dialog manager and a back-end component. At the core of the Galaxy architecture is a hub that acts as a router to send frames between servers. The use of such an architecture established a standard for dozens of groups working on dialog management and speech recognition issues in the project.

Several systems have been developed under the multi-site DARPA program. They all use the Galaxy architecture described above and target the same application domain but differ in a number of aspects. For instance:

- The CMU Communicator system (Rudnicky et al., 2000) uses the Sphinx II decoder in a real-time mode, state-specific language models, and the Phoenix parser (Ward and Issar, 1996) using domain-specific semantic grammar.

- The AT&T Communicator system (Levin et al., 2000) uses the AT&T Watson continuous speech recognition engine (Sharp et al., 1997) that supports audio barge-in

[1] Speech UNderstanding in DIALog

capabilities and the CHRONUS (Pieraccini and Levin, 1993)[1] spoken language understanding system.

- MIT's Mercury flight reservation system (Seneff and Polifroni, 2000) uses the TINA (Seneff, 1992) language understanding system and a dialog control strategy based on a set of ordered rules as a mechanism to manage complex interactions.

- The Communicator system of the University of Colorado (Ward and Pellom, 1999) uses CMU's Sphinx II recognizer with a class trigram language model, a modified version of the Phoenix parser and event-driven dialog manager in which the current context of the system is used to decide what to do next.

Jupiter (Zue et al., 2000) of MIT is another example of a conversational interface using the Galaxy architecture which provides access to online weather information for over 500 cities world wide over the telephone. Voyager (Glass et al., 1995) and Pegasus (Zue et al., 1994) of MIT are other examples of domain-specific spoken dialog systems in the domains of urban navigation and online-airline reservation, respectively.

The TRAINS (Allen et al., 1996) and its successor TRIPS (Ferguson and Allen, 1998) are other popular research efforts towards task-oriented conversational dialog systems developed at the University of Rochester. The TRAINS system involves the scheduling of a railroad freight system which is later extended to a more complex logistics and transportation problem in the TRIPS project. Like the DARPA communicator, TRIPS consists of a set of components that pass messages to one another through a hub, using the so-called Knowledge Query and Manipulation Language (KQML). The components of the system can be divided into three groups; namely, modality processing components, dialog management components and specialized reasoners. The modality processing components include speech recognition, speech generation, graphical displays and gestures while the dialog management components are responsible for managing the ongoing conversation, interpreting user communication in context and selecting the next communicative actions to perform in response. The specialized reasoners, on the other hand, help to solve hard problems such as planning courses of actions, scheduling sets of events or simulating the execution of plans.

[1] Conceptual Hidden Representation of Natural Unconstrained Speech

The spoken dialog systems discussed above differ in the architecture they use, the complexity of the task they target, the type of initiative they support, the dialog control strategy they employ and the system components they use. However, they share one common distinguishing feature – they are all domain or task oriented. When dealing with domain-specific applications, the use of domain-dependent resources is beneficial to achieve optimal recognition performance. However, a dialog system should be open to support multiple application domains without compromising performance. One of the objectives of this endeavor is, therefore, to build a robust, multi-domain spoken dialog system framework that covers multiple application domains possibly in different languages while using domain-dependent resources.

Once the the envisaged framework that can robustly carry out medium-length dialogs for multiple application domains while using domain-dependent resources is realized, we investigate the various robustness issues in telephone-based speech recognition and spoken language understanding. In particular, we introduce a new approach to spoken language understanding that essentially takes the output of the speech recognizer and semantically enriches it with hierarchically structured information which make the output convenient for dialog management.

For the sake of optimal performance, we use user-group dependent acoustic models and dialog state-specific language models. We use VoiceXML for dialog authoring and CCXML for writing the call handling policy. We use the Hidden Markov Model Toolkit (HTK) to build recognition resources and its multi-threaded API (ATK) to build a real-time speech recognizer integrated in a VoiceXML framework. We bring the convenience of VoiceXML for dialog authoring and the flexibility and power of HTK-based speech recognizers together to realize a robust telephone-based spoken language interaction system. We also aim to keep the development cost (in monetary terms) low.

2.3 Automatic Speech Recognition

Research in speech recognition technology can be traced back to the 1950s. One of the early speech recognizers is that of Bell Laboratories (Davis et al., 1952) that recognizes isolated digits from a single speaker by filtering the speech signal into first and higher formant frequency bands and measuring the formant frequencies in the vowel regions of each digit. In the 1960s several special-purpose devices were built for the purpose

of recognizing a small number of isolated words in the order of 10–100 words (Nagata et al., 1963; Sakai and Doshita, 1962). Speech recognition systems have become a topic of great interest not only to researchers but also to the general public since the inspirational movie of Stanley Kubrick "2001: A Space Odyssey" in 1968 where an intelligent computer named "HAL" spoke in a human-like voice and was able to understand fluently spoken speech (Juang and Rabiner, 2005). In 1970s advances in the use of pattern recognition ideas in speech recognition was demonstrated (Velichko and Zagoruyko, 1970), techniques of dynamic programming were advanced (Sakoe and Chiba, 1978) and the use of linear predictive coding (LPC) to speech recognition was shown (Itakura, 1975). Major advances in large vocabulary speech recognition systems have started in the 1980s mainly as a result of the advent of statistical methods such as the hidden Markov model (HMM) and stochastic language models.

In general, the various approaches that have been pursued over the years can be broadly classified in to four classes; namely, template matching, statistical methods, artificial neural networks and knowledge-based approaches. Template matching and knowledge-based approaches where competing paradigms in the 1970s. Since the 1980s, the statistical approach has become the dominant paradigm for automatic speech recognition mainly due to its superior performance and ease of modeling.

Despite the significant advances in the various fields that comprise a conversational speech recognition, a machine that can pass the Turing test (Turing, 1950) with performance comparable to humans is still not a reality. However, speech technology in general is mature enough to be successfully applied in task-oriented application domains. This study uses sub-word based HMMs and aims to achieve robust models that perform reasonably well in real-time telephone-based applications.

2.4 Group-Dependent Acoustic Models

Due to differences in articulatory mechanisms there is apparent difference between the voice of male and female speakers. At the same time, there is considerable acoustic similarity within speakers of the same gender due to similar vocal structures. This suggests that gender-specific models tailored to a group of users in the same gender can perform better than a gender-independent model.

On the other hand, non-native speakers of a language often tend to introduce some phonological and pronunciation patterns from their mother tongue while speaking a foreign language. This results in a noticeable pronunciation difference between native speakers and foreign speakers of a language. In this thesis, we refer to this linguistic phenomenon as accent. Accent is one of the most important factors that influence the performance of speaker-independent (SI) speech recognition systems next to gender (Hansen and Arslan, 1995). It has been reported in (Huang et al., 2001a) that a mismatch in accent between the speakers used in testing and training can lead to over 30% increase in word error rate (WER). Therefore, the use of accent-dependent acoustic models in a spoken dialog system is essential as people may need to communicate with the system in a language which is not their native. A number of studies have been carried out in this topic and a brief overview of related researches in the field is presented below.

Training accent-specific acoustic models using accented data is an obvious and easy approach to deal with accented speech. It has been shown in (Wang et al., 2003) that training on a relatively small amount of German-accented English from the Verbmobil conversational meeting-scheduling task resulted in significantly better performance than a model trained on a large amount of native English training material. In (Wang et al., 2003) and (Tomokiyo and Waibel, 2001), it has been shown that a model trained on a merger of in-domain native and accented data performs better on accented speech. It has also been shown in (Tomokiyo and Waibel, 2001) that applying a few more forward-backward iterations with accented data on a well-trained speaker-independent model improves recognition performance for accented speakers.

Applying speaker adaptation techniques such as Maximum Likelihood Linear Regression (MLLR) (Leggetter and Woodland, 1995a) and Maximum a Posteriori (MAP) adaptation (Gauvain and Lee, 1994) methods to adapt speaker-independent models trained on native speech data to a particular accent are other popular methods. MLLR has been successfully used in (Wang et al., 2003) and (Tomokiyo and Waibel, 2001) on German-accented and Japanese-accented English, respectively. However, in both cases only a single global transform was used to transform all models. On the German-accented English task described in (Wang et al., 2003) it has been shown that MAP adaptation performs better at decreasing WER than MLLR when more enrollment data is available. However, it has not been shown whether combining MAP and MLLR could yield further performance gain.

Cross-language accent adaptation, where native speech data of the target accent group is used as enrollment data, has been explored in (Tomokiyo and Waibel, 2001). However, no performance gain was reported. In (Wang et al., 2003) a model trained on a merger of large amount of native English and native German speech data using a common phoneme set was investigated and only slight improvement in recognition was reported. In (Liu and Fung, 2000) accent adaptation without accented data was investigated to adapt an English model to Cantonese-accented English using native Cantonese speech data and promising improvement in phoneme accuracy has been reported.

As can be observed, previous applications of MLLR to a group of German-accented speakers, use only a single global transform to adapt all models. It has also not been shown whether combining MAP and MLLR could be more useful. Therefore, in this thesis, we show a successful use of MLLR with multiple transforms where both mean and variance are transformed by using a small amount of accented data to adapt a speaker-independent model trained on native US-English speech data. We also show that using MLLR transformed models as an informative prior for MAP adaptation boosts performance. Moreover, we investigate the use of cross-language accent adaptation where native German speech from a different domain (train information inquiries) is used to adapt a speaker-independent native US-English model in the domain of airline travel planning. At the same time, we try to capture variability due to both gender and accent by adapting separate native US-English gender-dependent models to the German accent.

We also investigate the use of MLLR adaptation technique to adapt acoustic models trained on microphone-recorded data to the characteristics of the telephone channel using a small amount of telephone-recorded data.

2.5 Automatic Gender Recognition

The task of an automatic gender recognition system is to discern the gender of a person from a spoken utterance. Due to physiological differences in vocal tract length, vocal fold size, larynx thickness, etc. adult male voices have lower pitch range than adult female voices (Wu and Childers, 1991). Hence, the fundamental frequency (pitch) can be used as a strong cue for gender recognition (Hillenbrand et al., 1995; Linke, 1973; Linville and Fisher, 1985; Murry and Singh, 1980). The fundamental frequency (F0) for adult male

lies between 80–170 Hz while it lies between 150–260 Hz for adult women and between 300–500 Hz for children (Baken and Orlikoff, 2000).

The approaches to automatic gender recognition can be classified into three broad classes. The first approach uses gender-dependent features such as pitch. The second approach uses cepstral features such as Mel-Frequency Cepstral Coefficients (MFCCs) to discern the gender of a speaker from a spoken utterance. The third approach combines pitch, cepstral, prosodic and other features for improved performance.

In (Abdulla and Kasabov, 2001), average pitch frequency was used as a gender separation criterion and the system achieved 100% gender discrimination accuracy with TIMIT (Texas Instruments (TI) and Massachusetts Institute of Technology (MIT)) continuous speech corpus and Otago isolated words speech corpus. This confirms that pitch is a very strong source of information for gender identification of adult male and female speakers.

Another approach described in (Parris and Carey, 1996) combines hidden Markov models and pitch estimation giving less than 1.0% identification error rate with two seconds of speech on three British English databases. Further tests without optimization on the OGI (Oregon Graduate Institute) multi-language database resulted in an average error rate of 2.0%. Another approach based on Gaussian Mixture model in (Ting et al., 2006) combines MFCCs and pitch information to improve the performance of gender recognition and the system resulted in at most 3.3% recognition error rate on SRMC (Speaker Recognition for Mobile Communication) database.

In (Slomka and Sridharan, 1997) automatic gender identification systems using fusion of multiple knowledge sources using a linear classifier are investigated on speakers of 11 languages from the OGI speech corpus. The best reported accuracy is 98.5% averaged over all clean and adverse conditions. This suggests the use of multiple knowledge sources gives improved results in adverse acoustic conditions.

In (Harb and Chen, 2005) a system using a set of neural networks with acoustic and pitch related features is built and a classification accuracy of 90% is obtained for 1 second speech segments, independent of the language and the channel of the speech. Using multiple classifiers trained on different training data, the classification accuracy attains 98.5% for longer segments (5 seconds) on a subset of the Switchboard database.

A gender classification system proposed in (Zeng et al., 2006) is based on Gaussian mixture models using combined parameters of pitch and RASTA-PLP (Relative Spectral Transform - Perceptual Linear Prediction). The accuracy of the resulting model is 95%

on noisy speech and 98.3% on clean speech. The method is reported to be robust to noise and is independent of languages.

In (Metze et al., 2007), four approaches for age and gender recognition using telephone speech have been compared; namely, a parallel phone recognizer, a system using dynamic Bayesian networks to combine several prosodic features, a system based solely on linear prediction analysis, and Gaussian mixture models based on MFCCs. It was reported that the parallel phone recognizer is comparable to a human listener but loses performance on short utterances. The system based on prosodic features has shown little dependence on the length of the utterance.

Even though, pitch is a very strong cue to reliably estimate the gender of an adult speaker from a spoken utterance, it is often very weak or missing in telephone speech due to the band-limiting effect of the telephone channel. Hence, pitch may not be suitable for gender recognition in a telephone-based spoken dialog system. Therefore, we use cepstral features to build our automatic gender recognition system based on Gaussian Mixture Model (GMM). We investigate various cepstral features; namely, Mel-Frequency Cepstral Coefficients (MFCCs), Linear Prediction Cepstral Coefficients (LPCC) and Perceptual Linear Prediction (PLP) features in search of those that are better suited for gender identification. The resulting system should be capable of recognizing the gender of a speaker given the first, very short, single word utterance with acceptable accuracy.

2.6 Automatic Accent Recognition

The purpose of an automatic accent recognition model is to reliably estimate the accent or the language class of a speaker from a spoken utterance. A number of studies have been conducted in this topic and a brief review of related research efforts is given below.

Teixeira et al. (1996) investigated a hidden Markov model (HMM) based system to identify English accents from six different European countries; namely, Danish, German, British English, Spanish, Italian and Portuguese and a global identification score of 65.48% was reported suggesting the difficulty of the task. The model was built with a relatively small corpus of about 200 isolated words spoken by speakers from the six different countries.

Fung and Liu (1999) investigated the use of phoneme-class HMMs (stops, affricates, fricatives, nasals, vowels, semi-vowels and glides) to distinguish Cantonese English from

native English. They reported that energy, formant and fundamental frequency are the most discriminative features for identifying Cantonese accent. The use of English syllable structure knowledge to help recognize three accented speaker groups of Australian English – Vietnamese, Lebanese and native speakers – was investigated in (Berkling et al., 1998). The use of this knowledge improved accent identification performance significantly by 6–7% absolute.

It has been shown in (Hansen and Arslan, 1995) that by using multiple acoustic and prosodic features, an accent classification rate of 81.5% could be achieved among four different accent groups of American English – Turkish, Chinese, German and native. The rate increases to 88.9% when the test is limited to a known three word test-set. They constructed and used an accent sensitive database consisting of words and phrases spoken by foreign speakers of American English. They have also noted that accent information is most distinct at phoneme level, and the best features for accent classification are energy, duration, and spectral information.

In (Arslan and Hansen, 1997) it has been shown that the second and third formant frequencies (F2 and F3) are good sources of information for identifying accents and suggested that mel-scale frequency mapping is not particularly suitable for accent recognition. In another work Arslan and Hansen (1996) achieved a 93% accent recognition rate for four accents of American English using a phone-based, isolated word accent recognizer on isolated word strings of 7–8 words. They built and used a database of foreign language accents that consists of words and phrases that are known to be sensitive to accent. This approach requires sufficient amount of training data to build phone models.

In (Huang et al., 2001a) an accent recognition rate of 85% was reported using gender-dependent models to recognize four regional accents of Mandarin using a Gaussian mixture model with 32 components. This approach is essentially text-independent, hence, does not require phonetic labeling.

As can be observed, all of the above studies use accented speech data or a database consisting of accent sensitive phrases spoken by foreign speakers and/or acoustic features such as energy, duration, fundamental and formant frequencies, etc. to build models that recognize accent from a spoken utterance. In this thesis, we investigate the feasibility of building an accent recognizer on a merger of native speech data of the target accent groups. In particular, we take native speech data of English and German from two different application domains, merge them together to form a training set, and build an accent

recognition model that can distinguish if a given English utterance is accented or native. This was motivated by our hypothesis that accent-related information could be effectively captured from native speech data. Furthermore, we investigate various acoustic features to find out those that can detect accent more reliably.

2.7 Spoken Language Understanding

Spoken language understanding (SLU) has been a topic of research since the 1970s (Woods, 1983) and spontaneous spoken language understanding has been of particular interest since the early 1990s when multiple research laboratories participated in the DARPA-funded Air Travel Information System (ATIS) evaluation (Price, 1990). In general, the approaches in the domain of spoken language understanding can be broadly classified as data-driven, rule-based, and a combination of the two.

Data-driven approaches such as those implemented in CHRONUS of AT&T (Pieraccini and Levin, 1993), and Hidden Understanding Model (HUM) of BBN (Miller et al., 1994) estimate model parameters from data by counting the frequencies of transitions between states, word observations while in each state and which states start a sentence. These statistical models are robust and perform well but require a large corpus of fully annotated training examples, which is often not practically available. Another popular statistical approach is the hidden vector state model of Cambridge University (He and Young, 2005) where state transitions between two states are decomposed into separate stack operations that transform one state to the other. A remarkable feature of the hidden vector state model is that it can be trained on "lightly" annotated data and it captures hierarchical structure.

Rule-based systems, on the other hand, such as those implemented in TINA of MIT (Seneff, 1992), PHOENIX of CMU (Ward and Issar, 1996), and GEMINI of SRI (Dowding et al., 1994) use hand-crafted semantic rules to extract meaning from a spoken utterance. Rule-based systems do not require a large amount of semantically annotated data and they perform very well when the structure of the spoken utterance is covered by the grammar (rules). However, rule-based systems, in general, are very expensive to build and maintain since they require extensive manual involvement and expertise. Moreover, they are not robust in the face of unexpected input.

Different combinations of rule-based and statistical approaches have also been investigated. For instance, the generative HMM/CFG (context free grammar) model described in (Wang et al., 2005) integrates a knowledge-based approach into a statistical learning framework.

Statistical spoken language systems differ based on whether they encode hierarchical structure or not. Various statistical approaches that encode hierarchical structure in the domain of SLU have been proposed in (Charniak, 2001; Chelba and Jelinek, 2000; Erdogan et al., 2002; Fine et al., 1998; Miller et al., 1994). All these models require a large amount of annotated training data for parameter estimation while the hidden vector state model (He and Young, 2005) mentioned earlier encodes hierarchal structure and can be built using only an abstract annotation for each utterance.

In this thesis, we describe an approach towards spoken language understanding that requires no semantically annotated training data and encodes hierarchical structure. In this approach, a spoken utterance is conceived as a hidden sequence of semantic concepts expressed in words or phrases. Therefore, the problem of understanding the meaning underlying a spoken utterance in a dialog system can be partly solved by decoding the hidden sequence of semantic concepts from the observed sequence of words. The notable ability of hidden Markov models (HMMs) to estimate the probability of hidden events from observed ones makes them a natural choice for this kind of task.

We propose a model that outputs hierarchically structured semantic information which is suitable for dialog management. The idea we pursue is to capture longer context, resolve ambiguity, and obtain more useful output by a hierarchical organization of low-level semantic concepts into higher-level structures. For instance, low-level concepts like MINUTES, HOUR_OF_DAY, PERIOD_OF_DAY, etc. can be organized to form a high-level concept called TIME which can further be used in a higher-level entity like ARRIVAL_TIME, DEPARTURE_TIME, etc. This kind of structure can be readily produced by a dialog designer of a given application domain using domain knowledge and training examples. We show two different approaches that encode different amount of context, and compare each with the flat-concept model in terms of performance, predictive power, ambiguity resolution ability and information richness of the output.

2.8 Summary

In this chapter we reviewed the state-of-the-art in the field of spoken dialog systems, automatic speech recognition and spoken language understanding. The use of speaker-dependent characteristics such as accent and gender in speech recognition has also been reviewed. We also presented a historical overview and important milestones in the field of spoken dialog systems and automatic speech recognition. The significance of our work within the context of the state-of-the-art in each of these has also been pointed out.

Chapter 3

Tools and Methods

3.1 Introduction

The research objective of this thesis is to deal with robustness issues in speech recognition and spoken language understanding components of a multi-domain telephone-based spoken dialog system. To that end, we set up a framework that should be robust enough to carry out medium-length spoken language interactions with users in multiple application domains in different languages. In this chapter we discuss the various techniques, tools, application programming interfaces, standards and frameworks we use along with a justification of each. Besides, we describe the basic components of a telephone-based spoken dialog system.

Briefly, an interaction with a telephone-based spoken dialog system involves:

- Capturing a spoken utterance from a user through a telephone
- Recognizing the spoken utterance
- Understanding the meaning underlying the recognized utterance
- Performing an action based on the request
- Generating an appropriate response
- Playing the response back to the caller over the telephone.

A simplified architecture of a telephone-based spoken dialog system is depicted in Figure 3.1.

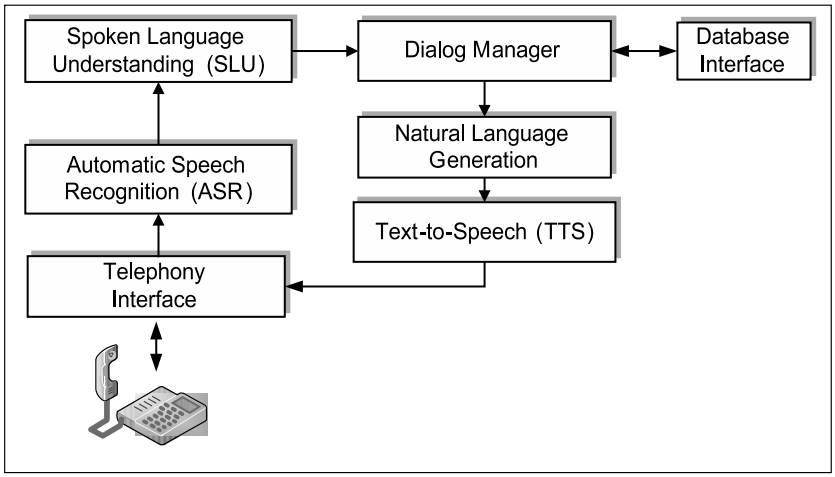

Figure 3.1: A simplified architecture of a telephone-based spoken dialog system

In the sections that follow we describe each of the components shown in Figure 3.1 along with a description of the methods and tools used to build each component.

3.2 Automatic Speech Recognition

The automatic speech recognition component of a spoken dialog system is responsible to determine the orthographic representation of the most likely sequence of words that represent what the speaker might have said from the speech signal. A typical speech recognition system consist of the parts shown in Figure 3.2.

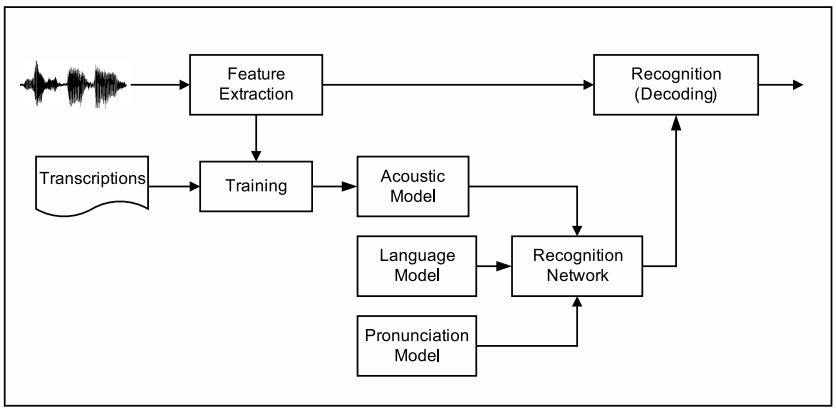

Figure 3.2: Basic architecture of a speech recognition system

3.2.1 Feature Extraction

In speech recognition, feature extraction aims to extract the most salient information from the speech signal that are essential to the recognition of a spoken utterance. It is a very important task because all the other recognition processes depend on the quality of the features extracted (Junqua and Haton, 1995). An ideal set of features for speech recognition are those that are independent of environmental, inter-speaker and intra-speaker variability and yielding similar values for the same acoustic unit regardless of the person who spoke the utterance and the situation in which it was spoken while exhibiting reliable variation between different acoustic units.

The first step in feature extraction is to convert the analog speech signal into digital representation so that digital signal processing methods can be used. This can be done in two steps – sampling and quantization. Sampling reduces the amount of data contained in speech signals without loss of linguistic content by taking representative samples from the continuous signal at a rate that guarantees the reconstruction of the original waveform from the samples. Telephone speech is filtered by the switching network, and only frequencies less that 4 kHz are transmitted over the telephone lines. Thus, according to the well-known Nyquist sampling theorem, a sampling rate of 8 kHz (i.e., 8000 amplitude measurements for each second of speech) is sufficient for telephone speech.

Quantization is a process by which the real-valued amplitude measurements are assigned either 8-bit or 16-bit integer values. The 16-bit representation gives better fidelity of a sampled waveform than the 8-bit representation. However, due to the bandwidth limitation of the telephone channel, telephone speech is often transmitted as 8-bit samples. Therefore, to improve the quality of the transmitted audio, some encoding is performed in which the audio data is first compressed to 8-bit samples, transmitted through the telephone channel, and expanded at the receiving end to 13-bit or 14-bit samples. This is called companding and comes in two variants – A-Law[1] and μ-Law[2]. We then convert the A-Law encoded telephone speech data, which has roughly the precision of 13-bit linear audio to 16-bit linear quantized audio for speech recognition. More on audio transmission over the telephone channel is presented in Section 4.4.2.3.

[1] A-Law is a companding scheme used in European ISDN telephone network

[2] μ-Law is a companding scheme used in the US and Japan

The next step is to transform the digitized speech waveform into a sequence of discrete acoustic feature vectors, each of which represents a short-term speech signal. For the duration covered by a single feature vector, the speech waveform can be considered stationary (Young et al., 2006). The features extracted are generally spectral or cepstral coefficients that condense the information in the speech signal to a vector of real-valued numbers (Gold and Morgan, 2000).

We investigate various acoustic features such as Mel-Frequency Cepstral Coefficients (MFCCs) (Davis and Mermelstein, 1980), Perceptual Linear Prediction (PLP) cepstral coefficients (Hermansky, 1990), Linear Predictive Coding (LPC) (Atal and Hanauer, 1971) features, and LPC-based Cepstral Coefficients (LPCC) (Atal, 1974) for speech recognition, gender recognition and accent detection. More on feature extraction is presented in Section 5.2.1.

3.2.2 Acoustic Model

Acoustic model is a statistical representation of the acoustic realization of the phonemes that make up each word which is influenced by the physical properties of the phonemes and external factors that include environmental, channel, speaker and contextual variabilities. These information are learnt from the feature vectors of the training speech data during a process known as training. Acoustic model is the principal model used in automatic speech recognition to recognize a spoken utterance given the feature vectors corresponding to the unknown utterance.

3.2.2.1 Hidden Markov Model

Speech can be conceived as a hidden sequence of phones observed as a waveform. The task of the required model is, therefore, to estimate the probability of the hidden sequence of phones from the observed signal. It is well-known that hidden Markov model (HMM) is an ideal choice for the task of estimating the probability of hidden events from observed ones. In HMM, speech is modeled as a sequence of hidden states each of which corresponds to a unit of recognition (phoneme, sub-phone, etc.) with transitions between states. Each state can produce a number of observations according to a unique probability distribution, and each distinct observation can be generated at any state. The state

output probability distribution is often modeled as a multivariate Gaussian mixture model (GMM).

We opt for HMMs as they are proven to be powerful enough to cope with the most important sources of speech ambiguity, and flexible enough to allow the realization of recognition systems with dictionaries of tens of thousands of words (Mori and Brugnara, 1997). Hidden Markov models as used in speech recognition are discussed in more detail in Section 5.2.3.

We also use hidden Markov models to build the semantic models we propose in this thesis. In this case, a spoken utterance is conceived as a hidden sequence of semantic concepts expressed in words or phrases. Hence, the goal of the required model is to determine the most likely sequence of the hidden semantic concepts that could have generated the observed sequence of words. As noted earlier HMMs are ideal for this task.

3.2.2.2 Hidden Markov Modeling Toolkit (HTK)

The Hidden Markov Model Toolkit (HTK) – developed and distributed by Cambridge University Engineering Department (CUED) – is an open source, portable toolkit for building and manipulating continuous density Gaussian mixture hidden Markov models. The tools provide facilities for feature extraction, acoustic model training, testing and analysis of results. The various tools in HTK (Young et al., 2006) are used to build the recognition resources that we use in our telephone-based spoken dialog system.

3.2.2.3 Application Toolkit for HTK (ATK)

ATK is a multi-threaded application programming interface designed to facilitate the development of real-time, speech-enabled applications that use HTK-derived resources (Young, 2007). The recognition resources including the acoustic models are prepared off-line using HTK and are provided to ATK as resources in a global configuration file where several HTK compatible acoustic models and other recognition resources can be specified. This makes it suitable for a multi-domain spoken dialog system framework where the necessary recognition resources for various application domains and languages can be built off-line and specified in the configuration file. Besides, ATK allows flexible use of resources during the recognition process.

3.2.3 Language Model

The accuracy of the recognition hypotheses produced by the acoustic model can be further improved by using a language model. A language model consists of prior information about what constitutes a possible word, what words are likely to co-occur and in what sequence (Huang et al., 2001b). The acoustic model might produce several alternative similar words that can be disambiguated by the language model using the encoded prior knowledge. The language model also limits the number sequences that are actually considered during the recognition process.

There are two approaches to language modeling; namely, grammar-based and statistical. In grammar-based approach, one has to specify alternatives via rules. Grammar-based approaches can give good performance but are restrictive in that they dictate the way one can formulate ones utterance. Moreover, since such a grammar can never foresee all the different utterance patterns that people may use in spontaneous speech, they are not appropriate for free, human-to-human like interaction. However, in cases where a user has to choose between a known set of limited words or phrases, the use of finite state grammars could be more reasonable to get better recognition performance.

Statistical language models, on the other hand, provide a probability distribution $P(W)$ over word strings W that reflects how frequently a string of words W occurs as a sentence (Huang et al., 2001b). The probability distribution depends on the amount of training data available. Given enough amount of training data, statistical language models can be more robust to spontaneous speech. However, sufficient amount of training data is often unavailable for each dialog state in a domain-dependent dialog system; hence, a language model built using insufficient data does not capture essential constraints (grammatical or domain-specific) and may not perform as good as grammar-based models.

Associated with the inherent problem of data-sparseness, smoothing techniques are often used in language modeling to assign reasonable probabilities to events that have never been observed in the training data but can occur in a test-set.

In this thesis, we use both bigram language models and hand-crafted finite state grammars to see which of the two perform better in real-time applications. In bigram language models the probability of a word depends on only the preceding word. Since what a user may say in a dialog state can be known in advance using domain knowledge, the use of dialog state-specific language models and/or grammars to utilize task specific issues to

constrain vocabularies at each point in a dialog may be useful to maximize recognition and clarity of intent while allowing a certain degree of freedom. The use of language models allows users to talk to the system in a fairly unconstrained manner. However, since dialog state-specific bigram language models are trained on the transcriptions of subsets of the training data, one can foresee the data insufficiency problem.

3.3 Spoken Language Understanding

The success of a spoken dialog system depends not only on the correct recognition of a spoken utterance but also on the correct comprehension of the intention underlying the spoken utterance. Automatic speech recognition systems commonly output the most probable transcription of a spoken utterance or a list of N most probable word sequences and need not perform syntactic or semantic analysis on the recognized input. Therefore, in a spoken dialog system there is often a separate unit that is responsible for inferring what is meant from what is said.

In this thesis, we develop a new, robust hierarchical HMM-based semantic concept labeling model that essentially enriches the raw text output of the speech recognizer with semantic information that can be used to infer the meaning of a given utterance in a given dialog state. The model is trained on semantically unlabeled data and offers a number of features in terms of performance, ambiguity resolution ability and expressive power of the output as discussed in Chapter 6.

3.4 Dialog Management

The choice of an open VoiceXML framework is a key design decision in developing a telephone-based spoken dialog system based on VoiceXML. We have chosen ©OptimTalk (OptimSys, 2006) VoiceXML platform that consists of a VoiceXML interpreter, a CCXML interpreter, and other abstract interfaces which allow us to build and integrate our own ASR engine, telephony interface, grammar component, semantic interpreter, TTS system, etc. VoiceXML 2.0 (W3C, 2004) is used to author the dialogs and CCXML 1.0 (W3C, 2007) is used to write the call handling policy.

3.4.1 VoiceXML

Voice eXtensible Markup Language (VoiceXML) is the World Wide Web Consortium's (W3C's) standard that specifies how a dialog between a caller and a speech-enabled application is constructed and executed. VoiceXML is an attempt to give developers the tools they need to express a conversational interface using existing web technologies and Internet standards (Sharma and Kunins, 2002). VoiceXML is convenient for dialog authoring and provides features to support complex dialogs. Moreover, it minimizes client/server interactions by specifying multiple interactions per document and separates user interaction code from service logic.

3.4.2 JavaScript

ECMAScript (ECMA-262) is the scripting language that provides client-side scripting capabilities to VoiceXML. We use JavaScript, which is an implementation of the ECMAScript standard, to write a program that essentially parses and validates the semantically enriched recognized utterance to extract meaning-bearing words or phrases to fill one or multiple dialog slots and to update the state of the dialog accordingly. We provide the user some degree of control over the conversation by allowing the user to respond more flexibly to the system's prompts; i.e., a user can provide more than one piece of information at a time to minimize the number of interactions required to complete a task.

3.4.3 VoiceXML Interpreter

A VoiceXML interpreter is a piece of software that reads and processes VoiceXML documents as described by the VoiceXML language standard (Edgar, 2001). Essentially, the core of a VoiceXML interpreter implements the Form Interpretation Algorithm (FIA) which specifies the procedure for walking through the various fields of a form to drive the interaction between the user and a VoiceXML document. The VoiceXML interpreter in ©OptimTalk is more than just an interpreter in that it loads the relevant dialog document from the Web server hosting the VoiceXML documents, and executes the dialog by calling appropriate methods of the various components of the system so as to play prompts, accept user input, and pass them on to a speech recognition engine, determine what to do

next according to the instructions in the active VoiceXML script. In short, it serves as the dialog manager of the system as shown in Figure 3.3.

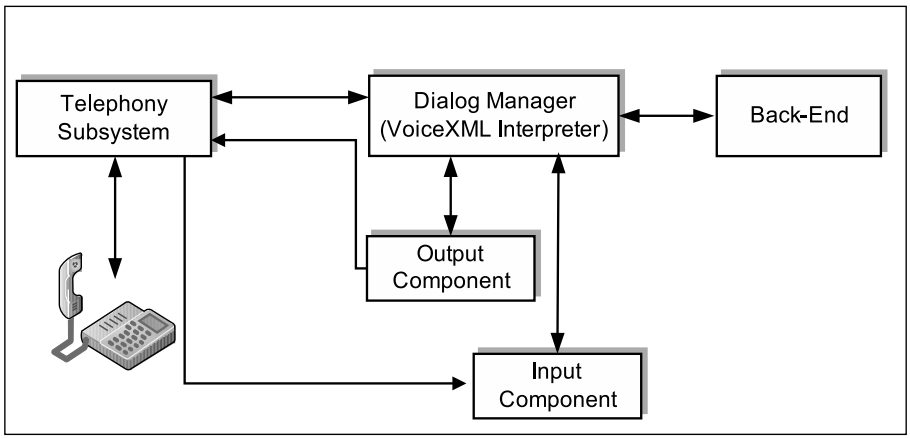

Figure 3.3: VoiceXML interpreter as a dialog manager

As can be seen, the VoiceXML interpreter orchestrates the whole interaction by activating the various components of the system as appropriate. A more comprehensive description of the system as a whole is given in Section 4.2.

3.5 Telephony Interface

The telephony subsystem is the interface between the external telephone network and the application. It consists of a telephony interface component and a call control component.

The telephony interface component makes any telephony hardware accessible to the CCXML interpreter through a unified interface. The rules for accepting and processing incoming calls are described in a CCXML document. The CCXML interpreter executes the commands in the CCXML document by calling the relevant methods of the telephony interface component for establishing a connection, answering the call, streaming audio to the telephone and capturing the spoken input from the user, etc. The telephony interface component is implemented using the application programming interface standard to access ISDN services – Common ISDN Application Programming Interface (CAPI)[1].

[1] http://www.capi.org/pages/home.php; last accessed February 27, 2009

The uniform resource identifier (URI) to the initial dialog script of the application is also specified in the CCXML document and the CCXML interpreter instructs the dialog manager to fetch and execute this document when a call is accepted.

3.6 Speech Output

The output component wraps a text-to-speech (TTS) synthesis engine that synthesizes prompts and responses to be played back to the caller over the telephone. We use ©Loquendo TTS engine in our system to play prompts and responses to the users over the telephone in English and German.

3.7 Database Interface

When an automated telephony system is based on a voice browser, most of the application logic resides on a web server (Edgar, 2001) and the data reside on a database server. MySQL database Server is used to build and store the database(s). The system is based on dynamic content where PHP server-side scripting language is used to dynamically generate VoiceXML documents containing data stored in the database. PHP and MySQL are both open source and form a very good combination for creating data-driven applications. In order to process HTTP requests from the client and serve VoiceXML documents Apache Web server is used. Apache is also free and works well with PHP and MySQL.

3.8 Evaluation Method

A vital and final step in spoken language interaction system development is to evaluate the usability and quality of the system. The quality and usability of a spoken dialog system is strongly related to user satisfaction which can only be obtained from subjective judgements collected from test users in a quantifiable form with questionnaires. We use questionnaires based on SASSI (Subjective Assessment of Speech System Interfaces) (Hone and Graham, 2001) and the recommendation of the International Telecommunication Union (ITU-T) (ITU_T Rec. P.851). We also extract complementary information about the performance of the various components of the system from logged interactions.

3.9 Summary

In this chapter, we described the components of a telephone-based spoken dialog system in general and presented the various components of our system in particular. Besides, we introduced the tools, techniques, standards, and APIs we use to realize the envisaged robust multi-domain, multilingual spoken dialog system. The modeling approaches we use to build the required acoustic, language and semantic models have also been described. In Chapters 5, 6, and 7 we present more detailed descriptions on speech recognition, gender identification, accent recognition, spoken language understanding, and spoken langauge interaction issues. The experiments conducted and the results obtained will be discussed in more detail in Chapter 8.

Chapter 4

System Description

4.1 Introduction

In this chapter, we present a fairly comprehensive description of the telephone-based spoken dialog system framework developed as part of this thesis in general, and the development of the various components in particular. In Section 4.2, we present a high-level description of the system and its component parts as well as a typical usage scenario that shows how the various components work together in a dialog session. Section 4.3 presents a short description of how VoiceXML and CCXML standards complement each other and work together. In Section 4.4, we describe the implementation of the telephony interface component in sufficient detail. Section 4.5 describes the implementation and integration of the input component consisting of an automatic speech recognition engine, a gender recognizer, a grammar component and a semantic interpreter. Finally, a summary of the chapter is presented in Section 4.6.

4.2 Components of the System

A telephone-based spoken dialog system generally consists of an interface to a telephone network, an automatic speech recognition engine, a spoken language understanding component, a mechanism for response generation, an audio output module and a dialog manager. Similarly, our telephone-based spoken dialog system framework comprises of:

1. A telephony interface component to deliver and process calls

2. An input component that consists of:
   (a) A media source component to capture audio stream from the telephony interface
   (b) An automatic speech recognition engine to recognize what is said
   (c) A gender-recognizer to identify the gender of a user from a spoken utterance
   (d) A grammar unit to prepare the grammar or the language model used by the speech recognizer
   (e) A semantic interpreter to enrich the output of the speech recognizer with semantic information so that the meaning underlying a spoken utterance can be easily extracted
3. An output component to synthesize the prompts and responses to be played back to the user
4. A back-end where the application logic, the database and the dialog scripts reside
5. A dialog manager that orchestrates the various components.

The telephony interface component controls the ISDN telephony card and is responsible for, among other things, capturing audio stream from the user and playing audio prompts to the user over the telephone. It is implemented using CAPI (Common ISDN Application Programming Interface) which enables application developers to access ISDN services without having to deal with the low-level ISDN details.

The core of the input component consists of an ATK-based speech recognizer which also supports automatic gender recognition. The recognition resources; namely, the acoustic models, the language models and the pronunciation dictionaries are prepared off-line using HTK (Young et al., 2006). The grammar component reads each pre-compiled grammar or language model file that is specified in each VoiceXML document and makes it available to the ASR engine at runtime. The semantic interpreter enriches the recognized utterance with semantic information to easily infer what is meant from what is said.

A high-level architecture of the system is depicted in Figure 4.1. At the core of the system is ©OptimTalk (OptimSys, 2006) – a VoiceXML framework that consists of a VoiceXML interpreter, a CCXML interpreter, and other abstract interfaces that allow us to

build and integrate our own components. The VoiceXML interpreter in OptimTalk serves as the dialog manager of the system as will be described in Section 4.2.1. In the following sections, the terms dialog manager and VoiceXML interpreter are used interchangeably.

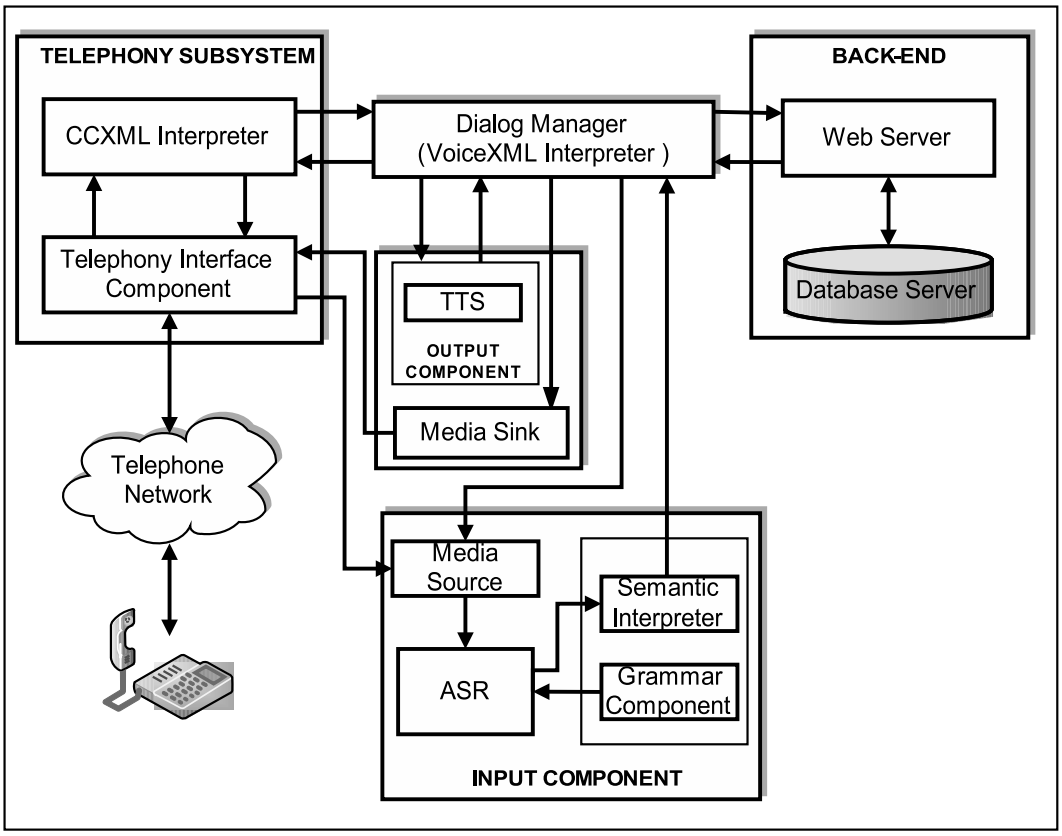

Figure 4.1: High-level block diagram of the system

4.2.1 Typical Usage Scenario

The interaction of the various components of the system to carry out a telephone-based spoken dialog can be described as follows:

1. A user calls the system and the telephony interface component receives the call.
2. The initial CCXML document is loaded.

3. The CCXML interpreter executes the commands in the document using the methods implemented in the telephony interface component. If the call is accepted, the CCXML interpreter instructs the VoiceXML interpreter to load the dialog script specified in the CCXML document.

4. The VoiceXML interpreter sends an HTTP request to the Web Server and the Web Server delivers the requested document. From this point on, the VoiceXML interpreter takes control of the interaction.

5. The VoiceXML interpreter executes the commands in the VoiceXML document. During the interpretation, the VoiceXML interpreter calls the methods of the various components to make the interaction possible as follows:

   - To play prompts, the dialog manager calls the methods of the output component to process prompts, perform text-to-speech (TTS) synthesis and store the synthesized data in a so-called audio container that is taken by the output manager (part of the dialog manager). The output manager then sends the audio container to the media sink component which streams the audio to the telephony interface component. In the telephony interface component, we process the streamed audio as appropriate and play it over the telephone.
   - When user input is expected, the input component uses an interface of the media source component provided by the VoiceXML interpreter to start capturing audio input. The media source component also provides the audio stream received from the telephony interface component to the speech recognizer.
   - When a spoken input is recognized, the recognition output is sent to the semantic interpreter component that is responsible to enrich the raw text output of the speech recognizer with semantic information. These semantic information are used to extract the meaning of the recognized utterance at a given dialog state.

6. When all the required information in a dialog are obtained, the scripts on the web server process the submitted parameters, perform some database operations and generate a new VoiceXML document. This is then sent back to the VoiceXML interpreter for interpretation.

7. When the dialog is finished the VoiceXML interpreter informs the call control component about the fact and the call is properly terminated.

8. The system continues running waiting for the next incoming call.

4.3 VoiceXML and CCXML Working Together

When VoiceXML and CCXML are used together, CCXML provides call control functions such as handling incoming calls, placing outgoing calls, bridging multiple call legs, disconnecting calls, etc. while VoiceXML is used as a dialog environment. CCXML does not provide any mechanism for interacting with callers directly but whenever interaction with a caller is required a CCXML session can initiate a separate dialog session provided by VoiceXML and connects the call to the VoiceXML dialog. When the VoiceXML dialog completes successfully or fails, the VoiceXML interpreter notifies the CCXML session about the fact and the CCXML interpreter then terminates the call.

In ©OptimTalk, the framework we use, the interconnection between the dialog manager and the telephony subsystem is provided by a dialog-to-telephony bridge facility. This bridge translates the commands from the form produced by the telephony component to the form understood by the dialog manager or vice versa (OptimSys, 2006).

A CCXML interpreter initiates a dialog using the <dialogstart> element. Execution of this element connects a dialog environment to a connection and instructs it to start interacting with the caller. For some dialog environments it may take some time to initialize the dialog environment and hence CCXML provides an option to prepare a dialog prior to starting it using the <dialogprepare> element.

4.4 The Telephony Interface Component

As its name suggests, the telephony interface component provides an interface between the system and a telephone network. The system is based on Integrated Services Digital Network (ISDN) and the interface is implemented using CAPI (Common ISDN Application Programming Interface). In the following sections we provide a brief overview of ISDN, introduce CAPI and discuss the implementation of the telephony interface component.

4.4.1 Overview of Integrated Services Digital Network (ISDN)

Integrated Services Digital Network (ISDN) is a system of digital phone connections that allows fast and reliable transfer of information in many different formats over the existing telephone infrastructure. The distinguishing features of ISDN are integrated services (data, voice, images, and video), improved transmission rate, and better transmission quality as a result of end-to-end digital transmission of data.

With ISDN, signaling information and data are conveyed through different channels. All signaling data (for call establishment and release) are transmitted through a channel called D-channel (Delta channel) while data and voice are transmitted through channels called B-channels (Bearer channels).

There are two basic levels of ISDN service: Basic Rate Interface (BRI) and Primary Rate Interface (PRI). Basic Rate Interface is intended for home and small enterprizes and consists of two bearer channels (each 64 kb/s) plus one delta channel (16 kb/s) (2B+D) for a total of 144 kb/s. For users with greater capacity requirements, the Primary Rate Interface provides a channel structure which is typically 23 B channels plus one 64 kb/s D-channel (23B+D) in USA and Japan; in Europe, Australia and other parts of the world, PRI consists of 30 B channels plus one 64 kb/s D-channel (30B+D).

An incoming ISDN line is terminated at the customer premises by a network termination device known as NT1. The network termination device has a 2-wire interface called U-interface on the network side of the device and a 4-wire interface called S0 (also known as S/T) interface on the terminal side. The purpose of the network termination device is to convert the 2-wire U-interface signal to the form recognized by the S0 interface.

4.4.2 Common ISDN Application Programming Interface (CAPI)

Common ISDN Application Programming Interface (CAPI) is a programming interface that enables ISDN application developers to develop applications that use ISDN hardware without having to deal with the low-level ISDN details. It provides a uniform, independent and easy to use interface for applications and offers a unified access to ISDN hardware components.

Under Windows operating system the CAPI services are provided via a Dynamic Link Library (DLL) known as "capi2032.dll" for 32-bit Windows-based applications and is usually included with most ISDN adapters. An application communicates to CAPI via

this library in order to use an ISDN card. The interface between applications and CAPI consists of a set of CAPI functions. In order to understand the telephony interface component, a brief overview of CAPI functions is presented in the following subsections. The main reference material for this section is Part I of the CAPI documentation[1].

4.4.2.1 CAPI Functions

Before an application can attempt to use any CAPI service, it must check whether CAPI is installed and is operational on the machine. The function CAPI_INSTALLED can be used for this purpose. Prior to any communication between an application and CAPI, the application must register with CAPI using the CAPI_REGISTER function. As the application registers, CAPI assigns a unique application ID (ApplID) to the application and sets up a message queue that the application uses to communicate with CAPI.

Communication between an application and CAPI is via messages. A message is a piece of information that is exchanged between a registered application and CAPI. A message going from an application to CAPI is known as a REQUEST and the corresponding answer from CAPI is known as a CONFIRMATION. A messages initiated by CAPI is known as an INDICATION and the corresponding acknowledgement is known as a RESPONSE. As can be observed, each REQUEST has a matching CONFIRMATION, and each INDICATION must have a corresponding RESPONSE. Every message name ends with a suffix (_REQ, _CONF, _IND, _RESP) to reflect the message type.

Messages are communicated via message queues and are processed in the order of their arrival. There is exactly one message queue for CAPI to accept messages from an application and one for each registered application to receive messages from CAPI. The application transfers its message by calling the CAPI_PUT_MESSAGE function and reads new messages from its own queue using the function CAPI_GET_MESSAGE.

If a registered application wants to terminate its connection to CAPI, the function CAPI_RELEASE is used. When an application is released, the previously used message queues are freed. An application must disconnect all existing connections before issuing a CAPI_RELEASE.

[1] http://www.capi.org/download/capi20-1.pdf; last accessed February 27, 2009

4.4.2.2 Establishing a Connection

Establishing a connection involves creating a physical link and a logical link. Table 4.1 shows the message exchanges that take place between the application and CAPI to create a physical connection.

Table 4.1: Establishing a physical connection

Message	Description
LISTEN_REQ	The application sends LISTEN_REQ to be informed when a call comes in.
LISTEN_CONF	CAPI confirms – CAPI is ready to inform the application when a call indication comes in.
CONNECT_IND	With an incoming call, the application receives a CONNECT_IND containing a PLCI number that identifies the physical connection.
CONNECT_RESP	The application acknowledges.
CONNECT_ACTIVE_IND	CAPI sends CONNECT_ACTIVE_IND to indicate that the call was connected.
CONNECT_ACTIVE_RESP	The application acknowledges – physical connection is established.

To create a logical link, the message exchanges that take place between the application and CAPI are shown in Table 4.2.

Table 4.2: Establishing a logical connection

Message	Description
CONNECT_B3_IND	CAPI indicates that a logical connection is coming. The message contains an NCCI number identifying the logical connection.
CONNECT_B3_RESP	The application acknowledges.
CONNECT_B3_ACTIVE_IND	CAPI indicates that a logical connection of the B channel is established.
CONNECT_B3_ACTIVE_RESP	The application acknowledges – logical connection is established.

Once both the physical and the logical links are set up, the application can send DATA_B3_REQ messages and receive DATA_B3_IND messages to send and receive audio data within the logical connection as described in the next section.

4.4.2.3 Audio Transmission

In every connection, there are two streams of audio data – the stream directed to a phone line and the stream coming from a phone line. The specific implementation of sending and receiving audio data to and from a telephone line depends on the telephone interface card used, which in our case is an ISDN card based on CAPI.

In this case, to send audio data to CAPI, the application sends DATA_B3_REQ messages and CAPI confirms with DATA_B3_CONF. The audio data is not contained in the message instead a 32-bit pointer is used to convey the address of the data area.

To avoid the inherent delay that may occur if each message had to be confirmed before receiving the next one, CAPI allows up to seven unconfirmed DATA_B3_REQ messages which will be confirmed later in the order of their arrival.

Figure 4.2 shows the process of transmitting audio data to CAPI.

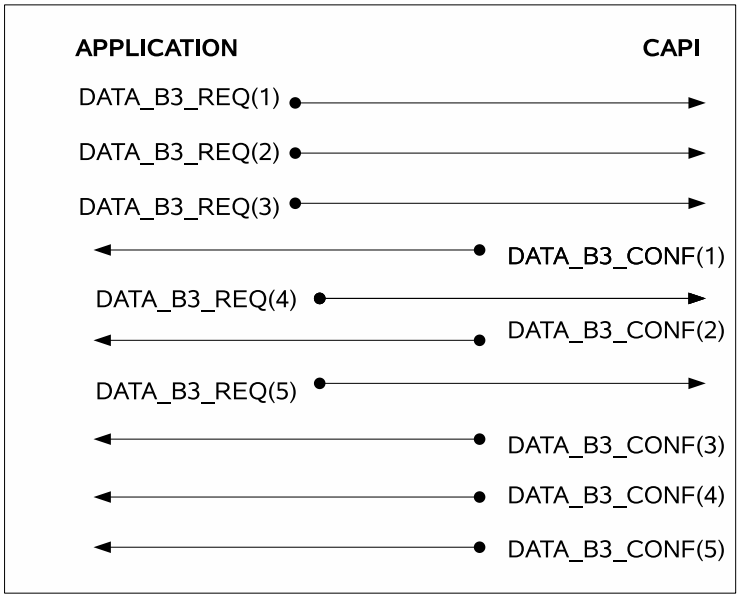

Figure 4.2: Transmitting audio data to CAPI: CAPI allows up to seven consecutive unconfirmed DATA_B3_REQ messages which will later be confirmed in the order of their arrival.

Before sending the audio data synthesized by a TTS engine to CAPI, it is necessary to convert the data to an appropriate format (A-Law format, in our case). µ-Law and A-Law are encoding schemes to encode 14-bit and 13-bit of sampled data in 8-bit logarithmic representation using the G.711 companding scheme. Companding is a scheme used to reduce bandwidth requirements for transmitting audio data over the telephone channel, where information is compressed at the sending end, transmitted through the telephone channel, and expanded at the receiving end. µ-Law is the standard used in the United States and Japan while A-Law is the European standard. Moreover, as CAPI sends and accepts each octet of A-Law or µ-Law data in a reversed bit order, it is necessary to reverse the bits in each byte before streaming the audio data to CAPI.

In the opposite direction, CAPI signals all incoming data from the phone line with DATA_B3_IND and the application acknowledges with DATA_B3_RESP. The data is not contained in the message instead a 32-bit pointer is used to communicate the address of the data area.

Figure 4.3 shows the process of receiving audio data from CAPI.

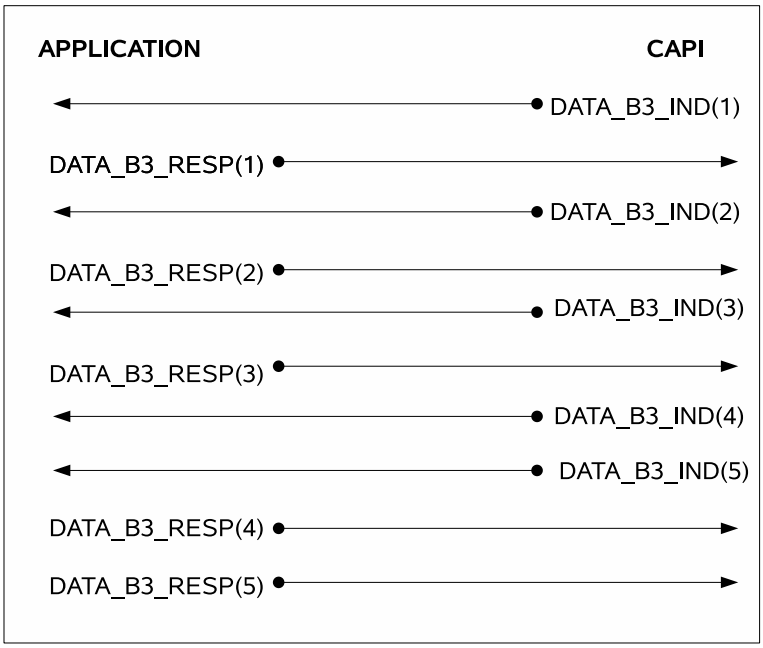

Figure 4.3: Receiving audio data from CAPI: For high data throughput, applications should respond to DATA_B3_IND messages promptly

Before we send the received audio data from the phone line to the automatic speech recognition system, we convert the audio data from A-Law to 16-bit linear PCM format and reverse the bits in each byte of audio data.

4.4.3 The CAPI-based Telephony Interface

The telephony interface component is implemented as a separate dynamic link library and is specified in a configuration file. The ©OptimTalk core executable loads it at runtime.

The core of the telephony interface component is implemented as a Finite State Machine (FSM). A finite state machine consists of a set of states, a set of possible input events and a function that determines the transition from one state to another for a given input event. The process begins at a start state and an input event which moves the state machine to the next state based on the transition function.

In this implementation, the FSM is implemented as a two dimensional array where one dimension corresponds to the states and the other dimension specifies the input event to be handled. Each array element consists a value that specifies the new state the machine moves to and an action to execute.

At the heart of the system is a function that can be called in a loop to continually see what CAPI messages are coming using the GET_MESSAGE function. As described in Section 4.4.2.1, the GET_MESSAGE function gives two types of messages – INDICATIONs and CONFIRMATIONs. The message in the queue is continually read in a loop and when CAPI_GET_MESSAGE returns an event, the function that processes incoming messages is called with the message as a parameter. If the message is an INDICATION, the function translates the message into an input event for the finite state machine and calls the transition function with the input event and the message. If the message is a CONFIRMATION the corresponding request gets confirmed. This is done repeatedly until the FSM sets the "finished" flag which signals disconnection.

The state diagram in Figure 4.4 shows the states and the inputs (telephony events) considered in the system.

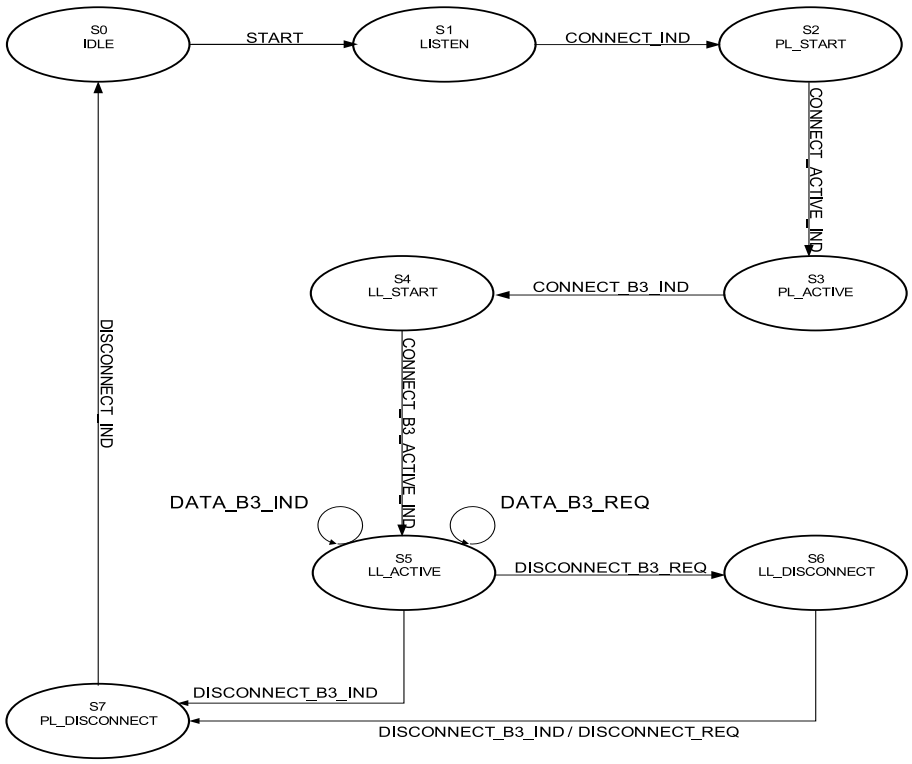

Figure 4.4: The finite state diagram

As can be seen in Figure 4.4, the state machine is fired up with the initial *START* event which moves the state machine to the *LISTEN* state (i.e. the application issues LISTEN_REQ and CAPI confirms with LISTEN_CONF). At the *LISTEN* state CAPI is ready to inform the application when a call indication comes in.

When a new call comes in, the application receives a CONNECT_IND which causes a transition to *PL_START* state where the establishment of a physical link starts. The application acknowledges the CONNECT_IND with CONNECT_RESP. Then the application receives a CONNECT_ACTIVE_IND message which should be acknowledged with the corresponding CONNECT_ACTIVE_RESP message. The FSM then transits to state *PL_ACTIVE* where the physical link is fully established.

Once the physical connection is established, a logical connection over the B channel needs to be set up as described in Section 4.4.2.1. Hence, the application receives a CONNECT_B3_IND message which the application should acknowledge with CONNECT_B3_RESP message to accept the logical connection. At this point, the FSM tran-

sits to state *LL_START* where the establishment of a logical link starts. Sooner or later, a CONNECT_B3_ACTIVE_IND message comes and the application acknowledges it with the corresponding CONNECT_B3_ACTIVE_RESP message. Consequently the FSM transits to state *LL_ACTIVE* where the logical link is fully established. At this state, the connection is fully established and hence it is possible to send and receive audio data to and from CAPI as described in Section 4.4.2.3.

The last two states, *LL_DISCONNECT* and *PL_DISCONNECT* are used for handling disconnection or hang up.

When the call is completed, we de-initialize the connection and call the method that fires up the FSM once again. When the FSM is fired, it transits to the *LISTEN* state listening for the next incoming call. This keeps the CCXML interpreter running all the time waiting for incoming calls.

4.5 The Input Component

The input component wraps the automatic speech recognition engine and is implemented as a separate dynamic link library. The grammar component is also implemented as a separate library and is passed to the input component at runtime. Once a spoken input is recognized, the recognition result is sent to the semantic interpreter which is implemented as part of the grammar component. The input component is specified in a configuration file and the ©OptimTalk core executable loads it at runtime. The grammar component and the semantic interpreter are used along with the speech recognizer and hence are considered as parts of the input component.

In the sections that follow, we describe the ATK-based automatic speech recognition engine, the gender recognizer, the grammar component and the semantic interpreter. We also give a helicopter view of the entire recognition process in Section 4.5.5.

4.5.1 The ATK-based Speech Recognizer

ATK is a multi-threaded API designed to facilitate building real-time applications that use HTK-derived recognition resources (Young, 2007). The core of ATK is based on three fundamental objects – packets, buffers and components. Packets are used for transmitting a variety of information between asynchronously executing threads (components)

while buffers provide the channel for passing packets from one thread to another. The information in a packet could be speech waveforms, feature vectors, or recognized phrases. A component in ATK is a task with its own thread, and it communicates with other threads by passing packets of information via buffers. The block diagram in Figure 4.5 depicts the components of the ATK-based speech recognition system integrated in the framework.

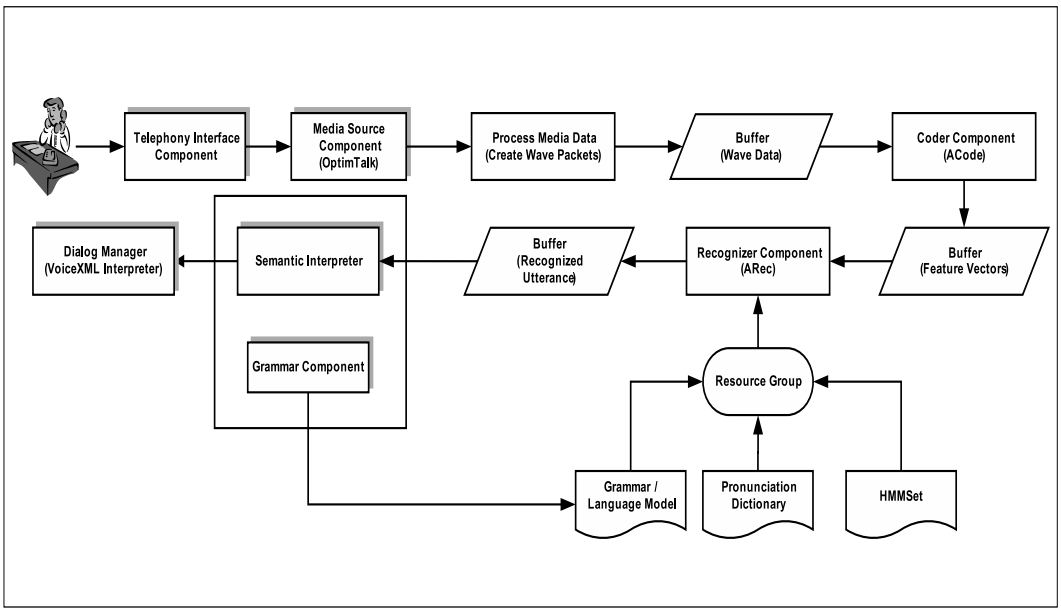

Figure 4.5: Block diagram of the ATK-based speech recognizer

In ATK there are three main components; namely, an audio source (ASource) component, a coder (ACode) component and a recognizer (ARec).

The primary function of the audio source component (ASource) is to capture input speech (Young, 2007). As can be seen in Figure 4.5, we do not use ATK's ASource component directly. Instead we use the media source implementation of ©OptimTalk. The spoken input is captured and processed in the telephony interface component and is streamed to the input component through an interface that is used to receive audio data. The media source component calls the methods of this interface repeatedly to stream audio data to the speech recognizer. Then we organize the received audio stream into wave packets and put the packets to the buffer which connects the audio source to the coder component.

The coder component (ACode) then gathers the incoming wave packets and passes them to the HTK module HParm that converts the observed acoustic signals to a series of feature vectors using the signal processing operations defined in HsigP module (Young et al., 2006). The extracted feature vectors are then placed in the buffer that connects the coder and the recognizer.

The task of the recognizer (ARec) is to generate a hypothesis or a lattice of hypotheses for the underlying speech unit sequence from the sequence of observation vectors corresponding to the unknown utterance. It depends on various resource objects such as HMMSet (AHmms), pronunciation dictionary (ADict), grammar (AGram) and optionally an n-gram language model.

An HMMSet defines the actual hidden Markov model for each linguistic unit which is always initialized from one or more external file(s) specified in a global configuration file. In other words, the HMMSet defines the acoustic model. A fairly detailed description on acoustic modeling is presented in Chapter 5. The inventory of the basic linguistic units (e.g., phonemes, triphones) for which distinct HMMs are built is stored in a file and should be specified in the configuration file as well. A pronunciation dictionary object defines the phonemes that form each word in the vocabulary of the recognizer and it is usually initialized from an external file. A grammar or a language model object defines a network of allowable word sequences. It can be loaded from an external file or created on the fly.

These recognition resources are stored in a resource manager (ARMan) organized into logical resource groups. At any one time, the recognizer is using the resources in a specific resource group. If the resource group is changed or if any member of the group is modified, a new resource group is recompiled for use by the recognizer for the next recognition task. A recognizer is instantiated with a name and pointers to an input buffer, an output buffer and a resource manager.

4.5.2 The Gender Recognizer

As discussed in Section 2.4, due to similar articulatory mechanisms there is considerable acoustic similarity of voice within speakers of the same gender while there exists apparent difference between the voice of male and female speakers. This suggests that using gender-dependent acoustic models can give better recognition performance than gender-independent acoustic models.

To be able to use a gender-dependent acoustic model at runtime, it should be possible to reliably estimate the gender of the user from a spoken utterance. Our gender recognizer is based on a Gaussian mixture model which will be described in Section 5.3.2.

At the start of a dialog, there are two recognizers running in parallel – the first one is used to recognize what is actually said and the second one estimates the gender of the speaker from the same acoustic input. The extract in Listing 1 shows the first part of a typical interaction where the gender recognizer is used.

Listing 1 Dialog extract

```
System: Hello! My name is KEY. I provide service in English and
        German. Which one do you prefer?
User:   German
System: Willkommen zum automatischen Bahnauskunftssystem der
        Universität Magdeburg! Bitte nennen Sie Ihren Reiseplan!
User:   ...
```

The gender of the speaker is estimated at the same time when the preferred language (i.e. "German" in this example) is recognized from the first utterance. Accordingly, the gender-dependent model corresponding to the estimated gender and the preferred language is loaded. At the same time the semantic model and dialog scripts corresponding to the preferred language or application domain are loaded. As a result, the rest of the dialog proceeds in the chosen language with a gender-dependent acoustic model and a domain-specific semantic model.

4.5.3 The Grammar Component

The Speech Recognition Grammar Specification (SRGS) format of W3C is a standard way to specify speech recognition grammars in VoiceXML-based applications. However, HTK-based speech recognizers require grammar files in HTK's Standard Lattice Format (SLF) and do not recognize SRGS. Therefore, we developed a separate grammar component to enable the use of grammar in the required standard lattice format within the VoiceXML framework.

In fact, an alternative approach would have been to convert a grammar written in SRGS format into a form that can be used by an HTK-based speech recognizer at run-time. However, this approach has not been pursued for two reasons. First, an SRGS-based grammar is written along with the semantic instructions according to the Semantic Interpretation for Speech Recognition (SISR) specification. Apparently, this involves a considerable amount of human effort in writing complex grammars along with semantic instructions. We would like to keep the grammar writing easy and automate the semantic tagging part. Second, we also intend to use dialog state-specific bigram language models as alternatives to grammars which makes the latter approach inconvenient.

The grammar component essentially gets the URI address of each grammar file or language model specified in the VoiceXML script and re-writes the content to a predefined location temporarily on the disk so that it can be used in the next recognition task. If the recognized utterance matches the active grammar at a given dialog state, the recognition output is sent to the semantic interpreter as shown in Figure 4.5. If a match is not found, then the application informs the user that no match is found and prompts for a matching input.

4.5.4 The Semantic Interpreter

Since we do not use SRGS for the reasons described in the previous section, we cannot use its companion Semantic Interpretation for Speech Recognition specification which defines the syntax and semantics of using semantic instructions in SRGS. Instead, we built our own statistical semantic interpreter which automatically adds semantic and hierarchical information to the recognized utterance. This is, in fact, one of the key issues we want to address in this thesis – to introduce an efficient and powerful way to semantic interpretation with a number of virtues as will be described in Chapter 6.

The semantic model for each application domain is trained off-line as described in Chapter 6 and is specified in the application. After the preferred language is recognized from the first spoken utterance, the corresponding semantic model is loaded for every recognized utterance matching the active grammar. The task of the semantic interpreter is to semantically enrich the output of the speech recognizer to enable easy extraction of the meaning underlying the recognized utterance in a given dialog state. The core of the semantic extraction algorithm is depicted in Listing 2.

Listing 2 The semantic tagging algorithm

```
hmm.loadProbs(semantic_model) // load the semantic model corresponding
                               // to the preferred language
hmm.readString(utterance)     //read the recognized utterance
for all words:
    hmm.addObservation(word)
path, joint_prob = hmm.viterbi()
obs_prob = hmm.obsProb()
for all (state, observation) in path:
     if non_emitting(state):
        if isEntry(state):
           print "("
        else if isExit(state):
            print ")", state
     else if emitting(state):
            print observation, state
hmm.reset()
```

The trained semantic model consists of two text files – one consisting of the transition probabilities and another consisting of emission probabilities. In Listing 2, the function "hmm.loadProbs()" loads these model files for a specified application domain. The function "hmm.readString()" reads the recognition output of the speech recognizer. Every word in the utterance is checked if it is in the lexicon of the tagger, otherwise it is marked as "oov". The function "hmm.addObservation()" constructs a trellis of state transitions. The hmm.Viterbi() implements the Viterbi algorithm to search the most likely sequence of states through the trellis. The function hmm.reset() resets the trellis and prepares the HMM for the next utterance.

4.5.5 Summary of Recognition Events

In order to give a helicopter view of the events in the input component, we provide a summary of the main events in a dialog session as follows:

1. The input component is initialized; i.e.:

- The VoiceXML interpreter reads the name of the input component from the global configuration file, creates an instance of the input component and initializes it.
- Initialize ATK which includes initializing the underlying HTK libraries.

2. When user input is expected, the VoiceXML interpreter calls the method of the input component that is responsible to start voice input collection.
3. The input component starts capturing audio data using an interface of the media source component.
4. The media source component calls the methods of the the input component that are responsible to receive audio data.
5. The received audio data are organized into wave packets and are streamed to the buffer that connects the audio source to the coder.
6. The coder reads the buffer, extracts the required features and makes the extracted feature vectors available for the speech recognizer.
7. The grammar component prepares the grammar specific to a given dialog state and makes it available to the speech recognizer.
8. The recognizer hypothesizes the most likely utterance from the sequence of feature vectors using a given set of recognition resources. The recognition result is then made available to the application through the output buffer of the recognizer.
9. When the recognition of an utterance is finished, the VoiceXML interpreter is informed about the fact and the recognizer is temporarily stopped.
10. The output of the recognizer is then passed to the semantic interpreter that enriches the raw text output with semantic information.
11. For the next run, the recognition resources are updated with a new dialog-specific language model or grammar.
12. When the next user input is expected, the recognizer is restarted with an updated resource group.

13. Steps 2-12 are repeated until the dialog is finished or terminated.

4.6 Summary

In this chapter we presented a comprehensive description of the system and the various components that make up our telephone-based spoken dialog system framework. The implementations of the components we have developed in this thesis; namely, the telephony interface component, and the input component that consists of a speech recognizer, a gender recognizer, a grammar component and a semantic interpreter are described. For the sake of clarity, we left out some low-level details. Now that we have described the testbed, the various models that make up a robust spoken dialog system will be discussed in the following chapters.

Chapter 5

Automatic Speech Recognition and Related Issues

5.1 Introduction

In this chapter we present the fundamentals of automatic speech recognition, gender identification and accent detection along with a description of the approaches we use to utilize speaker-dependent characteristics such as gender and accent to improve speech recognition performance. In Section 5.2, we describe speech recognition as a Bayesian inference problem and we present an overview of hidden Markov model as used in automatic speech recognition. As introduced in Section 3.2, speech recognition consists of feature extraction, acoustic model training, language modeling and decoding. Hence, we present a fairly detailed overview of these processes. Section 5.3 discusses the use of user-group dependent acoustic models based on gender and accent to improve speech recognition performance in a spoken dialog system. We further describe our Gaussian Mixture Model (GMM) based gender and accent recognition models. Moreover, the accent recognition approach we propose in this thesis; namely, using native speech data of two or more target accent groups to train an accent recognizer is described. In Section 5.4, we present a brief overview of Maximum Likelihood Linear Regression (MLLR) and Maximum a Posteriori (MAP) speaker adaptation techniques as they will be used for accent and channel adaptation. Finally, we summarize the chapter in Section 5.5.

5.2 Automatic Speech Recognition

Automatic speech recognition can be broadly defined as a process of transforming a speech signal into a string of words. As noted earlier, an utterance can be conceived as a hidden sequence of phones that are mentally formulated into words. The hidden sequence is observed as a speech signal. The goal of the required model is, therefore, to determine the most likely hidden sequence of phones that form linguistically meaningful words from the observed speech signal. Due to their ability to estimate the probabilities of hidden events (e.g. sequence of phones) from observed ones (e.g. speech signal) hidden Markov models (HMMs) are ideal for this task.

The HMM-based speech recognition problem can be conceptualized as a special case of the Bayesian inference problem (Jurafsky and Martin, 2008). The probability that the utterance W was spoken given the acoustic evidence O can be formulated as:

$$\underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} \frac{P(O|W) \times P(W)}{P(O)} \tag{5.1}$$

Since the probability of the observation sequence $P(O)$ doesn't change with each sentence hypothesis, the denominator of Equation 5.1 can be ignored and the problem reduces to:

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} P(O|W) \times P(W) \tag{5.2}$$

$P(O|W)$ in Equation 5.2, is the observation likelihood computed by an HMM-based acoustic model while $P(W)$ is the prior probability computed by a language model. The most probable string of words for a given observation sequence O is, therefore, the one for which the product of the two probabilities is maximum.

The main tasks involved in building an HMM-based speech recognition can be divided into three major subtasks – feature extraction, model training and decoding (recognition). Feature extraction is the first step that transforms the observed speech signal into a sequence of feature vectors. The training procedure estimates the parameters of a set of HMMs using training data and the associated transcription resulting in an acoustic model. The decoding task attempts to map the observed sequence of feature vectors to the hidden underlying sequences of symbols using the trained acoustic model and other recognition resources. The acoustic model should be trained on a large amount of training data prior to using the system to recognize a spoken utterance.

A basic architecture of an automatic speech recognition system is shown in Figure 5.1.

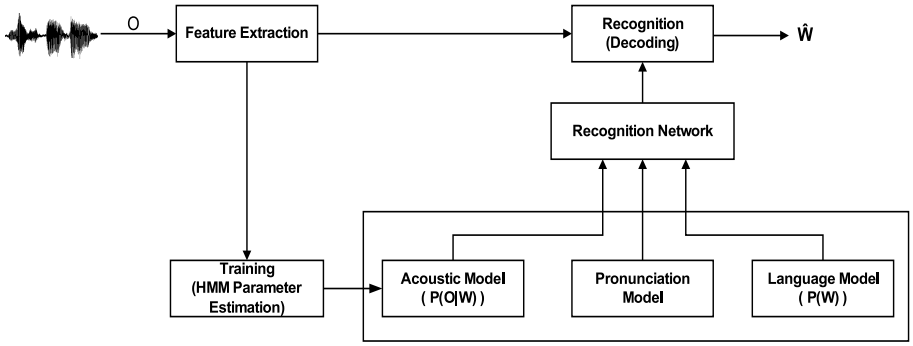

Figure 5.1: A simplified architecture of an automatic speech recognition system

5.2.1 Feature Extraction

The purpose of feature extraction as mentioned in Section 3.2.1 is to transform a given speech signal into a sequence of discrete acoustic feature vectors that are robust to acoustic, inter-speaker and intra-speaker variability but sensitive to linguistic content (Gold and Morgan, 2000). Each acoustic feature vector represents spectral and energy information of a short-term speech signal. For the duration covered by a single feature vector, a speech waveform can be assumed to be stationary (Young et al., 2006).

We look into various feature extraction methods in search of those features that are better suited for speech recognition over the telephone, gender recognition and accent detection. In particular, Mel-Frequency Cepstral Coefficients (MFCCs) (Davis and Mermelstein, 1980), Perceptual Linear Prediction (PLP) cepstral coefficients (Hermansky, 1990), Linear Predictive Coding (LPC) features (Atal and Hanauer, 1971; Itakura and Saito, 1968) and LPC-based Cepstral Coefficients (LPCC) (Atal, 1974) are investigated.

A summary of the various feature extraction methods used in this thesis is presented in Figure 5.2.

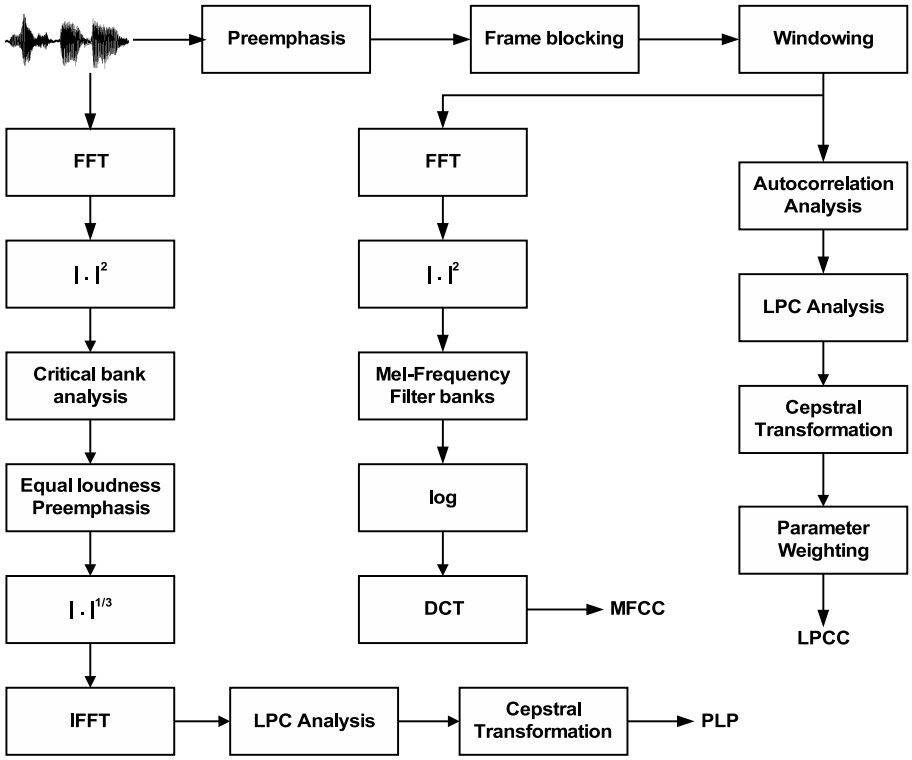

Figure 5.2: Block diagram of feature extraction methods

A description of the feature extraction techniques shown in Figure 5.2 as they are used in this thesis is presented below.

The computation of MFCCs consists of performing preemphasis on the acoustic signal, dividing the incoming waveform into overlapping blocks of 20–25 ms, and multiplying each block by a Hamming Window. The Fast Fourier Transform (FFT) of the windowed signal is computed and the square of the magnitude (i.e., the power spectrum) is fed to a series of filter bank channels. Then, Discrete Cosine Transform (DCT) is applied to the logarithm of the filter bank outputs. The Discrete Cosine Transform has a notable effect in favor of the diagonal covariance assumption commonly used in HMM-based acoustic modeling by de-correlating the features in the feature vectors so that the features can be assumed to be independent of each other. Finally, the first and second time differences (i.e., delta, and delta-delta coefficients) are computed to better model temporal variation of the speech spectrum. A feature vector is typically generated every 10 ms

each containing 13 cepstral components, including the 0^{th} order cepstral coefficient and the corresponding delta and delta-delta coefficients comprising 39 dimensions.

Linear predictive coding (LPC) views speech as a linear but time-varying system, where speech sample at discrete time t is estimated as a linear combination of the previous p samples. The computation of LPC coefficients consist of preemphasis, frame blocking, windowing, autocorrelation analysis (Itakura and Saito, 1968) followed by LPC analysis which converts the autocorrelation coefficients to an LPC parameter set. The resulting LPC coefficients are highly correlated and the diagonal covariance assumption does no longer hold true. Consequently, using LPC-derived cepstral coefficients (LPCC) rather than the LPC coefficients directly can be more useful because the additional cepstral transformation de-correlates the features in favor of the diagonal covariance assumption. Therefore, the LPC parameters are transformed to cepstral coefficients which are known to be more robust and reliable features for speech recognition (Rabiner and Juang, 1993) than the LPC coefficients. The cepstral coefficients are then weighted by a tapered window so as to minimize the sensitivity of the low-order cepstral coefficients to the overall spectral slope and that of the high-order cepstral coefficients to noise (Rabiner and Juang, 1993). Finally, the first and second derivatives can be computed to account for the temporal information of the speech signal. A typical feature vector of dimensionality 39 containing 13 cepstral coefficients including the energy term along with the delta and delta-delta features is used for speech recognition. A more complete discussion on linear predictive analysis can be found in (Makhoul, 1975; Rabiner and Juang, 1993).

Perceptual Linear Prediction is an LP-based analysis method that incorporates the known perceptual properties of human hearing; namely, critical band frequency resolution, preemphasis with an equal loudness curve, and the power law model of hearing. To compute PLP cepstral coefficients, a Fourier transform is first applied to compute the short-term power spectrum and the power spectrum is fed into a perceptually motivated filter bank. The resulting spectrum is multiplied by the equal loudness curve and raised to the power of 0.33 to simulate the power law of hearing (Stevens, 1957). The all-pole model of LPC is applied on the simulated auditory spectrum to give a smooth and compact approximation. Then cepstral coefficients are computed. On top of the static coefficients, the first and the second time differences between parameter values over successive frames – delta, and delta-delta coefficients are computed. As described in (Hermansky, 1990),

PLP features are more suitable in noisy conditions due to the use of different non-linearity compression (the cube root) instead of the logarithm on the filter-bank output.

Besides, cepstral mean normalization (CMN) can be applied to deal with additive noise and mismatch due to different microphone characteristics.

The experiments carried out using these features and various parameters for speech recognition are discussed in Sections 8.2.8, 8.2.9 and 8.3.2. For gender recognition and accent detection, the corresponding experiments are discussed in Sections 8.4.1 and 8.5.1, respectively.

5.2.2 Hidden Markov Model: Overview

A hidden Markov model (HMM) is composed of a hidden process (a Markov chain) and an observable process which associates the observed acoustic feature vectors to the states of the hidden process (Junqua and Haton, 1995). A hidden Markov model is characterized by a set of hidden states, a set of observations, state transition probability distribution, emission probability distribution and initial state distribution. The transition probabilities between states model the temporal variability while the emission probabilities model the spectral variability of speech. Figure 5.3 depicts a three-state, left-to-right hidden Markov model.

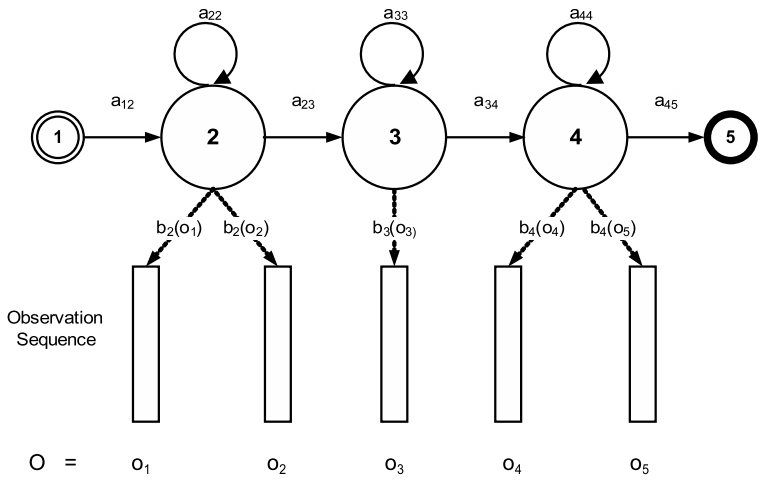

Figure 5.3: HMM-based phone model: Adapted from (Young, 1996)

As can be seen in Figure 5.3, each state j has an associated probability distribution $b_j(o_t)$ which determines the probability of generating observation o_t at time t and each

pair of states i and j has an associated transition probability a_{ij}. The model entry and exit states are non-emitting and consequently have no output probability distribution associated with them. They are used to glue models (HMMs) together to form composite HMMs that represent a word or an utterance.

In order to use HMMs for automatic speech recognition, three fundamental problems must be solved (Junqua and Haton, 1995; Rabiner and Juang, 1993). These are:

Problem 1 (Evaluation): Given a sequence of observations $O = o_1, o_2, ..., o_T$ and a model λ, how do we compute the probability that the model produced the observed sequence? This is a problem of evaluating how well a given model matches a given observation sequence. The forward pass of the forward-backward algorithm (Baum et al., 1970) can be used to solve this problem.

Problem 2 (Decoding): Given the observation sequence $O = o_1, o_2, ..., o_T$ and the model λ, what is the single best state sequence $Q = q_1, q_2, ..., q_T$ in the model that best explains the observations? This problem can be solved with the Viterbi algorithm (Viterbi, 1967). The Viterbi algorithm is based on dynamic programming and it looks through a network of nodes for a sequence of HMM states that most closely corresponds to the input.

Problem 3 (Learning): Given the observation sequence $O = o_1, o_2, ..., o_T$ and the model λ, how do we adjust the model parameters to maximize the probability of generating the observations? The Baum-Welch re-estimation algorithm (Baum et al., 1970) can be used to solve this problem using a finite observation sequence as training data.

Further details on the above mentioned problems and the corresponding algorithms can be found in (Jelinek, 1976; Rabiner, 1989; Rabiner and Juang, 1993; Wendemuth, 2004; Young et al., 2006).

5.2.3 HMM-based Acoustic Modeling

As noted in the previous section, an HMM consists of a set of states and changes state once every time unit. Each time t that a state j is entered, a feature vector o_t is generated with output probability density $b_j(o_t)$ (Young, 1996). The transition from one state to the other is probabilistic and the observation sequence is also a probabilistic function of the underlying states and state transitions.

The goal of acoustic modeling is to estimate the transition probabilities (A) and the observation likelihoods (B) of each HMM such that the likelihood of the training data is maximized.

The likelihood of generating an observed feature vector sequence $O = o_1, o_2, ..., o_T$ while following a state sequence $Q = q_1, q_2, ..., q_T$ given some model λ can be estimated by multiplying together all the acoustic likelihoods and the transition probabilities associated with the given sequence as in Equation 5.3.

$$P(O,Q|\lambda) = a_{q(0)q(1)} \prod_{t=1}^{T} b_{q(t)}(o_t) a_{q(t)q(t+1)} \tag{5.3}$$

where $q(0)$ is the model entry state and $q(T+1)$ is the model exit state.

In HMM, however, we only know the observation sequence O and the underlying state sequence Q is hidden. Therefore, $P(O|\lambda)$ can be found by summing Equation 5.3 over all possible state sequences in the model. This can be efficiently performed with the Baum-Welch algorithm or forward-backward algorithm which is a specific implementation of the Expectation-Maximization algorithm to find the Maximum-Likelihood estimate of both the transition and observation probabilities (Jelinek, 1976; Rabiner, 1989; Rabiner and Juang, 1993; Wendemuth, 2004; Young et al., 2006).

Equation 5.3, can be re-written in the log domain to separate the A (transition probability) and B (observation likelihood) terms as:

$$logP(O,Q|\lambda) = \sum_{t=0}^{T} loga_{q(t)q(t+1)} + \sum_{t=1}^{T} logb_{q(t)}(o_t) \tag{5.4}$$

The observation likelihood distribution term in Equation 5.4 can be represented by a mixture of Gaussian probability distribution functions where the means, covariances and mixture weights are to be learned from training data. The likelihood of an observation vector o_t being generated at time t from an HMM state j ($b_j(o_t)$) can be computed by assuming that the possible values of each dimension of the feature vectors is a weighted mixture of multivariate Gaussians.

A multivariate Gaussian is defined by a D-component mean vector μ and a covariance matrix Σ. The use of full co-variance matrix for acoustic likelihood estimation is computationally expensive and requires much more training data. Therefore, diagonal covariance matrix is commonly used that significantly reduces the required computation

by assuming that the components of a feature vector are statistically uncorrelated. With MFCCs, PLPs, and LPCCs this assumption is justified as discussed in Section 5.2.1.

The acoustic likelihood $b_j(o_t)$ for a D-dimensional feature vector o_t is a weighted sum of M component densities for a given HMM state with mean vector μ_{jm} and covariance matrix Σ_{jm} given by:

$$b_j(o_t) = \sum_{m=1}^{M} \omega_{jm} \frac{1}{\sqrt{(2\pi)^D |\Sigma_{jm}|}} exp(-\frac{1}{2}(o_t - \mu_{jm})^T \Sigma_{jm}^{-1}(o_t - \mu_{jm})) \tag{5.5}$$

where M is the number of Gaussian components per state, ω_{jm} is the weight of the m^{th} component in state j, and $\omega > 0$, $\sum_{m=1}^{M} \omega_{jm} = 1$ for all j.

Before a model can be trained, a number of modeling decisions must be made including the topology of the HMM, number of states per HMM, the type of output probability function and the initialization method. In our case, we represent each monophone by a hidden Markov model of three emitting states with left-to-right topology, where each emitting state has two transitions: back to itself and to the next state as shown in Figure 5.4. The left-to-right topology is generally used to account for the strong temporal constraints in speech (Junqua and Haton, 1995). We use continuous density HMMs (Liporace, 1982) where each observation probability distribution is represented by a finite mixture of Gaussian functions as described earlier.

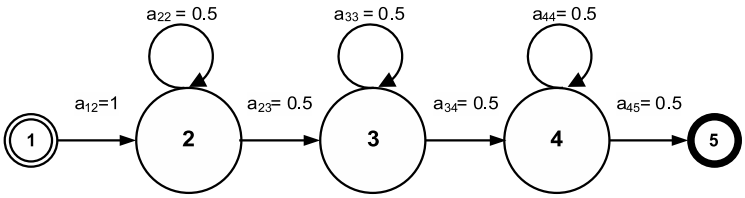

Figure 5.4: An example HMM topology with initial transition probabilities

The simplest way to initialize HMMs is with a flat start scheme (Young et al., 2006) where the mean and the variance of each Gaussian is set to the global mean and variance of the training data. In a flat start scheme, transition probabilities from an emitting state back to itself and to the next state are set equiprobable, the transition from the entry state to the first emitting state is set to 1.0 and all other transitions are set to zero as shown in Figure 5.4.

Most often HMM training is done with speech utterances for which phone-level transcription of the training data is available but the exact timing of phonetic segments is not known. The segmentation of speech into phones and phone alignment is done as part of the training process. This type of training procedure where each phone model is trained embedded in an entire sentence is known as embedded training (Jurafsky and Martin, 2008; Young et al., 2006).

More on acoustic model training is presented in Section 8.5.1.2.

5.2.4 Language Modeling

A language model is an important source of information that limits the set of possible sequences of words which are actually considered for a given recognition task. For every word in the vocabulary of a given application, the language model defines the list of words that can follow it with associated probability. As discussed in Section 5.1, the term $P(W)$ represents the contribution of linguistic knowledge in the form of a language model in the recognition process. For a sequence of words $W = w_1, w_2, w_3, ..., w_n$, $P(W)$ is given by:

$$\begin{aligned} P(W) &= P(w_1, w_2, ..., w_n) && (5.6)\\ &= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)...P(w_n|w_1, w_2, ..., w_{n-1}) && (5.7)\\ &= \prod_{i=1}^{n} P(w_i|w_1, w_2, ..., w_{i-1}) && (5.8) \end{aligned}$$

where $P(w_i|w_1, w_2, ..., w_{i-1})$ is the probability that w_i will follow, given the fact that the word sequence $w_1, w_2, ..., w_{i-1}$ has been observed. However, the conditional probability $P(w_i|w_1, w_2, ..., w_{i-1})$ cannot be reliably estimated even for moderate values of i (Huang et al., 2001b). In practice, therefore, an approximation is made where the probability of a word is assumed to be dependent only on the preceding $i-1$ words where i is either two (bigram) or three (trigram).

Since the vocabulary of each application domain we consider in this thesis is sufficiently limited, we use backoff-bigram language models. A bigram language model is essentially a matrix containing the probability of a given word being followed by another calculated from a training corpus. Backoff is a smoothing technique commonly used in speech recognition. When there are not enough examples of a particular N-gram, backoff uses lower order N-gram language models. For instance, when there are not enough

examples of a particular trigram, bigram probabilities are used and when there are not enough examples of a particular bigram, unigram probabilities are calculated.

5.2.5 Decoding

Decoding is a task of determining the sequence of words that has the highest posterior probability, given a sequence of observation vectors. As described in Section 5.2, the best sequence is the one that maximizes the product of the language model prior probability and the acoustic likelihood given by:

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} P(O|W) \times P(W) \tag{5.9}$$

The acoustic model likelihood – $P(O|W)$ is described in Section 5.2.3 and the language model prior – $P(W)$ is briefly described in Section 5.2.4. $P(O|W)$ relies on some incorrect independence assumption; i.e., successive observation vectors are assumed to be independent of past observations and states. This assumption underestimates the acoustic likelihood (Jurafsky and Martin, 2008).

Therefore, it is desirable to balance the probabilities of the acoustic model and the language model by finding an optimal language model scaling factor (LMSF) that defines how the language model log probabilities are scaled before they are combined with the acoustic log probabilities. Introducing a language model scaling factor may result in an increase in word insertion errors in the recognition output. To mitigate this effect, an optimal word insertion penalty (WIP) is introduced. Consequently, Equation 5.9 can be modified to:

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(O|W) \times P(W)^{LMSF} WIP^{N} \tag{5.10}$$

where N is the number of words in the utterance.

In practice, we do all computations in the log-domain, where multiplications of probabilities become additions which make the computation more efficient and avoids numeric underflow for long sequences. Therefore, Equation 5.10 can be re-written in the log-domain as:

$$\hat{W} = \underset{W}{\operatorname{argmax}} logP(O|W) + LMSF \times logP(W) + N \times logWIP \tag{5.11}$$

The LMSF and WIP values are experimentally determined for each application domain as will be described in Section 8.2.7.

The acoustic likelihood $P(O|W)$ can be approximated by considering the most likely state sequence which can efficiently be computed using the Viterbi algorithm. In large vocabulary speech recognition systems, a complete Viterbi search slows the system and beam search (Lowerre, 1976) is commonly used where unlikely candidates (paths) whose partial path scores lie more than a beam-width below the best score are pruned at the earliest stage.

5.3 User-Group Dependent Acoustic Models

An utterance conveys not only the intended message but also speaker-dependent information such as gender, accent, age group, etc. As defined in Section 2.4, accent as used in this thesis refers to the linguistic phenomenon in which specific pronunciation patterns from ones native language are introduced when speaking a foreign language.

Given a spoken utterance, it is easy to tell the gender, accent and/or age-group of a person with a high degree of accuracy regardless of the language spoken or the communication channel used. This suggests that there are common vocal features that a group of speakers belonging to the same gender, accent, age-group, etc. share. Identifying and using these vocal features to reliably estimate the gender, age or accent of a person automatically from ones spoken utterance can be useful to improve speech recognition accuracy. This is particularly important in spoken dialog systems, as it is often the case that there is considerable mismatch between training and actual usage environments in such applications. For instance, a speaker-independent (SI) model built using speech samples from a large group of native-speakers of English would perform very poorly with non-native speakers with typical accent. In automatic speech recognition systems, as reported in (Huang et al., 2001a) a mismatch in accent between the speakers used in testing and training can lead to over 30% increase in word error rate (WER). It has also been reported in (Tomokiyo, 2001) that on the same task, the word error rate is about 3–4 times higher on strongly Japanese-accented or Spanish-accented English speakers than on native English speakers. This suggests that accent-specific acoustic models tailored to the vocal characteristics of speakers in the same native language group, can perform much better than a generic acoustic model.

In this thesis, we group the users of the spoken dialog system based on gender and accent. In order to use accent-dependent and gender-dependent recognition resources, it is necessary to accurately estimate the accent and the gender of a speaker from a spoken utterance. To this end, we investigate various types of acoustic features in search of those that could detect gender and accent better. Details of experiments and results are presented in Sections 8.4.1 and 8.5.1.

5.3.1 Gender and Accent Recognition

It is a well-grounded fact that the most salient cue for distinguishing adult male and female speech is the fundamental frequency (F0) – pitch (Hillenbrand et al., 1995; Linke, 1973; Linville and Fisher, 1985; Murry and Singh, 1980). However, in telephone speech much of the low frequency energy is filtered out due to the band-limiting effect of the telephone channel. Hence, the required pitch information is either missing or weak. Therefore, we investigate various cepstral features commonly used in speech recognition in search of those that are suitable to reliably estimate the gender of a speaker from a spoken utterance. The results are quite promising as will be discussed in detail in Section 8.4.1.

For accent recognition, various approaches that use either accented speech data or other linguistic and acoustic features to build accent recognition models have been proposed as discussed in Section 2.6. The novelty of our approach for accent detection is on using native speech data of the target accent groups (viz. German and English) to train a model that decides if an utterance is accented or native. Even though the task on which we demonstrate the method is relatively simple, the approach can easily be extended to detect multiple accents using available native speech corpora of the target accent groups. The rationale for using native speech data instead of accented data for accent detection is twofold. First, we believe that accent-related information could be effectively captured from the native language speech of a speaker. Second, it is often hard to collect enough amount of accented data to build a reliable model. Therefore, it would be an advantage, if a reliable accent recognizer could be trained on existing speech corpora containing native spoken utterances of different languages. The model we built can serve the purpose with high accuracy as will be described in Section 8.5.1.3. This further motivates the use of cross-language accent adaptation, where native speech data of the target accent can

be used as enrollment data to adapt speaker-independent (SI) models trained on native speech data of another language (e.g., English).

The gender and accent recognition models are based on a Gaussian mixture model (GMM) to recognize the gender or accent of a speaker given the parameters of a spoken utterance. In the following section we describe Gaussian Mixture model as used in gender and accent recognition.

5.3.2 Gaussian Mixture Model

A GMM can be modeled as a single-state hidden Markov model (HMM) with a Gaussian mixture observation density with diagonal covariance matrix where there is no state transition probability within the model as shown in Figure 5.5.

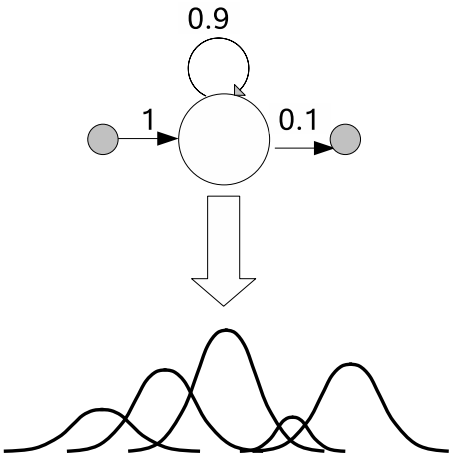

Figure 5.5: A GMM modeled as a single-state HMM

The parameters of an utterance are modeled with mixture weights, mean vectors, and variance parameters of the component densities. Assuming that successive observation vectors are independent of past observations and states, the log likelihood of a model λ for an observation sequence $O = o_1, o_2, ..., o_T$ is given by:

$$logp(O|\lambda) = \sum_{t=1}^{T} logp(o_t|\lambda) \tag{5.12}$$

where $P(o_t|\lambda)$ for a D-dimensional feature vector o_t is a weighted sum of M component densities for a given model λ given by:

$$p(o_t|\lambda) = \sum_{m=1}^{M} \omega_m b_m(o_t) \tag{5.13}$$

where ω_m is the m^{th} mixture weight and $\sum_{m=1}^{M} \omega_m = 1$.

Each component density $b_m(o_t)$ is a multivariate Gaussian function with mean vector μ_m and covariance matrix Σ_m given by:

$$b_m(o_t) = \frac{1}{(2\pi)^{D/2}|\Sigma_m|^{1/2}} exp\{-\frac{1}{2}(o_t - \mu_m)^T \Sigma_m^{-1}(o_t - \mu_m)\} \tag{5.14}$$

The objective is to find the model λ that has the maximum a posteriori probability for the input feature vector sequence O according to Equation 5.12.

5.3.3 Gender and Accent Dependent Acoustic Models

The simplest approach to obtain gender and accent dependent acoustic models is to train separate acoustic models for each gender and accent group using gender specific data or accented speech data from the target accent group. Building gender-dependent models using gender-specific data is feasible as long as enough training data is available for each gender group. However, using accented speech to build accent-dependent models is not feasible as collecting accented data in large enough amount to train reliable model is often hard, if not impractical. Nevertheless, the presence of within-accent acoustic similarity due to similar pronunciation pattern learnt from the same mother tongue suggests that accent specific characteristics can be captured from some adaptation data to transform the model parameters of the initial model to obtain accent-dependent models. Therefore, we investigated the effectiveness of adapting speaker-independent and gender-dependent models to the German accent and we obtained significant performance gain as will be discussed in Section 8.5.2.

Another possibility is to use cross-language accent adaptation, where native speech data from the target accent group of users is used to adapt a speaker-independent model trained on native speech data of another language. This is motivated by the fact that accent-related information could be effectively captured from native speech data of a group of speakers as discussed in Section 5.3.1. We show that promising performance

gain can be obtained by using cross language accent adaptation as will be discussed in Section 8.5.3.

5.3.4 Channel Adaptation

In the absence of enough amount of telephone-recorded data in a given application domain, one may have to use microphone-recorded data to train SI models for use in a telephone-based spoken dialog system. A commonsensical approach in such cases is to "simulate" telephone quality speech from microphone-recorded speech data by introducing the obvious effects of the telephone channel into the training speech data.

Simulating telephone quality speech involves down sampling the audio data to 8 kHz and applying a low-pass filter with a cutoff frequency of 3400 Hz and a high-pass filter of 300 Hz to approximate the band-limiting effects of the telephone channel. Furthermore, to approximate the loss due to the logarithmic encoding in the telephone channel, the 16-bit quantized signals are converted to A-Law companded signal and back to linearly quantized 16-bit signal. We then built a SI model using the "simulated" training data. However, the model so built performs not so well on actual telephone speech. Therefore, we adapted the resulting speaker-independent model to the telephone channel characteristics using a small amount of telephone recorded data to improve the performance of the speech recognizer on actual telephone speech. A modest performance gain has been achieved as will be discussed in Section 8.16.

5.4 Speaker Adaptation Techniques: Overview

Speaker adaptation techniques use information provided in an adaptation data to adjust the parameters (i.e., mean and variance of the Gaussian density functions) of the initial model to reflect the characteristics of the current environment or speaker. Considering the existence of within-accent acoustic similarity due to similar vocal characteristics of speakers in the same group as discussed in Section 5.3, adapting a speaker-independent and gender-dependent acoustic models to a particular accent can give robust user-group dependent acoustic models. A simplified schematic representation of speaker adaptation as used in HMM-based speech recognition models is shown in Figure 5.6.

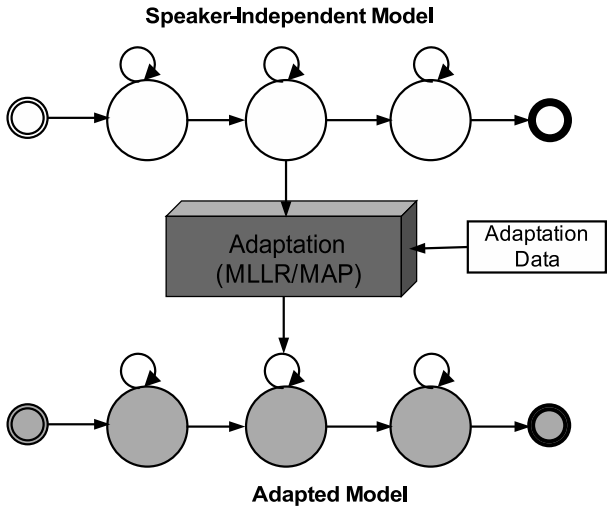

Figure 5.6: Schematic representation of speaker adaptation as used in HMM-based speech recognition systems

Maximum Likelihood Linear Regression (MLLR) and Maximum a Posteriori (MAP) adaptation techniques are briefly described in Sections 5.4.1 and 5.4.2. MAP and MLLR are known as model-based adaptation methods because the acoustic model parameters are modified based on the adaptation data from the new speaker(s) as opposed to speaker normalization (also known as feature-based adaptation) methods where the input feature vectors are normalized to match the parameters of the model.

5.4.1 Maximum Likelihood Linear Regression (MLLR)

Maximum Likelihood Linear Regression (Leggetter and Woodland, 1995b) estimates linear transformations for model parameters to maximize the likelihood of the adaptation data. The transformations modify the component means and covariances in the initial system so as to reduce the mismatch between the initial model set and the adaptation data.

For mixture component s, the transformation of the means is given by:

$$\hat{\mu}_s = W_s \xi_s \tag{5.15}$$

where W_s is the transformation matrix (i.e., an $n \times (n+1)$ matrix where n is the dimensionality of the feature vectors) and ξ_s is the extended mean vector given by:

$$\xi_s = [1 \;\; \mu_1 \;\; \mu_2 \;\; \ldots \;\; \mu_n]^T$$

The transformation matrix W_s is computed to maximize the likelihood of the adaptation data using the Expectation-Maximization (EM) algorithm.

The probability density of a feature vector o_t being generated by distribution s is, therefore, given by:

$$b_s(o_t) = \frac{1}{(2\pi)^{D/2}|\Sigma_s|^{1/2}} exp\{-\frac{1}{2}(o_t - W_s\xi_s)^T \Sigma_m^{-1}(o_t - W_s\xi_s)\} \tag{5.16}$$

Variance transformation can be applied using:

$$\hat{\Sigma} = B^T H B \tag{5.17}$$

where B is the inverse of the Choleski factor of Σ^{-1}, so that

$$\Sigma^{-1} = CC^T$$

and

$$B = C^{-1}$$

H is the $n \times n$ transformation matrix to be estimated.

When the adaptation data is very small a single global transform can be applied to every Gaussian component in the model set. When enough adaptation data is available and more rigorous transformation is required, a regression class tree can be used to cluster acoustically similar Gaussians into regression classes, so that similar components can share a common transform. This makes adaptation of distributions for which there were no observations in the adaptation data possible (Young et al., 2006).

The adaptation of the transition probabilities and the mixture component weight will have little effect on the final performance (Leggetter and Woodland, 1995a). However, transformation of the diagonal covariance matrix can give performance improvement. Further details on MLLR transformation can be found in (Gales, 1998; Leggetter and Woodland, 1995b).

5.4.2 Maximum a Posteriori (MAP) Adaptation

Maximum a Posteriori (MAP) (Gauvain and Lee, 1994) estimation (also known as Bayesian adaptation) maximizes the a posteriori probability using prior knowledge about the model parameter distribution. The prior information prevents large deviations of the parameters unless the new training data provide strong evidence (Huang et al., 2001b). Generally, the speaker-independent model parameter distribution is the prior information used in MAP adaptation. Given good models and large amount of adaptation data, MAP can perform better than MLLR. MAP is a re-estimation procedure; consequently, the adaptation data required is larger than the amount required for MLLR transformation.

For a state j and a mixture component m, the mean is computed as (Young et al., 2006):

$$\hat{\mu}_{jm} = \frac{N_{jm}}{N_{jm}+\tau}\bar{\mu}_{jm} + \frac{\tau}{N_{jm}+\tau}\mu_{jm} \tag{5.18}$$

where μ_{jm} is the mean of the speaker-independent model and $\bar{\mu}_{jm}$ is the mean of the observed adaptation data, τ is the weighting of the a priori knowledge to the adaptation speech data, and N is the occupation likelihood of the adaptation data, given by:

$$N_{jm} = \sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{jm}^{r}(t)$$

where $L_{jm}^{r}(t)$ is the occupancy probability for state j and mixture component m at time t of sequence r.

The mean of the observed adaptation data $\bar{\mu}_{jm}$ is given by:

$$\bar{\mu}_{jm} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{jm}^{r}(t)o_t^r}{\sum_{r=1}^{R}\sum_{t=1}^{T_r} L_{jm}^{r}(t)}$$

More details on MAP adaptation technique can be found in (Gauvain and Lee, 1994; Young et al., 2006).

MAP and MLLR can be effectively combined to improve the performance of a recognizer further by using the MLLR transformed means as the priors for MAP adaptation.

5.5 Summary

In this chapter we described a hidden Markov model (HMM) based speech recognition with some details on feature extraction, acoustic modeling, language modeling and decoding. Gaussian mixture model (GMM) based gender and accent recognition models using cepstral features are also described. The rationale for using native speech data of two or more accent groups to train an accent recognizer are described. Then we discussed the use of user-group dependent acoustic models in spoken dialog systems to achieve robustness using gender-dependent and accent-specific models. Finally, we briefly described the standard speaker adaptation techniques – MLLR and MAP since they will be used for accent and channel adaptation in Sections 8.5.2, 8.5.3 and 8.16. The results of the experiments corresponding to this chapter are presented in Sections 8.2, 8.3, 8.4 and 8.5.

Chapter 6

Spoken Language Understanding

6.1 Introduction

Automatic speech recognition systems generally output the most probable transcription of a spoken utterance or an N-best lattice of possible transcriptions and are not required to perform the subsequent semantic and/or syntactic analysis. Even though the correct recognition of a spoken utterance is vitally important, a spoken dialog system can hardly serve any purpose using only the raw output of a speech recognizer. Therefore, spoken dialog systems incorporate a spoken language understanding (SLU) unit that is responsible to infer the meaning underlying a recognized utterance.

One approach to SLU is to "enrich" the output of the speech recognizer with semantic information so that the added information can later be used to infer what is meant from what is said. A spoken utterance can be conceived as a hidden sequence of semantic concepts expressed in words or phrases. The goal of the required model is, therefore, to determine the most likely sequence of the hidden semantic concepts that could have generated the observed sequence of words. In other words, the problem of understanding the meaning underlying a spoken utterance in a spoken dialog system can be partly solved by decoding the hidden sequence of semantic concepts from the observed sequence of words. This can efficiently be realized using hidden Markov model. In this chapter, we introduce a new, robust, hierarchical, HMM-based approach to semantic concept labeling that offers a number of advantages over the conventional flat-concept approach.

The rest of the chapter is organized as follows. After briefly reviewing the different approaches to spoken language understanding in Section 6.2, we describe how the hidden

Markov model fits in the task of spoken language understanding in Section 6.3. In Section 6.4, we describe the smoothing technique we use to account for the sparse data problem. A description of a flat-concept semantic tagging model trained on semantically unlabeled data is provided in Section 6.5. In Section 6.6, we extend the flat-concept model so as to capture longer context and resolve ambiguity. Section 6.7 introduces a robust hierarchical semantic concept labeling model that organizes low-level semantic concepts into higher-level hierarchical structures. The hierarchical model is integrated in our spoken dialog system and encodes longer context, offers better ambiguity resolution ability, has better predictive power and provides semantically richer output than the flat-concept model. Moreover, the required additional human effort to design the proposed model is much less than the time and effort that would be required to semantically annotate the training data which would also require a detailed analysis of the application domains to define semantic labels and organize them into hierarchical structures.

6.2 Approaches to Spoken Language Understanding

As discussed in Section 2.7, the approaches in the domain of spoken language understanding can be broadly classified as knowledge-based, statistical, and a combination of the two. Knowledge-based systems rely on the lexical, syntactic, semantic, discourse, etc. knowledge encoded in the system. These systems perform very well when the structure of the spoken utterance is covered in the knowledge stored in the system. Nevertheless, crafting the required syntactic and semantic knowledge in order to extract meaning from a given utterance requires a great deal of expertise and heavy human involvement. Moreover, they are often fragile in the face of unexpected input. Examples of knowledge-based systems include TINA of MIT (Seneff, 1992), PHOENIX of CMU (Ward and Issar, 1996), and GEMINI of SRI (Dowding et al., 1994).

Statistical approaches, on the other hand, estimate model parameters from data by counting the frequencies of transitions between states, word observations while in each state and which states start a sentence. These statistical models are robust, require less human supervision and expertise, and perform well. However, they require a large corpus of fully annotated training examples, which is often not available for many application domains. Examples of statistical systems include CHRONUS of AT&T (Pieraccini and Levin, 1993), and Hidden Understanding Model (HUM) of BBN (Miller et al., 1994).

Another popular statistical approach that can be trained on "lightly" annotated data is the hidden vector state model of Cambridge University (He and Young, 2005) where state transitions between two states are decomposed into separate stack operations that transform one state to the other.

Approaches to integrate knowledge-based approach into a statistical learning framework have also been investigated. The generative HMM/CFG composite model described in (Wang et al., 2005) is a typical example.

We can classify statistical spoken language understanding systems into two categories based on whether they can encode hierarchical information or not. Various statistical approaches that encode hierarchical structure have been proposed in (Charniak, 2001; Chelba and Jelinek, 2000; Erdogan et al., 2002; Fine et al., 1998; Miller et al., 1994). However, all these models require a large amount of annotated training data for parameter estimation. The hidden vector state model (He and Young, 2005) mentioned earlier encodes context using only an abstract annotation for each utterance.

The approach to spoken language understanding we propose in this thesis does not require semantically annotated training data, instead uses a commonplace prior domain knowledge to counterbalance the lack of annotated training corpus.

6.3 HMM in Spoken Language Understanding

Hidden Markov model (HMM) as used in speech recognition is described in Section 5.2.2. In HMM-based semantic concept labeling, the hidden states correspond to the semantic concepts in a given application domain while the observation set corresponds to the set of words in the lexicon of the system. The model parameters are the transition probabilities between states, observation probabilities in each state and which states start a sentence. These parameters should be given good initial values so that the Expectation-Maximization (EM) algorithm could effectively be used to iteratively refine these parameters during training.

The HMM-based semantic labeling problem, like the HMM-based speech recognition problem discussed in Section 5.2, is essentially a Bayesian inference problem. Assuming that the string of words $\overrightarrow{W} = w_1, w_2, ..., w_n$ hypothesized by the speech recognizer was generated by some hidden sequence of semantic concepts $\overrightarrow{S} = s_1, s_2, ..., s_n$, the goal is to

assign the most likely sequence of semantic concept labels to the sequence of words in the recognized utterance. Hence, the problem can be formulated as:

$$\underset{\overrightarrow{S}}{\operatorname{argmax}} P(\overrightarrow{S}|\overrightarrow{W}) = \underset{\overrightarrow{S}}{\operatorname{argmax}} \frac{P(\overrightarrow{W}|\overrightarrow{S}) \times P(\overrightarrow{S})}{P(\overrightarrow{W})} \tag{6.1}$$

As we are interested in evaluating different sequences of semantic labels for the same observation sequence, the denominator in Equation 6.1 can be ignored. Therefore, the problem reduces to:

$$\underset{\overrightarrow{S}}{\operatorname{argmax}} P(\overrightarrow{S}|\overrightarrow{W}) = \underset{\overrightarrow{S}}{\operatorname{argmax}} P(\overrightarrow{W}|\overrightarrow{S}) \times P(\overrightarrow{S}) \tag{6.2}$$

Since it is hard to compute Equation 6.2 directly, the following simplifying assumptions are commonly used.

- Independence assumption: the probability of a word in a sentence depends only on its state and is independent of other words around it.
- First order Markov assumption: the probability of a state depends only on the previous state.

Consequently, Equation 6.2 reduces to:

$$\underset{\overrightarrow{S}}{\operatorname{argmax}} P(\overrightarrow{S}|\overrightarrow{W}) \approx \underset{\overrightarrow{S}}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i|s_i) \times P(s_i|s_{i-1}) \tag{6.3}$$

The semantic prior, $P(s_i|s_{i-1})$ in Equation 6.3, models the probability of the system to go to state s_i given the preceding state s_{i-1} while the lexicalization model, $P(w_i|s_i)$, represents the probability that the word w_i is emitted when the system is at state s_i.

If we had a semantically annotated corpus, the maximum likelihood estimate of the semantic prior and the lexicalization model can be computed by collecting frequencies of transitions between states, word observations in each state and states that start a sentence. In the absence of semantically labeled data, these parameters need to be learnt via unsupervised training using the EM algorithm from unlabeled training data. The EM algorithm iteratively maximizes the probability of the training sequences given initial HMM parameter values and guarantees only local maximum. It depends heavily on the initial

parameter values of the model; hence, it is crucial to determine a reasonable model structure and well-informed initial parameter values of the model. The use of prior domain knowledge is particularly essential to provide good initial parameter values and define a reasonable model topology.

A high-level description of the algorithm we used to train the HMM-based semantic models is shown in Listing 3.

Listing 3 The core of the training algorithm

```
hmm.loadProbs(initial_model)
hmm.readSeqs(training_data)
    for k in range (0,maxIterations):
        totalLogProb = hmm.count_seqs()
        if change(totalLogProb) < threshold:
            break
        hmm.updateProbs()
    hmm.saveProbs(trained_model)
```

In Listing 3, the function hmm.loadProbs() loads the initial model compiled as will be described in Sections 6.5, 6.6 and 6.7 while hmm.readSeqs() reads the training data. The parameter "maxIterations" denotes the number of training iterations specified to train the model. One iteration of the Baum-Welch algorithm consists of "hmm.count_seqs()" which computes the expected counts and "hmm.updateProbs()" which estimates the new HMM parameters. If the change in total log likelihood between two iterations no longer increases; i.e., the current model is at a local maximum, the training algorithm terminates.

Given a well-trained model, the highest probability semantic label sequence which corresponds to the sequence of observed words can be computed by the Viterbi algorithm (Viterbi, 1967) as described in Section 4.5.4.

6.4 Smoothing

The occurrence of events in a test-set which were not seen in the training set is inevitable in statistical methods that use a finite amount of training data. This phenomenon is commonly known as the the sparse data problem. Events that can normally occur in spontaneous conversation may not occur at all in a given training data. If these unseen events

are assigned zero probabilities, the system becomes very fragile and fails when these unforeseen events occur. Smoothing is a method used to combat the "zero probability" and the data sparseness problem. There are a number of smoothing approaches such as add-one smoothing (also known as Laplace smoothing) (Lidstone, 1920), back-off smoothing (Katz, 1987), deleted interpolation (Jelinek and Mercer, 1980), etc.

Smoothing, as used in this thesis for semantic modeling, allows a model to parse all utterances including those that contain "unseen transitions" and "out-of-vocabulary" words which otherwise could not be parsed. If we had a semantically annotated training corpus, the maximum likelihood estimate of the semantic prior $P(s_i|s_{i-1})$ can be computed by collecting frequencies of transitions between states, and smoothing can be applied based on these counts. However, since we do not have semantically annotated training data, we compute the expected counts on the training corpus, with the forward-backward recursion of the Baum-Welch algorithm. Then the smoothed parameters are computed on the basis of the expected counts.

To smooth the transition probabilities we used the simple add-one smoothing technique which adds one to all counts before normalizing them into probabilities as shown in Equation 6.4.

$$\hat{P}(s_i|s_{i-1}) = \frac{\hat{C}(s_{i-1},s_i)+1.0}{\hat{C}(s_{i-1})+C(s_{i-1})} \tag{6.4}$$

where $\hat{C}(s_{i-1},s_i)$ represents the expected counts of transitions from state s_{i-1} to state s_i and $\hat{C}(s_{i-1})$ represents the expected state occupation counts computed by the Baum-Welch algorithm. $C(s_{i-1})$ represents the number of all possible transitions that leave the state s_{i-1}. Finally, a lower limit on all transition probabilities that are not explicitly prohibited is imposed so that all possible utterances can be successfully parsed. We set this value to a very small non-zero value – $minProb = 10^{-7}$).

In order to smooth emission probabilities, we use a commonsensical approach where a vocabulary item "oov" is introduced in classes such as CITY, STATE, AIRLINE, DUMMY, etc. where the vocabulary list in the class is not exhaustive. All words not in the lexicon are mapped to the "oov" word and the probability of the "oov" word in a concept class is set to the sum of the probabilities of all words in that class that occur only once in the training set. Since we do not have annotated data, the estimation of these probabilities is based on the expected count of emitting the symbol w_k in state s_j (i.e., $\hat{C}(s_j,w_k)$), as

computed by the Baum-Welch algorithm on the training data. An upper and lower limit to the probability of the "oov" word are set such that $minProb < \hat{P}(oov|s_j) \leq 0.8$. These values are empirically determined. Then, the probabilities in that class are normalized so that they add up to one. This approach, though simple, serves the purpose well as will be discussed in Section 8.6.

6.5 The Flat-Concept Model

A flat-concept model labels each word in an utterance with a corresponding semantic label and does not encode hierarchical relationship between concepts. CHRONUS of AT&T (Pieraccini and Levin, 1993) and IBM's fertility model (Pietra et al., 1997) are two examples using the flat-concept model. In CHRONUS and similar approaches, the model parameters are estimated by simply counting the relative frequencies from semantically annotated training data, where each word is labeled with the state (semantic class) it belongs to. In this section we describe a flat-concept model where no annotated data is required.

Modeling an application domain requires a precise identification of the activities, entities, events, attributes and relations within the domain of discourse. In this thesis, we are interested in two application domains; namely, airline travel planning in English and train inquiries domain in German. A detailed list of concepts that are relevant in each application domain is identified using prior domain knowledge and domain-specific example sentences in the training data. As a result of the detailed domain analysis, we identified over 76 semantic classes[1] in the airline travel planning domain and 51 semantic classes in the domain of train inquiries. Listing 4 depicts a partial listing of concepts identified for the airline travel planning domain. A complete list is provided in Appendix A for both application domains.

Listing 4 Example list of semantic classes (semantic concepts) identified for the airline travel planning domain

```
CITY_P1, CITY_P2, CITY_P3, STATE, COUNTRY, DAY_OF_WEEK, DAY_OF_MONTH,
MONTH, MINUTES, AMPM, HOUR_OF_DAY, FLIGHT_NUMBER, FLIGHT_CLASS, FROM, TO,
ON, AIRLINE_NAME, AIRPORT_NAME, ARRIVAL, DEPARTURE, YES, NO, DUMMY, ...
```

[1] The terms semantic class, semantic concept, semantic label, and semantic tag are used interchangeably in this thesis

As can be observed in Listing 4, a single concept can be broken down into several sub-concepts, in some cases. For instance, the concept CITY is modeled with three sub-concepts – CITY_P1, CITY_P2 and CITY_P3 in order to capture multi-word city names such as "New York City", "Washington D. C." or train stations such as "Berlin Zoologischer Garten", etc.

The initial HMM for the flat-concept based approach is a fully connected network such that any state (semantic concept) can follow any other state (semantic concept) with equal probability as shown in Figure 6.1.

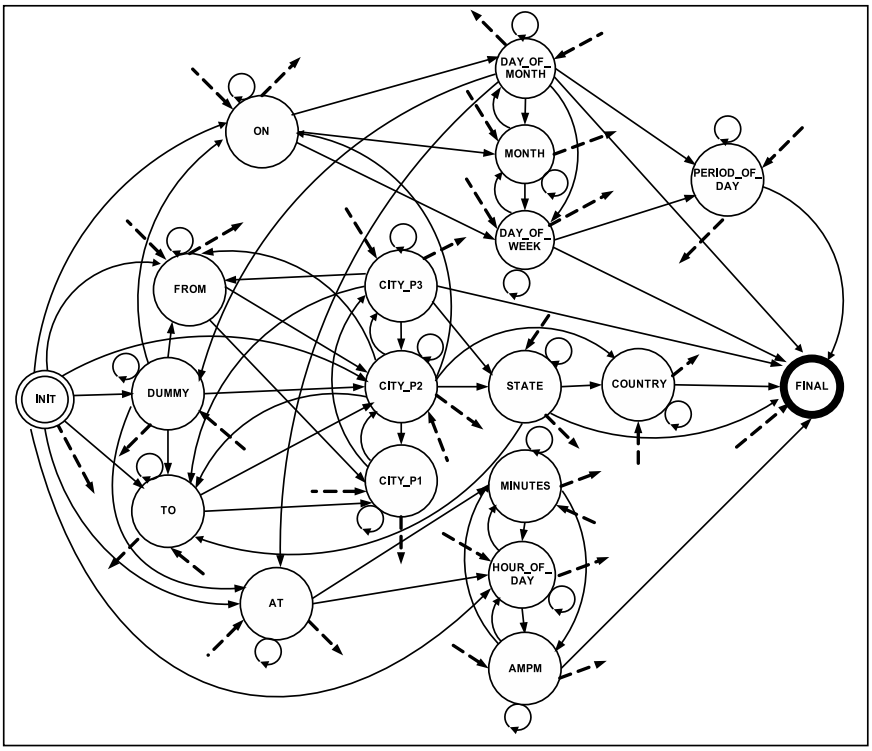

Figure 6.1: A partial network depicting the initial flat-concept semantic model

The emission probabilities are initialized by classifying the words in the system's vocabulary into the known set of semantic classes manually such that a set of words belonging to a semantic class are initially equiprobable.

As can be seen in Figure 6.1, the INIT and the FINAL states mark the beginning and end of the HMM and are non-emitting. The dotted arrows represent those transitions to

and from other states that are not shown in the diagram. To account for the effects of spontaneous speech such as stammering, hesitation, etc. and to allow multiple observations from the same state self-loops are initially permitted for all emitting states.

The state transitions for the following utterances can be easily traced in Figure 6.1.

- (I would like to fly) DUMMY (from) FROM (Los) CITY_P1 (Angeles) CITY_P2 (California) STATE (to) TO (Boston) CITY_P2 (on) ON (May) MONTH (first) DAY_OF_MONTH (at) AT (nine) HOUR_OF_DAY (p. m.) AMPM
- (Monday) DAY_OF_WEEK (June) MONTH (fifteen) DAY_OF_MONTH (early) PERIOD_OF_DAY (in the) DUMMY (morning) PERIOD_OF_DAY

The initial model is too unconstrained to be of practical use. Therefore, it is essential to introduce some informative structures by prohibiting arbitrary and unlikely state transitions based on prior domain knowledge and training examples. For instance, in order to disambiguate words belonging to multiple semantic classes, some unlikely transitions can be explicitly prohibited. For example, "twenty six" in "May twenty six" would be labeled as DAY_OF_MONTH if the unlikely one-step transitions from MONTH to other confusable states such as FLIGHT_NUMBER, ID_NUMBER, QUANTITY, HOUR, MINUTES, etc. are explicitly prohibited. This may require several iterations of testing on a training set in order to learn semantic structures from unannotated data. To simplify this process, we implemented a model compiler that allows us to modify the model parameters easily using a modeling language in order to generate a better initial model where the transition probabilities are tuned based on prior domain knowledge.

The initial transition probabilities can be tuned as required using the keywords "all", "high", "low", "only", "except", and "none". Tuning, in this context, is the process of introducing constraints to modify the initial model structure using these keywords. An excerpt of the model definition for the flat-concept model is given in Figure 6.2.

```
INIT
-> except{FINAL}
...
CITY_P1
-> except{STATE} high{CITY_P2}
"city_p1.txt"
CITY_P2
-> except{CITY_QUALIFIER}
"city_p2.txt"
CITY_P3
-> all
"city_p3.txt"
...
DAY_OF_MONTH
-> except{FLIGHT_NUMBER,QUANTITY} high{DAY_OF_MONTH}
"day_of_month.txt"
...
TO
-> except{DUMMY} high{CITY_P1,CITY_P2,HOUR_OF_DAY}
"to.txt"
...
FINAL
-> none
```

Figure 6.2: An excerpt of model definition for the flat-concept model

The keyword "all" means that all transitions out of a state are equally likely including self-loops and "none" means no transition out of a state is possible (e.g. out of the FINAL state). The keyword "except" as in ->except{FINAL} means that a one-step transition to FINAL is prohibited while allowing all other transitions to any other state. The keyword "only" is used to specify the only allowable transition(s) out of a state. The keyword "high" assigns to a specified set of states a transition probability value which is twice as much as the rest of the probabilities in that class and "low" assigns half as much. The entries "city_p1.txt", "city_p2.txt", "day_of_month.txt", etc. in Figure 6.2 are simple text files that contain the lexical items belonging to the sub-concepts CITY_P1, CITY_P2, DAY_OF_MONTH, etc., respectively.

After a number of testing and tuning parameters on the training data, the initial model becomes "good" enough for EM training. The training algorithm starts with the HMM consisting of the "tuned" parameters as initial values and iteratively refines the model until convergence is reached using the training data.

6.5.1 Limitations of the Flat-Concept Model

The flat-concept model built in this way yields satisfactory performance for relatively simple application domains. A detailed discussion of the experiments carried out and the results obtained will be presented in Section 8.6.3.

However, it suffers from the following main limitations:

- The output is less expressive as a single concept is often broken down into a sequence of low level concepts that are individually labeled. For instance, phrases like "Tuesday November the eleventh two thousand nine" or multi-word city names such as "New York City", etc. would be more informative if labeled as DATE and CITY, respectively instead of simply providing a fragmented output containing each word along with a corresponding atomic semantic label.
- Its predictive power is very weak as adjacent semantically related concepts are loosely coupled.
- It does not allow any hierarchical grouping of concepts and the encoded context is quite narrow.
- Several iteration of testing and hand-tuning of model parameters may be required on either a training or a development set before sufficient coverage can be achieved.

These shortcomings of the flat-concept model motivate a need for models like the ones we propose in this thesis where we capture longer context by grouping semantically and hierarchically related low-level concepts into higher level structures. In this thesis, we propose two approaches that target different levels of hierarchical organization of concepts. In the following section, we describe a model that allows medium-level hierarchical organization of concepts by grouping semantically related low-level concepts together so as to encode longer contextual information.

6.6 The Medium-level Hierarchical Model

In order to encode longer context, we use the detailed list of concepts that have been identified in Section 6.5 and we group two or more low-level concepts that describe a

single semantic concept into a single cohesive unit. For instance, low-level concepts HOUR_OF_DAY, MINUTES, AMPM describe a concept TIME, and low-level concepts like MONTH, DAY_OF_MONTH, DAY_OF_WEEK and YEAR represent a single concept DATE, etc. As can be clearly seen, the knowledge required to determine which attributes should belong together to form such a structure is a commonplace knowledge.

Accordingly, we identified 18 cohesive units containing semantically related low-level concepts for the domain of airline travel planning and 11 for the domain of train inquiries. Listing 5 depicts a partial list of groups comprising a set of low-level concepts (attributes) in the domain of airline travel planning.

Listing 5 Partial list of grouped semantic concepts

```
CITY: (CITY_P1, CITY_P2, CITY_P3, SPELT_CITY)
AIRPORT: (AIRPORT_NAME, AIRPORT_TYPE, AIRPORT_QUALIFIER, SPELT_AIRPORT)
DATE: (DAY_OF_MONTH, DAY_OF_WEEK, MONTH, YEAR)
TIME: (MINUTES, HOUR_OF_DAY, AMPM)
AIRLINE: (AIRLINE_QUALIFIER, AIRLINE_NAME)
CAR_INFO: (CAR, RENTAL_COMPANY, CAR_TYPE)
FLIGHT_INFO: (FLIGHT_CLASS, FLIGHT_NUMBER, FLIGHT_TYPE, FLIGHT_QUALIFIER)
HOTEL_INFO: (HOTEL_TYPE, HOTEL_QUALIFIER, LOCATION)
USER: (ID,ID_NUMBER,NAME_OF_USER)
PRICE: (FARE, AMOUNT_OF_MONEY, FARE_CLASS)
```

The rationale behind grouping of related sub-concepts together is threefold. First, it improves the predictive power of the model since adjacent related concepts are well coupled. Second, the model produces outputs that are semantically rich and more meaningful. Third, it offers better ambiguity resolution power than the flat-concept model. For instance, "twenty six" in "May twenty six" would not be confused with other possible semantic labels such as MINUTES, QUANTITY, ID_NUMBER, etc. as DATE is a single entity where the attributes DAY_OF_MONTH and MONTH are well coupled. Hence, the model resolves ambiguities of this sort which otherwise had to be manually tuned.

Each of these semantically organized units is modeled as a sub-network with two non-emitting states that mark the entry and exit states of the sub-network. The transitions between the states within a sub-network are initially ergodic which will later be refined through tuning and training. The non-emitting states of sub-networks are used to glue a

sub-network with other sub-networks and states in the global network. The initial global HMM for the medium-level hierarchical model for each application domain is a fully connected network such that any state or sub-network can follow any other state or sub-network with equal probability. The global network has two more non-emitting states INIT and FINAL that mark the entry and exit states of the network. A one-step transition from the entry state to the exit state is explicitly prohibited to prevent non-emitting loops.

Figure 6.3 shows a partial structure of the HMM for the domain of airline travel planning. The dotted arrows represent the transitions to and from states and sub-networks that are not shown in the diagram.

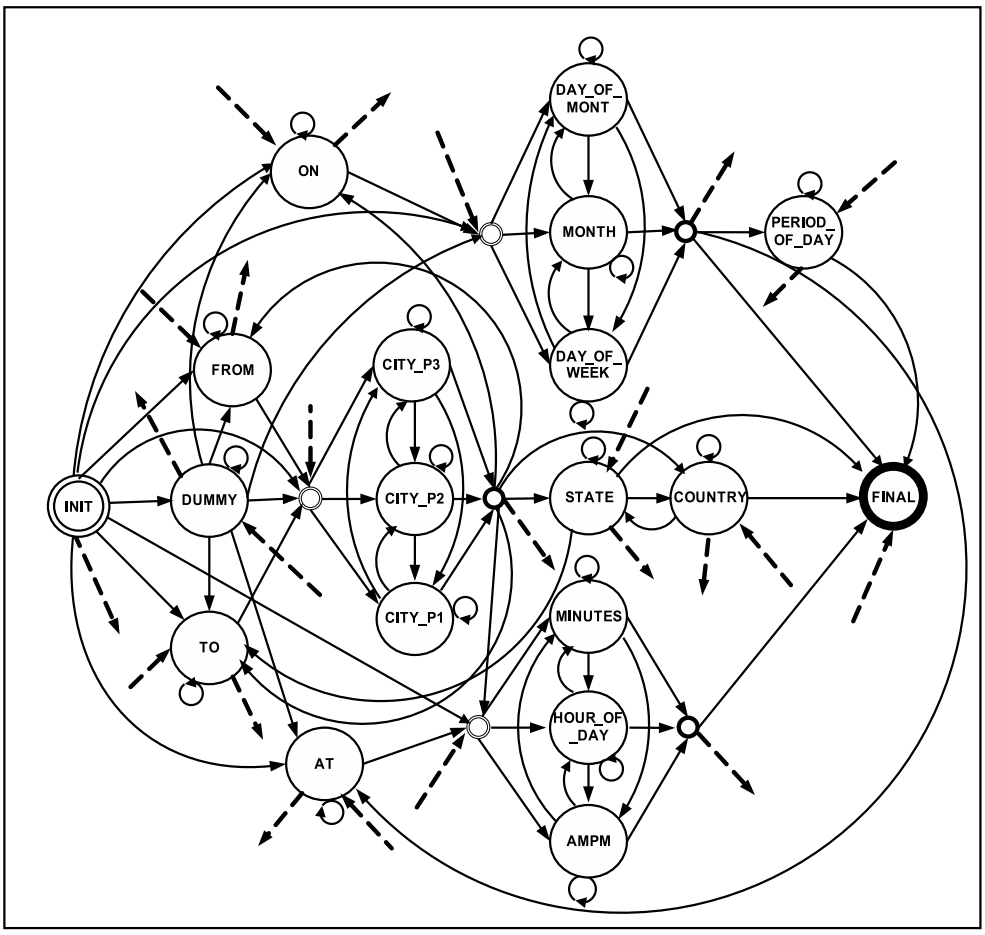

Figure 6.3: A partial structure of the initial medium-level hierarchical model

It is easy to trace the state transitions for the following utterances in Figure 6.3.

- (I would like to fly) DUMMY (from) FROM (Los Angeles) CITY (California) STATE (to) TO (Boston) CITY (on) ON (May first) DATE (at) AT (eight thirty p. m.) TIME
- (Monday June fifteen) DATE (early morning) PERIOD_OF_DAY

The emission probabilities are initialized in the same way as in the flat-concept model – by classifying the words in the vocabulary of the application domains into the known set of lexical classes where all words belonging to a semantic class are initially set equiprobable.

Once we define the model structure, it may be necessary to bias the initial transition probabilities of the HMM to help the disambiguation of some lexical items that belong to multiple semantic classes which the modeling approach could not resolve on its own. This can be done by performing preliminary tests on the training or a development data and introducing necessary constraints as required until the training data is sufficiently covered. To provide easy tuning and to keep the cost of tuning low, we extended the model compiler introduced in Section 6.5 so that it accommodates the new modeling approach. An excerpt of the model definition for the medium-level hierarchical model is shown in Figure 6.4.

As described in the previous section the initial model transition probabilities can be easily tuned as required using the keywords "all", "high", "low", "except", "only" and "none". As a convention, the entry state of a sub-network is denoted by the name of the concept (e.g. CITY) and the exit state is denoted by a tilde followed by the name of the concept (e.g. ∼CITY).

Given a "well-informed" initial model tuned as described above, the EM algorithm can be used to further refine the model parameters.

The data used, the experiments carried out, the results obtained and some illustrative examples for the medium hierarchical model will be discussed in Section 8.6.4.

The medium-hierarchical model offers better ambiguity resolution ability and produces more structured output than the flat-concept model. However, it can further be extended to encode more hierarchical relationship of concepts. For instance, the concepts FROM and TO have strong hierarchical relation with the concepts CITY and AIRPORT and when combined represent higher-level semantic information such as DEPARTURE_LOC and ARRIVAL_LOC which were unavailable in the models discussed above.

```
INIT
-> except{FINAL}
...
CITY
{
        CITY
        ->except{~CITY}
        CITY_P1
        ->all high{CITY_P2} low{~CITY}
        “city_p1.txt”
        CITY_P2
        ->all high{CITY_P3,~CITY} low{CITY_P1}
        “city_p2.txt”
        CITY_P3
        ->all high{CITY_P3, ~CITY}
        “city_p3.txt”
        SPELT_CITY
        ->all high{SPELT_CITY}
        “spelt_city.txt
        ~CITY
        ->none
}
->except{INIT} high{AIRPORT, STATE, COUNTRY}
DATE
{
        DATE
        ->except{~DATE}
        MONTH
        ->all high{DAY_OF_MONTH}
        “month.txt”
        DAY_OF_MONTH
        ...
        ~DATE
        ->none
}
->except{INIT}
...
TO
->except{INIT} high{CITY,AIRPORT}
“to.txt”
...
FINAL
->none
```

Figure 6.4: (Mengistu et al., 2008a): A partial model definition for the medium-level hierarchical model

6.7 The Hierarchical Model

In this section we extend the modeling approach discussed in Section 6.6 further by including hierarchically related concepts into similar cohesive units discussed in the previous section. The hierarchical extension of the model provides additional, higher-level semantic concepts such as DEPARTURE, ARRIVAL, etc. information and hence provides a richer semantic output.

The modeling approach described in this section involves two stages. First, as described in Section 6.5 we define a detailed ontology of each application domain by identifying the relevant concepts and their interrelationships, then we group semantically and hierarchically related concepts together into units called super-concepts. For example, the high-level concept TIME can further be put into a higher-level entity called ARRIVAL_TIME or DEPARTURE_TIME. This kind of structure can be readily produced by a dialog designer of a given application domain using domain knowledge and training examples.

The initial global HMM consists of an ergodic network of sub-networks and single state nodes similar to the one described in Section 6.6. An example sub-network (LOCATION) that can represent phrases like "Washington Dulles International Airport", "Los Angeles California", etc. is shown in Figure 6.5.

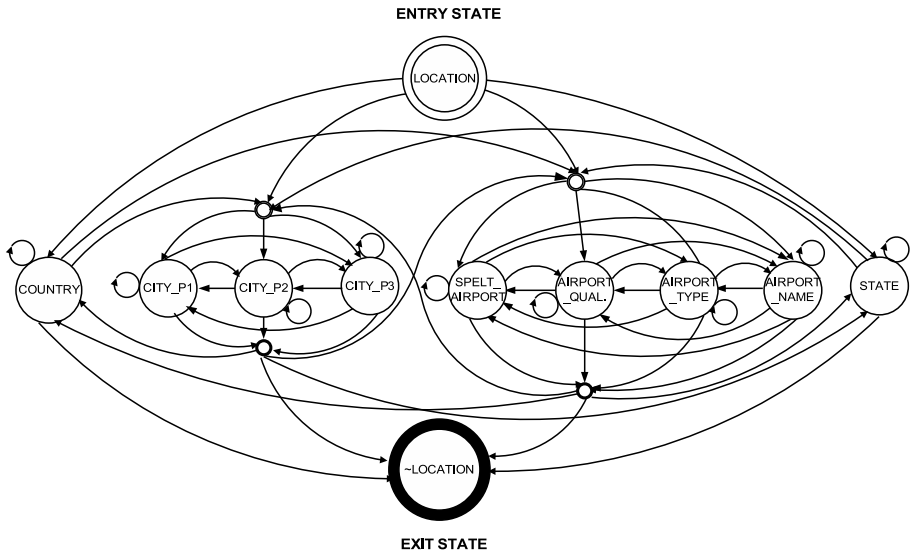

Figure 6.5: (Mengistu et al., 2008b): A sub-network (LOCATION) that contains single state concepts (COUNTRY and STATE) and sub-networks (CITY and AIRPORT)

As can be seen in Figure 6.5 a sub-network (e.g., LOCATION) can contain other sub-networks (e.g., CITY and AIRPORT). When sub-networks are used in many other bigger sub-networks the model gets more complex. In order to deal with the consequent data sparsity problem sub-networks can be tied to enable sharing of emission and internal transition probabilities. For instance, the multi-word concept CITY is expected to have the same internal transition and emission probabilities regardless of whether it is in a departure location (DEPARTURE_LOC) or arrival location (ARRIVAL_LOC). When sub-networks are tied, all the data that would have been used to estimate the individual untied parameters are pooled together to estimate the parameters of the tied sub-network. We refer to these kinds of tied sub-networks that are defined once and reused in different super-concepts as macros. For example, LOCATION_MACRO can be used in ARRIVAL_LOC, DEPARTURE_LOC or as a self-contained entity LOCATION.

As is the case with the other modeling methods, biasing some transition probabilities may be necessary with the help of domain and linguistic knowledge to obtain better initial models. This process is simplified with the use of an extended version of the modeling language described in Section 6.5 where one can easily modify the initial transition probabilities with the keywords "none", "except", "high", "low", "only" and "all". Accordingly, the model compiler is extended to accommodate the new features. Figure 6.6 depicts an excerpt of the hierarchical model definition for the airline travel planning domain.

In Figure 6.6, CITY_MACRO and AIRPORT_MACRO are sub-structures that hold semantically related information together. LOCATION_MACRO is an example of a nested macro containing other smaller macros – AIRPORT_MACRO and CITY_MACRO. As a convention, the entry state of a sub-network is denoted by the name of the sub-network itself (e.g. CITY_MACRO) and the exit state is denoted by $\sim$ followed by the macro name (e.g. $\sim$CITY_MACRO). The notation "->return" marks the end of a macro definition. To use a macro in another sub-structure we use a notation like LOC_CITY=>CITY_MACRO to mean CITY_MACRO is reused as LOC_CITY. A one-step transition to the exit state of a sub-network directly from the entry state is explicitly prohibited with constraints like "->except{$\sim$CITY_MACRO}" to prevent non-emitting loops. As described in Section 6.5, "city_p1.txt", "city_p2.txt", "arrival.txt", etc. are simple text files that contain the lexical items belonging to the sub-concepts CITY_P1, CITY_P2, ARRIVAL, etc. The required additional human effort and expertise to design

```
INIT
-> except{FINAL}
=>CITY_MACRO
   {
        CITY_MACRO
        -> except{~CITY_MACRO}
        CITY_P1
        -> all high{CITY_P2} low{~CITY_MACRO}
        "city_p1.txt"
        CITY_P2
        -> all high{CITY_P3,~CITY_MACRO} low{CITY_P1}
        "city_p2.txt"
        CITY_P3
        -> all high{CITY_P3,~CITY_MACRO}
        "city_p3.txt"
        ~CITY_MACRO
        -> none
   }
   -> return
=>AIRPORT_MACRO
   {
        AIRPORT_MACRO
        ->except{~AIRPORT_MACRO}
        ...
        ~AIRPORT_MACRO
        ->none
   }
   ->return
=>LOCATION_MACRO
   {
        LOCATION_MACRO
         -> except{~LOCATION_MACRO}
         LOC_CITY=>CITY_MACRO
         -> all high{STATE,COUNTRY}
        LOC_AIRPORT=>AIRPORT_MACRO
        ->all
        STATE
        -> all high{STATE,~LOCATION_MACRO}
         "states.txt"
        COUNTRY
         -> all high{~LOCATION_MACRO}
        "countries.txt"
        ~LOCATION_MACRO
        -> none
   }
-> return
ARRIVAL_LOC
   {
        ARRIVAL_LOC
        -> only{ALOC_ARRIVE,ALOC_TO}
        ALOC_ARRIVE
         -> except{~ARRIVAL_LOC}
        "arrival.txt"
        ...
        ALOCATION=>LOCATION_MACRO
        -> only{~ARRIVAL_LOC}
        ~ARRIVAL_LOC
         -> none
}
-> all
...
FINAL
->none
```

Figure 6.6: (Mengistu et al., 2008b): Excerpt of model prototype for the domain of airline travel planning

the described hierarchical model is clearly much less than would be required to semantically annotate the training data which would also require detailed analysis of the application domains to define semantic labels and organize them into super-concepts. Besides, the cost of producing such an organization is much less than hand-crafting a semantic grammar as no particular linguistic (syntactic and semantic) expertise is required – only a commonplace prior domain knowledge is assumed.

Another robust feature of the hierarchical model is that it can label out-of-vocabulary words (unseen vocabulary items in the training data) correctly. The example in Listing 6 illustrates this clearly. Given that city name "Addis Ababa" and airline name "Ethiopian" were not seen in the training corpus, the utterance "I'm looking for a flight from Addis Ababa to Frankfurt Germany on September the twenty third late in the afternoon on Ethiopian airlines" would be labeled as shown in Listing 6.

Listing 6 Example output of the hierarchical model

```
(I'm looking for a) DUMMY (flight) FLIGHT_INFO
(from oov[Addis] oov[Ababa]) DEPARTURE_LOC
(to Frankfurt Germany) ARRIVAL_LOC
(on September the twenty third) DATE_INFO
(late in the afternoon) TIME_INFO
(on oov[Ethiopian] airlines) AIRLINE
```

The notation oov[...] in Listing 6 denotes an out-of-vocabulary word. As can be seen, in addition to providing a more useful and structured output, the hierarchical model can correctly label out-of-vocabulary words using the surrounding context. Moreover, the output can also be made to show the low-level concepts within the high-level structure. For instance, the detailed output for the DEPARTURE_LOC in the above example would look like:

Listing 7 Sample detailed output of the hierarchical model

```
((from) FROM (((oov[Addis]) CITY_P1 (oov[Ababa]) CITY_P2) CITY_MACRO)
 LOCATION_MACRO) DEPARTURE_LOC.
```

In Section 8.6.5 we will present more illustrative examples and a detailed discussion of the data used, the experiments conducted and the results obtained for the hierarchical model.

6.7.1 Robustness to Noisy Data

Spoken language understanding can be easy for simple application domains where users are restricted in the choice of their formulation of a spoken request and the vocabulary size is very small. However, if a dialog system allows human-to-human like conversation the task becomes more challenging due to the natural phenomena of spontaneous speech such as hesitations, false starts, filled pauses, etc. that introduce undesirable noise. Hence, a spoken language understanding model must be robust to properly deal with these effects of natural language.

Listing 8 shows some noisy utterances from the DARPA Communicator corpus along with the corresponding output of the hierarchical model.

Listing 8 Example noisy utterances and the corresponding tagged output of the hierarchical model

```
no no no no not at six thirty I'd like to arrive at six thirty.

(no no no no not) NO (at six thirty) TIME_INFO (I'd like to) DUMMY (arrive
 at six thirty) ARRIVAL_TIME
```

```
United Airlines from Los Angeles to from to to London.

(United Airlines) AIRLINE (from Los Angeles) DEPARTURE_LOC (to) TO (from)
FROM (to to London) ARRIVAL_LOC
```

```
I'd like to arrive leave early morning.

(I'd like to) DUMMY (arrive) ARRIVAL (leave early morning) DEPARTURE_TIME
```

```
I'd like to fly from Los Las Vegas to oh never mind um Tucson to Las Vegas

(I'd like to fly) DUMMY (from Los Las Vegas) DEPARTURE_LOC (to oh oov[never]
 oov[mind] um) DUMMY (Tucson) LOCATION (to Las Vegas) ARRIVAL_LOC
```

As can be seen, each of the utterances in Listing 8 contain some form of undesirable noise – repetitions, self-repairs, interjections, etc. However, the hierarchical model can gracefully deal with these as can be seen in the examples. In fact, the output of the hierarchical model has to be analyzed within the context of a dialog state so that some ambiguities can be resolved internally or with explicit confirmations. This can help us to analyze only the relevant information at a given dialog state and safely ignore everything else. In the last example, for instance, there is a conflict; i.e., Las Vegas is labeled as both departure and arrival location. Hence, the dialog management program can confirm it explicitly (e.g. "Do you want to fly to Las Vegas or from Las Vegas?") or use some rules to resolve conflicts of this sort. Note also that the two out-of-vocabulary words (never, and mind) are correctly labeled in the last example. The output of a speech recognition engine could also be noisy, and hence the model should be robust to deal with recognition errors.

6.8 Summary

In the first three sections of this chapter, we introduced the spoken language understanding problem, reviewed related approaches to spoken language understanding and described HMMs as used in spoken language understanding. The remaining sections in this chapter describe our HMM-based approach to semantic concept labeling at three different but interrelated levels. We started with the conventional flat-concept approach and incrementally extended it to encode more context at different levels of hierarchy. The hierarchical models offer better ambiguity resolution ability, higher predictive power, and produce more structured, semantically richer information than the flat-concept model. Moreover, the hierarchical models are more robust to the effects of natural language than the conventional flat-concept model. It has also been shown that out-of-vocabulary words could be more correctly labeled with the hierarchical model than with the flat-concept model.

All the described approaches can be readily trained on unlabeled data with relatively less human supervision. We also introduced a modeling language and a model compiler that essentially minimize the required level of human effort by allowing users to easily tune the initial model parameters. Tuning of initial model parameters using prior domain knowledge and training examples is essential to provide a well-informed initial model to the EM algorithm. The effort is shifted from the laborious and error-prone manual

semantic labeling of the full training corpus to manually designing semantic models at a required level of hierarchy using ones prior domain knowledge and training examples. The detailed analysis of the application domains to define semantic labels and organize them into super-concepts must be done anyway, even for manually labeling the training data. Hence, the required additional effort to design the hierarchical model is considerably low.

Chapter 7

Spoken Language Interaction

7.1 Introduction

As discussed in the previous chapters, the performance of a telephone-based spoken dialog system depends on the performance of the underlying technologies that include automatic speech recognition, spoken language understanding, text to speech synthesis and the telephony interface. Dialog quality and dialog management strategy are equally important as the overall usability and acceptability of any dialog system mainly depends on these. In the preceding chapters we have discussed the general architecture, the automatic speech recognition and the spoken language understanding components of our telephone-based spoken dialog system framework. In this chapter we describe the fundamentals of spoken language interaction and related issues.

The rest of the chapter is organized as follows. In Section 7.2, we describe spoken language interaction and the essential characteristics of a dialog. Dialog management issues are discussed in Section 7.3 followed by the discussion of dialog design principles in Section 7.4. VoiceXML and VoiceXML-based spoken dialog systems in general are reviewed in Section 7.5. In Section 7.6, we discuss the dialog system evaluation procedure we used to evaluate the quality and usability of our framework.

7.2 Spoken Language Interaction: Overview

Historically, research in spoken language interaction has followed two main lines of thought; namely, discourse analysis and conversation analysis. Discourse analysis (Searle, 1976), views dialog as a rational cooperation and assumes that utterances in a conversation are well-formed sentences and ignores the disfluencies of spoken language. Conversation analysis (Levinson, 1983; Sacks, 1992), on the other hand, views dialog as a social interaction in which the phenomena of spontaneous speech such as hesitations, false starts, filled pauses, abrupt shift of focus, etc. are taken into consideration. As can be noted, an ideal spoken dialog system should, in principle, consider all the disfluencies that are inherent in natural conversation. However, in practice, the design of a dialog system is constrained by the limitations of the underlying technologies such as automatic speech recognition, speech understanding, speech synthesis, etc. Hence, generally man-machine interactions are more constrained and less complex than human-to-human communications.

Spoken language interaction is a complex joint activity characterized by turn taking, speech acts, grounding, shift in dialog initiative and other discourse phenomena such as ellipsis, anaphora, etc. In the sections that follow, we describe these features briefly.

7.2.1 Turn-taking

Turn-taking (Sacks et al., 1974) is a fundamental organization of conversation by which participants alternate in "taking the floor". A dialog participant in a human-to-human communication signals the desire to take, maintain or yield a turn by a verbal or non-verbal signal, such as eye contact, a pause or another means. Human-to-human communication appears to be very permissive, with a rich array of turn-taking cues (Baber, 1993). Back-channels such as "hmm", "uh-huh", "yeah", etc. are often used to indicate that the listener is actively listening and encourage the speaker to continue talking. One approach to manage turn-taking in man-machine interaction is pairing the utterances of the dialog partners where the first part of the pair requires the second part of the pair for a meaningful interaction to occur. These pairs are known as adjacency pairs (Schegloff and Sacks, 1973). Adjacency pairs are two subsequent matching utterances produced by different speakers constituting a dialog exchange such as question-answer, greeting-greeting, statement-acknowledgement, etc.

7.2.2 Speech Acts

Each turn or utterance in a dialog is equivalent to an action being performed by the speaker (Austin, 1962). An utterance can change the state of the world as in "I now pronounce you husband and wife" which results in a new social reality. Speech act theory describes an utterance in a dialog at three levels; namely, locutionary, illocutionary and perlocutionary acts. A locutionary act is simply the act of uttering a meaningful utterance. An illocutionary act, on the other hand, is the real action performed by the utterance as in asking, welcoming, informing, apologizing, warning, etc. A perlocutionary act is the effect(s) of the utterance on the listener, who is, for instance, welcomed, informed, or warned. The term speech act is generally used to describe illocutionary acts rather than either of the other two (Jurafsky and Martin, 2008).

7.2.3 Grounding

As a dialog is a collaborative process to perform a common task, it is necessary that the participants establish a common ground (Stalnaker, 1978) so that possible misunderstandings can be repaired early and the dialog participants get evidence that their intention is understood by each other. The listener must somehow make it clear (ground) that the speaker's intention is understood. Clark and Schaefer (1989) introduce a concept of contribution which has two phases – presentation and acceptance. In a spoken dialog system, the speaker (the user) presents an utterance in the presentation phase. In the acceptance phase the listener (the system) has to ground explicitly or implicitly to indicate whether correct understanding has been achieved.

7.2.4 Dialog Acts

Speech acts do not model the key features of conversations such as grounding, contributions, adjacency pair, etc. (Jurafsky and Martin, 2008). Therefore, a higher level concept that bears the relationship of an utterance with the neighboring dialog turns is used. This construct is known as a dialog act (Bunt, 1994). It indicates the function of an utterance in a given dialog. Dialog acts are a finite set of labels applied to utterances in a discourse such as YES-NO QUESTION ("Would you like to fly on May first?"), REQUEST ("I

would like to fly from Hartford to Boston on May first"), OPINION ("I think the computer is not listening to me"), INTRODUCTION ("My name is KEY"), BACKCHANNEL ("uh-huh").

7.2.5 Ellipsis and Anaphora

In natural language dialogs, utterances normally contain discourse phenomena known as ellipsis and anaphora. Anaphora is the phenomenon of a linguistic expression used when a speaker wants to refer back to something mentioned earlier in the conversation. For instance, in the sequence of utterances "I am looking for flights to Boston" "I want to arrive there early in the morning", the adverb 'there' in the second utterance refers to 'Boston' which is mentioned in the preceding utterance. The process of associating 'there' with 'Boston' in the above example is known as anaphora resolution.

Ellipsis is a fragment of a sentence where a word or a phrase is left-out and the missing part should be inferred or extracted from previous utterances or context. For instance, in the fragment "Are there any to San Francisco?", one has to fill in the missing word to get what is meant from dialog context and general knowledge. In the above example it appears that 'flights' might be the missing word if the discourse history was about flights. As can be inferred, to resolve ellipsis and anaphora, it is essential to keep the discourse history and use them to resolve these discourse phenomena.

7.3 Dialog Management

Dialog management provides a lucid overall structure to a spoken interaction that goes beyond a single turn and properly manages a spoken language interaction between the dialog participants.

7.3.1 Dialog Initiative

Dialog initiative refers to who has the conversational lead in a dialog (Walker and Whittaker, 1990) or who is in control of the dialog. In human-to-human conversation each participant may alternately own initiative to direct the flow of the conversation. This type

of interaction where a shift in dialog initiative can take place from one participant to the other in the course of a dialog freely is referred to as mixed-initiative.

In man-machine interaction, the initiative does not change between the system and the user as freely as it does in human-to-human conversations. In many implemented commercial systems, the system owns the conversational lead and directs the dialog by asking questions to elicit information from the user and the role of the user in the dialog is limited to responding to the system prompts. These kind of systems are referred to as system-initiative or system-directed. An example of system-directed dialog is presented below.

```
System:  What is your departure city?
User:    London
System:  What is your arrival city?
User:    Berlin
System:  On what day do you wish to travel?
User:    tomorrow
System:  At what time?
User:    nine p. m.
```

System-directed dialogs have a virtue of simplicity. Since the vocabulary and grammar at any point in a dialog are very small and can be specified in advance, the speech recognizer and the semantic interpreter can be more accurate. The added accuracy and simplicity of the dialog makes them particularly suitable for novice users. However, such a dialog structure is not natural, puts grave restriction on the user, and is not suitable for frequent users as the dialog advances very slowly.

On the other extreme are user-directed systems where the user always has control over the conversation and the system passively answers the user questions. User-initiative systems are difficult to realize as they require comprehensive speech recognition and language understanding capability in order to understand requests from a potentially wide spectrum of users and application domains. The system should also have the ability to keep record of the previous utterances to answer elliptical and anaphoric questions that may follow. An example of user-directed dialog is given below.

```
User:    How many flights are there today?
System:  Fifty five
```

```
User:    How many of them are departing from Boston?
System:  Five
User:    And to San Francisco?
System:  Seven
```

Mixed initiative systems, in principle, should allow shifts of initiative between the user and the system freely. However, most implemented "mixed-initiative" systems use the term in a restricted sense – that is, the system maintains the control of the dialog but the users are given some freedom to give more than the requested information at a time. For instance:

```
System:  Please tell me your departure and destination?
User:    I want to fly from London to Berlin on next Tuesday
System:  Next Tuesday, at what time?
User:    nine p. m.
```

As can be noted, mixed-initiative systems can use system-initiative strategy while giving the users more freedom of expression. The dialog strategy we employ adopts the latter approach and attempts to give the user freedom to provide more information in a single utterance and the system guides the user to fill missing information. A typical dialog scenario is given below:

```
System: What is your travel plan?
User: I want to fly from Hartford to Boston on May twenty first
System: Would you like to fly from Hartford to Boston?
User: Yes
System: At what time would you like to fly on May first?
User: No, I want to fly on May twenty first early in the morning
System: Would you like to fly on May twenty first early morning?
User: Yes
System: Do you have any airline preference?
User: No
System: OK. Just a moment ....
```

7.3.2 Dialog Control

Dialog control strategies specify the dialog flow logic in spoken dialog systems. The commonly used dialog control strategies include finite-state based, frame-based, plan-based (Allen, 1979; Cohen and Perrault, 1979) and information-state based (Traum and Larsson, 2000) approaches. In practice, finite-state and frame-based approaches are the most commonly used ones mainly because of their relative simplicity. Table 7.1 shows the dialog control strategy commonly used (✓) for the various types of initiatives discussed in Section 7.3.1.

Table 7.1: Initiative and dialog control strategy

	Initiative		
Dialog Strategy	System	User	Mixed
Finite State	✓	-	-
Frame-based	✓	-	✓
Information-state	-	✓	✓
Plan-based	-	✓	✓

Finite-state based dialog control is the most straight forward dialog control approach where a dialog is expressed as a network of nodes connected by arcs. The nodes represent the dialog states and the arcs represent the transitions between the dialog states as shown in Figure 7.1.

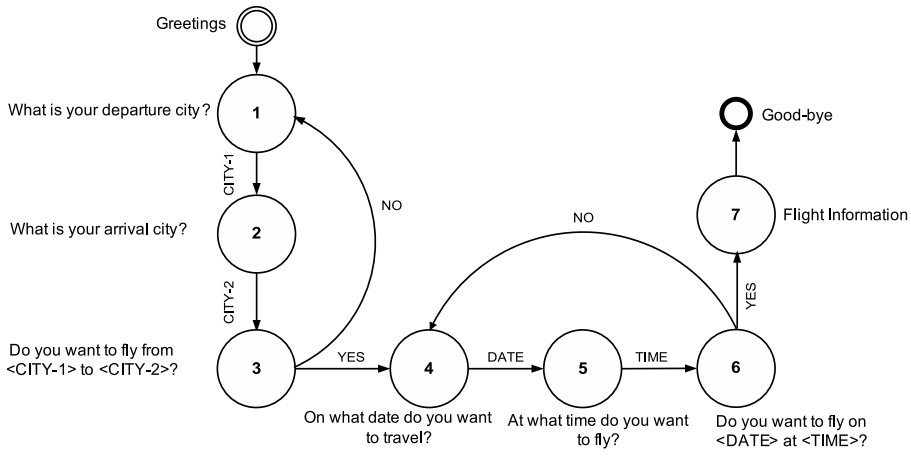

Figure 7.1: An example finite-state based dialog control architecture

Despite its simplicity, the approach quickly becomes inefficient when the number of dialog states and the transitions between the states increases as the model rapidly becomes

unmanageable. Nevertheless, finite-state based control remains to be suitable for small-scale, well-structured and system-initiative applications.

Frame-based systems are based on the slot-filling concept where slots are filled with information that is elicited from the user. The required information is fixed but the dialog flow is flexible. The system asks the user a series of questions to gather information that are required to fill one or more dialog slots. When all the required information are obtained, the system performs a database query or other required operation. Frame-based approach allows filling multiple dialog slots from a single utterance giving the user some degree of freedom to formulate his/her request in any order. However, the amount of dialog context that can be encoded is still limited, and it is difficult to model more complex conversations. Human communication over the telephone lacks the richness of face-to-face communication and is often task-oriented and limited to short phrases (Fielding and Hartley, 1987). Therefore, frame-based approach appears to be adequate for telephone-based and task-oriented interactions. The dialog control strategy we use in the two demonstration application domains is mainly frame-based which is well-suited to carry out medium-sized interactions and allows us to achieve limited "mixed-initiative" dialog capability. In some cases, when task-completion is at risk, we fall back to the system-initiative dialog strategy and solicit information piece by piece, one at a time.

Even though the frame-based approach is good enough for domain-specific conversations, a more complex dialog control strategy such as the information state model (Traum and Larsson, 2000) is required to extend a dialog system beyond a specific domain. Information-state based dialog control is built on an abstract concept known as an information-state. An information state contains a representation of the dialog history, the common ground of the dialog participants, the actions that can be taken next, etc. The information-state architecture also contains a set of dialog moves that trigger a set of update and selection rules and a control structure for deciding which update rules to apply at a given point. The update rules modify the information state of the system when the user produces an utterance and the selection rules select the next dialog move to be executed. For instance, when a question is recognized, an update rule may specify the need to answer the question; when a proposal is recognized an update rule may specify the need to update the information state with the new information and to perform grounding. As can be observed, it is necessary to decide if an utterance is a question, a proposal, a rejection, a suggestion, etc. given a recognized utterance. Our semantic concept labeling model

discussed in Chapter 6 can effectively decide if an utterance is a question, a request or a negation as in the following example and can be extended for use in an information-state framework.

```
(Do you have a) QUESTION (flight) FLIGHT_INFO (that) DUMMY
(leaves in the morning) DEPARTURE_TIME
```

Plan-based approach to dialog control is an advanced approach based on the plan-based theories of communicative action and dialog (Allen and Perrault, 1980; Allen, 1979; Appelt, 1985; Cohen and Levesque, 1990; Cohen and Perrault, 1979). The plan-based theories state that the speaker's speech act is part of a plan and that it is the listener's job to identify and respond appropriately to this plan (Bui, 2006). The approach is based on the view that a dialog is goal oriented and an utterance in a dialog is performing speech acts (Searle, 1976) to achieve these goals. The task of the agent listening to the utterance is, therefore, to discover the underlying plan of the speaker and react appropriately. For instance, an utterance "I want to attend a conference in Hamburg Germany from the fifteenth to the eighteenth of December" in an airline booking system could be interpreted by a plan-based agent as follows. The user wants to fly to Hamburg, and the departure date should be at least a day before the fifteenth and the return flight should be at least one day after the eighteenth of December.

7.4 Dialog Design

The dialog interface is the only way users can communicate with a spoken dialog system, hence interface design is one of the most important part of any speech-based application. A good interface contributes to the success of a spoken dialog system in that it enhances the user experience and the usability of the system.

Speech-based applications do not enjoy the flexibility and richness of web-based Graphical User Interfaces (GUIs) that can present a lot of information in parallel screens with a number of ways to enable easy navigation and selection of options. In GUI-based systems, the user can see a lot of items to get to the desired information easily. Moreover, users can initiate and terminate each step at their own pace. Simplicity, consistency and flexibility are other virtues of GUIs. However, speech is inherently sequential and some of the features we have in GUIs are not easily available. For instance, there's no way to

present more than one piece of information at a time, users would have to carefully listen to various lists, options, prompts, etc. before they can proceed to the next action and the amount of information that can be offered is limited by the amount of information that can be retained in the 'short-term memory' of the users. However, with careful design, spoken dialog systems can provide the required service with reasonable performance in a more natural way. Combining GUIs and speech interfaces can be more helpful, in some applications.

Shneiderman (1997) introduces the "eight golden rules" described below that can be used in the design of man-machine dialogs.

1. **Strive for consistency**: The required actions in similar situations should be consistent; the terminology used in prompts, menus, and help screens should be similar; and consistent commands should be used throughout.

2. **Enable frequent users to use shortcuts**: As the frequency of use increases, so does the user's desire to reduce the number of interactions and to increase the pace of interaction. Hence, the use of shortcuts is handy for expert users.

3. **Offer informative feedback**: For every user action, there should be some system feedback. For frequent and minor actions, the response can be modest, while for infrequent and major actions, the response should be more informative.

4. **Design dialog to yield closure**: Each sequence of actions should be organized into a group with a beginning, middle, and end. The informative feedback at the completion of a group of actions is important because it gives the user a satisfaction of accomplishment and a sense of relief.

5. **Offer simple error handling**: As much as possible, design the system to prevent serious errors. However, since errors are inevitable in spoken dialog systems, the system should be able to detect the error and offer simple, comprehensible mechanisms for handling the error.

6. **Permit easy reversal of actions**: Let the user know that errors can be undone and actions are reversible. This feature encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data entry, or a complete group of actions.

7. **Support internal locus of control**: Design the dialog to make users think that they are in control of the conversation and the system responds to their actions not the other way round.

8. **Reduce short-term memory load**: The limitation of human information processing in short-term memory requires that outputs be kept simple and consolidated.

Another set of principles related to the ergonomic design of dialog between user and interactive systems is the ISO 9241 part 110 (ISO9241-110, 2006) that is summarized below.

1. **Suitability for the task**: A dialog is suitable for a task when it supports the user to complete the task effectively and efficiently.

2. **Self-descriptiveness**: A dialog is self-descriptive if its users can tell which dialog and dialog-state they are in at anytime. The dialog should make it clear what the user should do next.

3. **Conformity with user expectations**: A dialog conforms with user expectations if it behaves according to the contextual needs of the user and avoids unexpected behavior.

4. **Suitability for learning**: A dialog is suitable for learning when new users can begin effective interaction easily and the system guides the user in learning to use the system.

5. **Controllability**: A dialog is controllable when the user is able to initiate and control the direction and pace of the interaction until the interaction goal has been achieved.

6. **Error tolerance**: A dialog is error-tolerant if the intended result can be achieved even with erroneous input with minimal corrective actions by the user.

7. **Suitability for individualization**: A dialog is capable of individualization when users can modify interaction and presentation of information to suit their individual capabilities and needs.

We have considered the above two sets of principles as guidelines in designing the dialogs in our system.

7.5 VoiceXML

As has been mentioned in Chapters 3 and 4, we use VoiceXML as a dialog scripting language in our application, hence we describe it further in this section. VoiceXML is an XML-based dialog scripting language standardized by the World Wide Web Consortium (W3C) to create speech interfaces mainly for telephone applications. A VoiceXML document contains a set of dialogs organized in a form or a menu.

A form consists of a set of form items – namely, input items and control items. Input items (e.g. <field>) are elements for collecting user input and control items (e.g. <block>) contain procedural items for audio output or computation. A menu presents a list of choices that the user can choose from and the links to the next dialogs to be executed. A simple VoiceXML dialog script based on forms is depicted in Listing 9.

A VoiceXML document is processed as described in the VoiceXML specification (W3C, 2004) by a VoiceXML interpreter. A VoiceXML interpreter implements, among other things, the Form Interpretation Algorithm (FIA) that specifies the procedure for walking through the various fields of a form to drive the interaction between the user and the system. The VoiceXML interpreter fetches the dialog scripts from the Web Server, accepts values extracted from the spoken input to fill various dialog states and determines what to do next according to the instructions in the dialog script. Depending upon the input received, the VoiceXML interpreter may load another dialog script from the Web Server or submit the collected information to the Web Server to query the database and present information back to the user. The VoiceXML interpreter may also issue basic telephony functions like <disconnect> and <transfer>. However, for advanced call control operations, the Call Control eXtensible Markup Language (CCXML) of W3C is commonly used.

The communication between the application server and the VoiceXML interpreter is via HTTP – the HTTP methods POST and GET are used to submit results obtained from the user and to request a new VoiceXML document.

Listing 9 Simple VoiceXML dialog script

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml" application="root.vxml">

<form id="intro">
<block>
<prompt>
 Hello! My name is KEY!
 I provide service in English and German! Which one do you prefer?
</prompt>
<goto next="#form_1"/>
</block>
</form>

<form id="form_1">
<field name="language">
<grammar src="http://localhost/language.slf" type="application/x-slf"/>
<filled>
<if cond="language == 'english'">
<prompt> Welcome to the Airline Travel Planning System of Magdeburg University!
</prompt>
<goto next="airline_main.vxml"/>
<elseif cond="language =='german'"/>
<prompt> <voice name="Katrin ">
Willkommen zum automatischen Bahnauskunftssystem der Universität Magdeburg!
</voice>
</prompt>
<goto next="http://localhost/german/train_main.vxml"/>
</if>
</filled>
</field>
</form>

</vxml>
```

In general, a VoiceXML-based application consists of:

- A telephony interface to deliver calls into the system
- A VoiceXML interpreter that executes the dialog by activating the ASR engine to collect spoken input, semantic interpreter to extract the meaning of a spoken utterance, TTS system to play prompts and responses, etc.
- An application server (typically, a Web server), where the application logic resides, and may contain interfaces to a database server
- For advanced call control functions, a CCXML interpreter is often used to process CCXML documents that specify the call control policy.

The architecture of our system has been described in more detail in Chapter 4, however, for quick reference we provide a simplified architecture of VoiceXML-based applications in Figure 7.2.

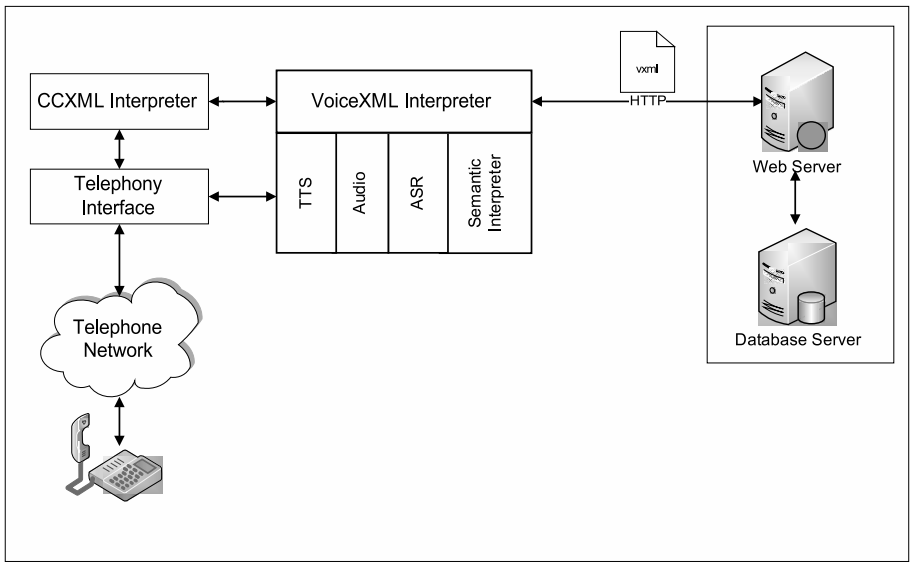

Figure 7.2: A simplified architecture of VoiceXML-based applications

One strong virtue of VoiceXML is that it is built around the existing web technologies, hence it is completely interoperable with many existing infrastructures (e.g. Web infrastructures), protocols (HTTP, TCP/IP) and standards (XML) that have made the Internet

ubiquitous. VoiceXML separates user interaction scripts from service logic. The server-side logic manages interactions with back-end applications such as database servers and creates VoiceXML documents as appropriate using standard Web development technologies. This feature also distinguishes VoiceXML from other proprietary spoken dialog environments that require special languages and application programming interfaces to access Web servers and external databases and applications. Moreover, VoiceXML provides features to support limited "mixed-initiative" dialogs in a frame-based architecture as described in Section 7.3.2. Detailed description of the W3C VoiceXML specification can be found at VoiceXML 2.0 Specification[1].

7.6 Evaluation

Spoken dialog system evaluation is a necessary step in order to assess the usability and quality of the system and it helps developers to identify problems that should be fixed to make the system more useful. The evaluation of spoken dialog systems is considerably more complicated and difficult than their graphical counterparts due to the unrealistically high expectation of users which follows their natural spontaneous daily experience in human-to-human communication. On the other hand, due to the limitations of the various technologies that constitute a spoken dialog system, there is often noticeable difference in performance between man-machine and human-to-human communication which may lead to less user satisfaction.

User satisfaction is one of the most important metric of spoken dialog system evaluation which is related to the quality and usability of the system. Quality is a compromise between what the user expects or desires, and the characteristics he/she perceives while using the system (Möller et al., 2007). Therefore, quality measurements can only be obtained from subjective judgements given by human users collected in a quantifiable form with questionnaires. However, the perceived quality of a system is influenced by usage environments (environmental factors), expertise of the user (agent factor), the complexity of the task offered by the system (task factors), as well as non-physical contextual factors (Möller, 2005).

[1] http://www.w3.org/TR/voicexml20/; last accessed February 27, 2009

Usability, as defined by ISO 9241, is the effectiveness, efficiency and satisfaction with which users accomplish tasks. Effectiveness refers to the output of the interaction while efficiency refers to the amount of resources expended to achieve the desired goal.

User satisfaction rating can be measured by inviting users to interact with the system to perform a task, and then asking them to complete one or more quantifiable post-interaction questionnaire(s) and average responses over all questions to get a total user satisfaction rating.

The SASSI (Subjective Assessment of Speech System Interfaces) (Hone and Graham, 2001) questionnaire for evaluating systems with speech input capability and the recommendation of the International Telecommunication Union (ITU-T) (ITU_T Rec. P.851) for evaluating telephone services based on speech technology are two popular de-facto standards for collecting user judgements on which we based our evaluation questionnaires. The questionnaires used in this evaluation are adapted from (Möller et al., 2007) and are given in Appendix B. Another well-known model for predicting quality judgements on the basis of collected interaction parameters is the PARADISE (PARAdigm for DIalog System Evaluation) model of AT&T (Walker et al., 2000).

The evaluation experiment we used consisted of four parts:

- A short oral presentation (for about 2 minutes) is given to each user about the dialog system and the purpose of the experiment.
- Each user fills an initial questionnaire through which general information about the test participants including their background knowledge and experience is solicited.
- The author calls the system and performs a demonstration interaction with the system to give the subjects some idea of what an interaction could look like, how recognition errors can be corrected, etc.
- Each user makes two calls to the system and performs two goal-oriented interactions – one in English and another in German. The English service is on airline travel planning and the German service is on Train information inquiries. After the interactions, the user is asked to complete a questionnaire that is designed to solicit the current impression of the user after using the system.

The subjects of the experiment are 20 native German speakers (10 male and 10 female) most of which are either students or employees of the University in the age range of 18-30. 50% of the subjects already had successful but unpleasant experience with other spoken

dialog systems in various application domains and most of the subjects (90%) have little knowledge about speech recognition and speech synthesis technology.

We prepared 10 English and 10 German dialog scenarios with brief task description as in the example below.

```
You live in Hartford [Connecticut] and you want to fly to Boston
[Massachusetts] to visit a friend. You want to fly on June fifteen
[2009] in the morning. Book a flight, if available, on that date and
time on United Airlines.
```

The exact formulation of the requests is left to the test users. The order and the number of relevant information a user wishes to convey in the first utterance is also not predefined. After the first request, the system guides the user to fill the missing dialog slots. However, the user can use commands like "help" or "start over" at any point, if necessary.

To complement the subjective evaluation, we also log each interaction and extract relevant interaction parameters such as number of user turns, number of system turns, number of correction turns, number of completed tasks, etc. that can describe the interaction quantitatively.

The detailed analysis of the results obtained using both subjective and objective evaluation are presented in Section 8.7.

7.7 Summary

In this chapter we presented an overview and the basic features of spoken language interactions such as turn-taking, grounding, speech acts, dialog acts and the common discourse phenomena known as ellipsis and anaphora. Besides, we discussed dialog initiatives, dialog control strategies and dialog design principles. The rationale for using VoiceXML and the dialog system evaluation method we adopted to evaluate the usability of our telephone-based spoken dialog system were also explained.

Chapter 8

Experiments and Discussion of Results

8.1 Introduction

In the previous chapters, we discussed various combination of techniques to achieve robustness in speech recognition and introduced a new approach to robust spoken language understanding. In this chapter, we discuss the experiments conducted to build and evaluate the required models to realize a robust telephone-based spoken dialog system and discuss experimental results. We also evaluate the various models and the integrated system as a whole in real-time interaction scenarios with actual test users in two application domains in two languages. The considered application domains are airline travel planning in English and train information inquiries in German.

The experiments conducted can be broadly classified into four major categories; namely, speech recognition experiments, gender and accent related issues, spoken language understanding experiments and evaluation of the demonstration system. Most of the approaches discussed in this chapter have been published in appropriate international media.

8.2 Speech Recognition: English

In the sections that follow, we describe the development experiments conducted in search of optimal parameters for acoustic and language models for the airline travel planning domain in English.

8.2.1 Data Description: English Speech Data

The data used to build the acoustic, language and semantic models for the airline travel planning application domain in English consists of a total of 22 hours of telephone speech and the associated transcriptions from the DARPA Communicator 2001 Evaluation corpus (Walker et al., 2003) procured from Linguistic Data Consortium. The corpus consists of utterances recorded as several users interacted with eight[1] different airline travel planning dialog systems via telephone.

We held out 2 hours of speech consisting of 1,987 utterances spoken by 14 speakers (4 male and 10 female) as a development test-set. To effectively use the limited data we have, we use 5-fold cross-validation technique where we divided the remaining 20 hours of speech into 5 subsets and in each run, one of the 5 subsets is used as a test-set and the other 4 subsets are put together to form a training set. Then the average word accuracy across all 5 tests is computed. Table 8.1 describes the 5 partitions of the 20-hour speech data.

Table 8.1: Data description: 5-fold cross-validation

	No. of Speakers		
Set	Male	Female	No. of Utterances
Set-1	10	25	4143
Set-2	7	18	3529
Set-3	7	18	3086
Set-4	7	18	3508
Set-5	7	18	3605

The held-out development test-set is used in order to determine optimal values for various feature extraction, acoustic and language modeling parameters. In the experiments that follow in Sections 8.2.5, 8.2.6, 8.2.7 and 8.2.8 we use all except the first set as training data spoken by 100 speakers (28 male and 72 female) and the held-out data as development test-set.

A significant amount of the speech data described above contain long silences and non-speech, noisy segments at the beginning of the utterances which could result in poor

[1] AT&T, BBN, CMU, IBM, Lucent Bell Labs, MIT, SRI and University of Colorado at Boulder

performance. Therefore, as part of the data preparation, we wrote a small program to remove these silence segments using the sox utility (Norskog, 1995).

8.2.2 Acoustic Model Training

8.2.2.1 Context-Independent Models

The initial prototype of each context-independent monophone is represented as a hidden Markov model (HMM) of 3 emitting states with left-to-right topology with one Gaussian component per state and no skip transitions as can be seen in Figure 8.1.

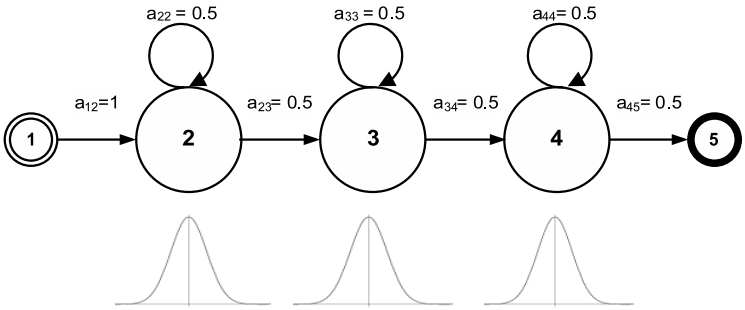

Figure 8.1: Initial context-independent monophone model

In this work, only diagonal covariance matrix systems are considered where the features in each feature vector are assumed uncorrelated. The monophone set consists of 42 HMMs including silence and short pause (sp).

The HMMs are initialized with the flat-start scheme as described in Section 5.2.3. Then, the parameters of the models are re-estimated in 2 consecutive runs of the Baum-Welch algorithm using the monophone transcription of the training data. To handle impulsive noises in the training data, extra transitions are added from state 2 to 4 and from state 4 to 2 in the silence model. The backward transition provides a mechanism to absorb impulsive noises without exiting the silence model. Besides, in order to account for any pauses introduced by the speaker between words of an utterance, a one state short pause (sp) model is created whose emitting state is tied to the center state of the silence model. This short pause model (also called tee-model) has a direct transition from entry to exit state. Then 2 more iterations of the Baum-Welch algorithm are run. As the pronunciation dictionary contains some words with multiple pronunciations, a new transcription is

generated that best matches the acoustic evidence by running the Viterbi algorithm over the training data (known as forced-alignment) (Jurafsky and Martin, 2008; Young et al., 2006).

We then increment the number of Gaussian components up to the desired number. As will be discussed in Section 8.2.4, 32 mixture components per state give optimal performance. To increment the number of Gaussian components, the component with the largest mixture weight is cloned, the weight is divided by 2 and the means are perturbed by a small fraction of the standard deviation (typically +/- 0.2σ). The resulting HMMs are then re-estimated with 4–8 consecutive runs of the Baum-Welch algorithm. This is repeatedly done until we have estimated the models with the required number of mixtures.

8.2.2.2 Context-Dependent Models

As context-independent models do not capture phonetic context, their phonetic discrimination ability is poor. Therefore, in order to achieve good phonetic discrimination, it is common to use triphones where every phone has a distinct HMM model for every unique pair of left and right neighbors. We consider word-internal triphone models where context does not span word boundaries and cross-word triphone models where word boundaries are ignored.

The single-Gaussian monophone models trained as described in the previous section are used to generate triphone prototypes. The transition probability matrix is tied across all triphones of a phone. The resulting triphone model parameters are re-estimated with the Baum-Welch algorithm with a triphone list and triphone transcriptions.

When triphones are used, usually training data becomes insufficient as there are too many models whose parameters must be estimated, hence it is necessary to reduce the number of parameters in an HMM. Diagonal covariance assumption and parameter tying are commonly used methods to reduce the number of parameters that ought to be estimated. Tying (Bahl et al., 1983) is a method where two or more states that represent similar acoustic data are clustered together to create tied states. When states are tied, all the data which would have been used to estimate each individual untied parameter are effectively pooled leading to more robust estimates for the parameters of the tied state (Young et al., 2006). Decision tree based clustering, which will be briefly discussed in Section 8.2.6, is used to identify the states that can be tied together.

Once we have single-Gaussian, tied-state triphones, the next step is to increment the number of Gaussian mixture components. For triphones as well, it was experimentally found that 32 Gaussian mixtures per state is optimal as will be shown in Section 8.2.4. We start with single Gaussian per state and increment the number of Gaussian components as described in section 8.2.2.1.

8.2.3 Initial Parameter Settings

The initial setting for the HTK parameters used in the experiments that follow is given below.

```
SOURCEKIND   = WAVEFORM  # Defines the natural form of the input data
SOURCEFORMAT = WAV       # Defines the format of the speech data
SOURCERATE   = 1250      # Sampling rate (in 100 ns = 8 kHz)
ZMEANSOURCE  = TRUE      # Removes DC offset from the input audio
                            at the frame level
ENORMALISE   = FALSE     # Energy Normalization
TARGETKIND   = MFCC_E_D_A # Defines the parameter kind and the
                            coefficients to use
WINDOWSIZE   = 250000    # Window length (25 ms)
TARGETRATE   = 100000    # Frame rate (10 ms)
NUMCHANS     = 26        # Number of filter bank channels
NUMCEPS      = 12        # Number of cepstral features
USEHAMMING   = TRUE      # Use of Hamming function for windowing
PREEMCOEF    = 0.97      # Pre-emphasis coefficient
CEPLIFTER    = 22        # Cepstral liftering coefficient
```

Word insertion penalty (WIP) and language model scale factor (LMSF) discussed in Section 5.2.5 are set to -4 and 12, respectively. For the triphone models we set the outlier threshold (RO) and the log-likelihood threshold (TB) that will be discussed in Section 8.2.6 to 200 and 1000, respectively. These values are "informed initial parameters" determined from preliminary experiments on the development set.

In the experiments that follow, the parameters that are found to be more useful in a given experiment are used in the subsequent experiments.

8.2.4 Number of Gaussian Mixture Components

When there is a huge amount of training data covering a wide spectrum of speakers, environments, and application domains, training acoustic models using a large number of Gaussian mixture components can improve the performance of speaker-independent acoustic models. However, the gain in recognition accuracy is at the expense of speed since computation of too many Gaussian parameters may slow down the recognition process. Therefore, the number of Gaussian mixtures components to use in a real-time system is a trade-off between accuracy and speed.

In order to determine an optimal number of Gaussian mixture components for our setup, we investigate number of Gaussian mixture components from 4 to 48 by steps of 4. Figure 8.2 shows the results of the experiments.

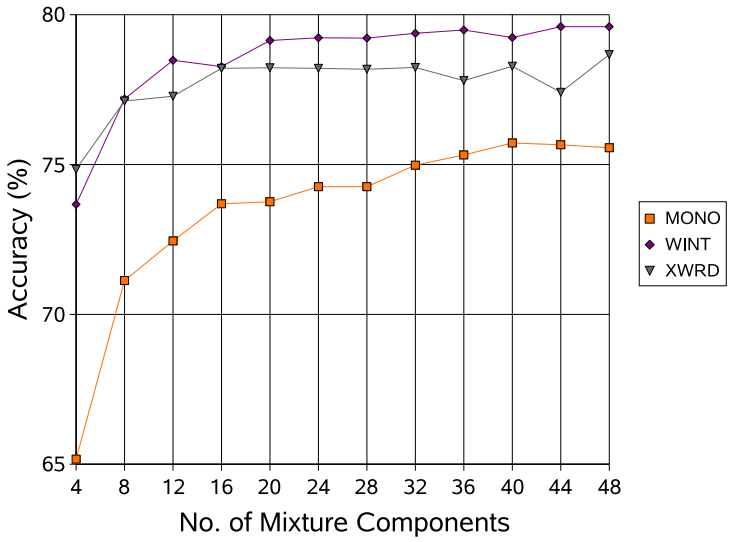

Figure 8.2: Number of Gaussian mixture components for monophone (MONO), word-internal (WINT) triphone and cross-word (XWRD) triphone based models

As can be seen, the performance gain that can be achieved by using Gaussian mixture components greater than 32 is quite insignificant at the expense of more computation. Since the models are for real-time use, speed of recognition is of paramount importance. Hence, as a compromise we use 32 Gaussian mixture components for all (monophone, word-internal and cross-word triphone) models which may incur little loss of recognition accuracy.

8.2.5 Feature Extraction

In order to determine the optimal feature parameters and values to build a robust acoustic model, we carried out a series of experiments on the development test-set. Mainly, we investigated the use of power spectrum versus magnitude spectrum, the 0^{th} order cepstral coefficient versus absolute energy, the use of cepstral mean normalization, etc. We also investigate PLP and LPCC features discussed in Section 5.2.1 in addition to MFCCs. In the following sections we discuss the results of selected experiments.

8.2.5.1 Power versus Magnitude Spectrum

In this subsection, the effect of computing MFCCs from the magnitude spectrum and the power spectrum of the Fourier Transform is investigated.

Table 8.2: Power vs. magnitude spectrum. The notation MONO stands for monophone models, WINT stands for word-internal and XWRD for crossword triphone models. USEPOWER = TRUE means use power spectrum instead of magnitude

USEPOWER	HMMTYPE	Accuracy (%)
FALSE	MONO	74.98
TRUE	MONO	76.07
FALSE	WINT	79.38
TRUE	WINT	80.12
FALSE	XWRD	78.24
TRUE	XWRD	78.99

As can be seen in Table 8.2, using the power spectrum yields better results than the magnitude spectrum in all the cases. This may be attributed to the fact that power spectrum causes large variation in amplitude between voiced and unvoiced speech (since magnitude is squared) as well as smaller variations between different articulations of phonemes (O'Shaughnessy, 2000). Hence, the information needed to determine the phonetic identity of a portion of speech could be captured better in power spectrum than in magnitude spectrum.

8.2.5.2 Mel-Frequency Cepstral Coefficients (MFCCs)

In addition to the first twelve basic Mel-frequency cepstral coefficients, a feature vector can consist of an energy term and information about the rate of change of spectral features. The energy term can be the log of the signal energy or the 0^{th} cepstral coefficient. We investigate the effect of using the 0^{th} cepstral coefficient which is often discarded instead of the log energy as energy term, and the effect of applying cepstral mean normalization. Table 8.3 presents the results of selected experiments on the development set

Note: Since using the power spectrum was found more useful in the previous section, we set USEPOWER = TRUE in the subsequent experiments.

Table 8.3: MFCC Parameters. The notation _E stands for log of the signal energy, _0 represents the 0^{th} order cepstral coefficient, _D stands for Δ coefficients, and _A for $\Delta\Delta$ coefficients while _Z represents CMN

HMMTYPE	TARGETKIND	Accuracy (%)
MONO	MFCC_E_D_A	76.07
	MFCC_E_D_A_Z	76.17
	MFCC_0_D_A	79.12
	MFCC_0_D_A_Z	**81.07**
WINT	MFCC_E_D_A	80.12
	MFCC_E_D_A_Z	79.53
	MFCC_0_D_A	82.45
	MFCC_0_D_A_Z	**82.62**
XWRD	MFCC_E_D_A	78.99
	MFCC_E_D_A_Z	78.96
	MFCC_0_D_A	82.32
	MFCC_0_D_A_Z	**83.50**

As can be clearly seen in Table 8.3 using the 0^{th} cepstral coefficient as the energy term yields significantly improved results in all the cases. Traditionally the 0^{th} MFCC coefficient is considered futile and is often replaced with the log of the signal energy. However, as we observed in a series of experiments in various applications the 0^{th} coefficient is more useful; hence, should be not be simply ignored. The 0^{th} coefficient contains a col-

lection of average energies of each frequency band in the signal that is being analyzed (Fang and Guoliang, 2000).

It can also be observed that cepstral mean normalization (CMN), which is used to handle mismatch in channel conditions, gives a performance boost when used with the 0^{th} coefficient. For telephone recordings where the microphone and transmission quality of each apparatus is different, CMN can provide significant robustness. CMN also mitigates the effect of additive noise. One drawback of CMN is that it does not discriminate silence and speech in computing the utterance mean (Huang et al., 2001b). Preliminary experiments conducted before removing the silence and non-speech noisy segments in the input utterances resulted in degraded performance with CMN. It has also been suggested in (Alsteris and Paliwal, 2005) that CMN is effective when applied to utterances longer than 2-4 seconds.

8.2.5.3 Filter-bank Channels and Window Size

A filter-bank is a collection of filters that separates the input signal into a number of frequency bands where the signal energy is measured. The number of filter-bank channels and the spacing of their central frequency (logarithmic vs. linear) are essential factors that affect the quality of the features extracted from a speech signal. The number of filters should be large enough to resolve the speech spectrum effectively and small enough to allow that all the bands have sufficient filter bandwidth. The number of filters varies for different implementations from 24 to 40 (Huang et al., 2001b). We investigated various sensible values (in the range of 24 to 30) in an attempt to find optimal values for the various models. As a result 26 is found to be optimal for the monophone and triphone models.

In feature extraction, window size and frame rate are also important parameters. A window should be long enough to capture sufficient salient information to calculate the desired parameters and short enough to maintain the assumption that a signal is short-time stationary. In practice, window size is on the order of 20 ms to 30 ms (Huang et al., 2001b). In an attempt to find an optimal window size for each model, a series of sensible window size values in the range of 20 ms to 30 ms have been tried and it was found that 20 ms and 25 ms are optimal values to capture the salient short-time events for the

monophone models and triphone models, respectively. Table 8.4 shows the window size and the number of filters at which better results are achieved for the various models.

Table 8.4: Number of filters and window size

HMMTYPE	NUMCHANS	WINDOWSIZE	Accuracy (%)
MONO	26	20	81.75
WINT	26	25	82.62
XWRD	26	25	83.50

8.2.5.4 Cepstral Liftering

Since higher order Mel-frequency cepstral coefficients are usually numerically small, cepstral liftering is applied in order to re-scale cepstral coefficients so that all dimensions have about the same magnitude. However, as described in (Paliwal, 2005) liftering of cepstral coefficients has no effect when used with continuous observation density hidden Markov models. Our experiments on the development set are also in agreement with the fact that cepstral liftering has little effect on the performance of continuous observation density HMMs.

8.2.6 Clustering: Triphones

To distinguish clusters and tie acoustically similar states within triphone sets, we use a phonetic decision tree that is based on asking phonetic questions about the left and right contexts of each triphone. A phonetic decision tree is a binary tree in which a yes/no question about phonetic context is attached to each node. The tree is used to recursively partition a set of states into subsets by answering the questions as appropriate for the triphone context in which each state occurs (Nock et al., 1997). Those states that end at the same leaf node are judged to be acoustically similar and are then tied.

When clustering, one needs to look for appropriate values for the stopping criteria; namely, the outlier threshold (denoted by RO in HTK) and the threshold specifying the increase in log likelihood that has to be achieved by any question at any node (denoted by TB in HTK). These values affect the degree of tying and the number of states output in the clustered system (Young et al., 2006).

The outlier threshold determines the minimum number of triphone states each leaf in the decision tree must have. This means each cluster must have at least this value of samples associated with it, otherwise it is merged with its next nearest cluster. On the other hand, if a split in the decision tree increases the log-likelihood by less than the value denoted by TB, splitting stops and the decision tree is complete.

In order to determine the optimal values for TB (keeping the value of RO at 200), we investigated values from 800 to 1200 by steps of 100 and 1000 (i.e., the value used in the experiments so far) was found optimal for both crossword and word-internal triphones.

8.2.7 Language Modeling

Due to the incorrect independence assumptions used in acoustic modeling, the acoustic models are underestimated as discussed in Section 5.2.5. Therefore, it is desirable to balance the probabilities of the acoustic model and the language model. This is often done by finding an optimal language model scaling factor (LMSF) that defines how the language model log probabilities are scaled before they are combined with the acoustic log probabilities. In other words, the language model scaling factor (also known as language weight) balances the acoustic and language model scores in word sequence likelihood computations.

Adjusting the language model scaling factor may result in more insertion errors, as the decoder prefers a greater number of shorter words to long ones. One can control the rate of word insertion and word deletion rate, by adjusting the word insertion penalty (WIP). A very large word insertion penalty reduces the word insertion rate and increases the word deletion rate, and a very small penalty has the opposite effect (Rabiner and Juang, 1993). A value for word insertion penalty is experimentally determined at a point where the insertion and deletion errors are nearly equal.

We looked for the optimal values in the range 10 to 20 for the language model scaling factors and -4 to -12 for word insertion penalties on the development test-set to find the point where the number of insertion and deletion errors are nearly equal. Table 8.5 shows the performance of the various models at the optimal values.

Table 8.5: Language model scaling factor and word insertion penalty

HMMTYPE	LMSF	WIP	Accuracy (%)
MONO	12	-4	81.75
WINT	16	-8	84.02
XWRD	16	-9	85.46

8.2.8 Comparison of MFCC, PLP and LPCC Features

In the preceding experiments, we used the MFCC features. As described in 5.2.1 PLP and LPCC features are also quite suitable for speech recognition. Hence, we investigated these features to see how well they perform in our setup. Table 8.6 shows the performance of the models using these features on the development test-set.

Table 8.6: Comparison of features

Feature	HMMTYPE	Accuracy (%)
MFCC_0_D_A_Z	MONO	81.75
	WINT	84.02
	XWRD	85.46
PLP_0_D_A_Z	MONO	81.72
	WINT	83.67
	XWRD	84.60
LPCC_E_D_A_Z	MONO	81.00
	WINT	84.76
	XWRD	85.06

As can be observed, the models based on the three different front-ends give quite comparable results.

8.2.9 Evaluation: English

Using the optimal values found for the various parameters in the development experiments, we performed 5-fold cross-validation experiments. Table 8.7 presents the results.

As can be seen in Table 8.7 and Figure 8.3, MFCC, PLP and LPCC-based models give quite comparable performance.

Table 8.7: Performance of the English system: 5-fold cross validation

Feature	HMMTYPE	Accuracy (%)					
		Set-1	Set-2	Set-3	Set-4	Set-5	Mean
MFCC_0_D_A_Z	MONO	82.58	83.99	78.94	87.07	81.70	**82.86**
	WINT	85.66	86.93	82.27	90.13	84.90	**85.98**
	XWRD	86.75	88.60	83.02	90.78	85.77	**86.98**
PLP_0_D_A_Z	MONO	82.48	84.12	78.89	86.82	81.51	**82.76**
	WINT	87.11	86.74	81.76	89.86	84.57	**86.06**
	XWRD	87.47	87.85	81.89	90.41	85.73	**86.67**
LPCC_E_D_A_Z	MONO	82.04	84.03	77.55	86.53	83.19	**82.67**
	WINT	86.12	87.05	81.20	89.25	87.69	**86.26**
	XWRD	86.28	87.42	81.51	89.80	87.94	**86.59**

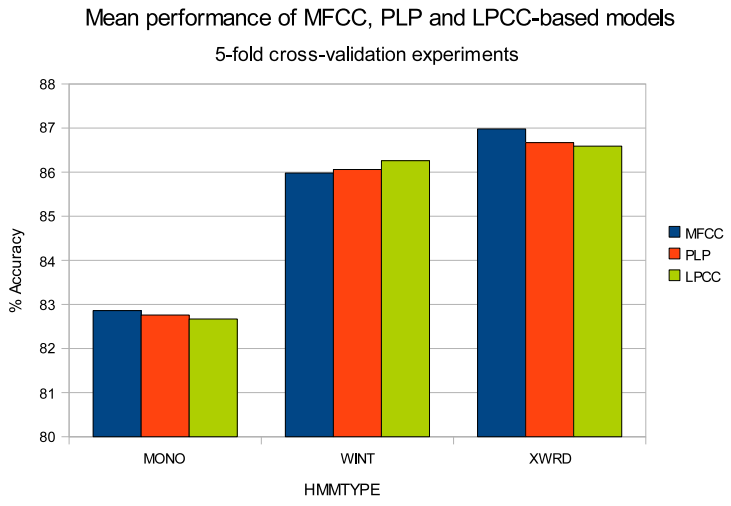

Figure 8.3: Comparison of MFCC, PLP and LPCC-based features for speech recognition

For telephone-based speech recognition, the performance of the models based on the features discussed in Section 5.2.1 is quite good. By using gender-dependent acoustic models, dialog state-dependent language models or grammars and ignoring semantically irrelevant filler phrases the performance can be even better in real-time application. However, since these models are trained on native US-English speech, they perform rather poorly for non-native speech as will be discussed in Section 8.5.2.3. Hence we need to

tailor these models to suit the vocal characteristics of the target users of the system as will be discussed in Sections 8.4, 8.5.1, 8.5.2, and 8.5.3.

8.3 Speech Recognition: German

8.3.1 Data Description

The German speech recognizer is built using 15 hours of speech data spoken by 101 (40 female and 61 male) speakers from the German domain-dependent speech database ©"Erlanger Bahnansage" (ERBA) obtained from the Bavarian Archive for Speech Signals (BAS)[1]. The speech data was recorded with close-talking microphone in a quiet office environment at 16 kHz sampling rate in the domain of train information inquiries. The test-set consist of 100 unique utterances spoken by 5 (2 female and 3 male) native-German speakers. Each speaker is recorded twice – one under the same recording conditions as ERBA and another recorded over a public telephone line – giving a total of over 1000 utterances.

An ideal corpus for spoken dialog systems would be one containing natural spontaneous speech as in everyday conversations, or recorded from Wizard of Oz simulations and/or mixed-initiative human-computer interactions. The ERBA corpus is prepared mainly taking acoustic and domain coverage into consideration and consists of read speech in a quiet office environment and is not directly suitable for building acoustic models for use in a spoken dialog system. As an effort to make the data suitable for a dialog system we recorded additional 204 utterances (e.g. utterances consisting of "ja", "nein", etc.) from 3 female and 6 male speakers with close talking microphone. We use 136 of these utterances from 2 female and 4 male speakers in the training set and the rest in the test-set.

We intend to investigate if one can build a usable acoustic model from "simulated" telephone-quality speech for use in a telephone-based spoken interaction system. Telephone quality speech can be simulated from microphone recorded data by introducing the obvious effects of the telephone channel such as band-limiting, down sampling to 8 kHz, etc. as described in Section 5.3.4.

[1] http://www.phonetik.uni-muenchen.de/Bas/

The language model used is a back-off bigram language model trained on the transcriptions of the training utterances.

8.3.2 Evaluation: German

We carried out preliminary experiments using a small part of the training data as a development set to find optimal parameters for the German system. Most of the parameters that were found optimal for the English system were also found good for the German system. Additional parameters are set to introduce the band-limiting effect of the telephone channel that discards any information of the speech spectrum within 0–300 Hz and 3400–4000 Hz bands. Considering the specific set of German phonemes and their phonetic characteristics, we also modified the phonetic questions used to generate the phonetic decision tree used for clustering. The results of a final evaluation on the test-set are given in Table 8.8.

Table 8.8: Performance of the German system on the evaluation test-set

Feature	HMMTYPE	Accuracy (%)
MFCC_0_D_A_Z	MONO	84.42
	WINT	88.02
	XWRD	89.30
PLP_0_D_A_Z	MONO	84.46
	WINT	88.33
	XWRD	89.55
LPCC_E_D_A_Z	MONO	85.57
	WINT	89.45
	XWRD	90.03

In general, it can be seen that the performance of MFCC, PLP and LPCC based models are comparable, although linear prediction derived cepstral coefficients slightly outperform MFCC and PLP-based models. We observed that CMN is particularly essential to mitigate the effect of channel mismatch. However, these models under-perform on actual telephone speech as will be discussed in Section 8.16.

8.4 Gender in Speech Recognition

8.4.1 Gender Recognition

Due to differences in articulatory mechanisms between male and female speakers, it is easy to discern the gender of a speaker given an utterance with a high degree of accuracy. If a spoken dialog system can reliably tell the gender of a speaker from a spoken utterance, then gender-dependent models that are tailored to the vocal characteristic of speakers in the same gender can be used so as to obtain better recognition performance. In the following sections we describe the approach we used to build a gender recognizer used in our system.

8.4.1.1 Data Description

The gender recognizer we build should be able to robustly estimate the gender of a speaker from a single short utterance. Therefore, the data used to train the gender recognition model consists of 6250 single-word utterances such as "yes", "yeah", "no", "Boston", etc. spoken by 28 male and 72 female speakers extracted from the training set (i.e., the merger of Set-2, Set-3, Set-4, and Set-5) described in Section 8.2.1. For testing purposes, we extracted 1750 single-word utterances from the test-set (i.e., set-1) spoken by 10 male and 25 female speakers. The transcriptions of each utterance is changed to either "Male" or "Female" according the gender of the speaker.

8.4.1.2 Gender Recognition Experiments

The most salient cue for distinguishing adult male and female speech is the fundamental frequency (F0) – pitch as discussed in Section 2.5. However, in telephone speech much of the low frequency energy is filtered out, hence the required pitch is either missing or weak. Therefore, as an alternative, we use the common cepstral features used in speech recognition to estimate the gender of a speaker given a spoken utterance.

The gender recognizer described in this thesis is based on a Gaussian Mixture Model (GMM) discussed in Section 5.3.2 using cepstral features to decide the gender of a speaker from the first, short utterance in a dialog session. The number of Gaussian mixture components required to adequately train the model and the number of iteration between each Gaussian increment were experimentally found to be 32 and 4, respectively. In order to

find out which features and coefficients yield better result, we investigate MFCC, LPCC, and PLP features with 39 coefficients extracted as described in Section 5.2.1 and the results are shown in Table 8.9.

Table 8.9: GMM-based gender recognizer

Feature Kind	Accuracy (%)
MFCC_E_D_A	71.31
MFCC_0_D_A	96.34
PLP_E_D_A	92.06
PLP_0_D_A	95.71
LPCC_E_D_A	92.86

As can be seen in Table 8.9, MFCC feature vectors including the 0^{th} cepstral coefficient as the energy term give the best result while MFCC features with the log of the signal energy give the worst result. The use of 0^{th} cepstral coefficient is also shown to be more useful for speech recognition as described in Section 8.2.5.2. A likely conjecture is that MFCC features along with the 0^{th} cepstral feature as the energy term encode more gender-specific information sufficient to recognize the gender of a speaker from a single-word utterance over the telephone with acceptable performance – 96.34% than the other features investigated.

From the confusion matrix in Table 8.10, one can observe that both male and female speakers are identified at a comparable accuracy, although female speakers are slightly better recognized than male speakers.

Table 8.10: Gender recognition confusion matrix using MFCC_0_D_A

	Male	Female
Male	95.77%	4.23%
Female	3.31%	96.69%

8.4.2 Gender-Dependent Acoustic Modeling

8.4.2.1 Data Description

A straight forward approach to take advantage of gender recognition to improve speech recognition performance is to train gender-dependent acoustic models. For this purpose, we split the training data (i.e., the merger of Set-2, Set-3, Set-4 and Set-5) into two gender-dependent sets – "female training set" consisting of 10,133 utterances spoken by 72 female speakers and "male training set" consisting of 3,594 utterances spoken by 28 male speakers. For evaluation, we split the test-set (i.e., Set-1) also into two gender-dependent sets – "female test-set" consisting of 3,107 utterances spoken by 25 speakers and "male test-set" consisting of 1,036 utterances spoken by 10 speakers.

8.4.2.2 Gender-Dependent Models

To measure how well the speaker-independent model performs for a given male or female user, we evaluated the performance of the MFCC-based SI model on the "male-only" and "female-only" test-sets as shown in Table 8.11. For reasons that will become evident in Section 8.5.2, we use monophone models.

Table 8.11: Performance of the SI model on separate male-only and female-only test-sets

HMMTYPE	Gender	Accuracy (%)
MONO	Male	79.57
MONO	Female	83.73

To see if gender-dependent models give better performance, we train separate male and female acoustic models and evaluated them on the corresponding gender-dependent test data. The results are shown in Table 8.12.

Table 8.12: Performance of gender-dependent acoustic models

HMMTYPE	Gender	Accuracy (%)
MONO	Male	80.47
MONO	Female	84.66

It can be seen in Tables 8.11 and 8.12 that the performance of the gender-dependent models perform better than the SI model. In general, the gender-dependent models are

quite good for telephone-based speech recognition. However, as mentioned earlier, these models perform rather poorly for non-native speakers of English. In the following sections, we discuss the approaches we used to deal with non-native speech.

8.5 Accent in Speech Recognition

8.5.1 Accent Recognition

In multi-user systems that serve people with different demographic and sociolinguistic background, discerning the gender and the accent of a speaker from a spoken utterance may be useful to load proper acoustic models specifically tailored to the vocal characteristics of a particular group of speakers. A straight forward approach to build an accent recognizer is to train the model on accented speech data collected from different groups of non-native speakers of a language. However, accented speech is rarely available in enough amount to build a reliable accent recognizer. Therefore, in this experiment we show the feasibility of training an accent recognizer on native speech data of the target accent groups. In particular, we train a GMM-based accent recognizer with 32 components on a merger of native German and native English speech data and we use the resulting model to distinguish if a given English utterance is accented or native.

8.5.1.1 Data Description

The training data used for accent recognition consists of over 17,800 English utterances spoken by 135 native US-English speakers drawn from the DARPA 2001 Communicator Evaluation corpus (Walker et al., 2003) and over 10,300 German utterances from ©ERBA speech corpus described in Section 8.3.1 and additional utterances recorded at our laboratory. The test-set consists of 1200 German-accented English utterances recorded from 30 (15 male and 15 female) native German speakers over the telephone and 1800 native English utterances selected from the held-out development test-set described in Section 8.2.1 giving a total of 3000 utterances. Since the German training data is microphone-recorded, it was converted to a "simulated" telephone quality speech before it was used to train the desired model. The transcriptions of the utterances in the training set consist of either "German" or "English" based on the language of the spoken utterance while the

transcriptions of the test data are set to either "German" or "English" based on whether a given English utterance was spoken by a German-accented or a native English speaker.

8.5.1.2 Training

We trained one GMM for each langauge (accent class); namely, one for English and another for German. The initial prototype of each GMM is represented as a single-state HMM with one Gaussian component where there is no state transition probability within the model. We model silence as a hidden Markov model (HMM) of 3 emitting states with left-to-right topology with one Gaussian component per state and no skip transitions.

The parameters of the initial models are re-estimated in 2 consecutive runs of the Baum-Welch algorithm using the training data and the associated transcription. The transcription of the training data, as noted in Section 8.5.1.1 indicates the language of each spoken utterance in the training data – i.e., either German or English. To handle impulsive noises in the training data, two more transitions are introduced from state 2 to 4 and from state 4 to 2 in the silence model. The backward transition provides a mechanism to absorb impulsive noises without exiting the silence model. Then 2 more iterations of the Baum-Welch algorithm are run.

Finally, we convert the single-Gaussian models to 32-mixture component models as described in Section 8.2.2.1. After each mixture increment, the resulting models are re-estimated with 4 consecutive runs of the Baum-Welch algorithm until we have estimated the models with the required number of mixtures.

8.5.1.3 Accent Recognition Experiments

In order to find out which features and coefficients are best suited for accent detection, we investigated the use of MFCC, PLP and LPCC features where each feature vector is composed of the basic 12 static coefficients and the energy (or 0^{th} order coefficient) with and without the corresponding delta and delta-delta coefficients. The performance of the resulting models is shown in Table 8.13.

As can be seen in Table 8.13, generally LPCC-based models outperform MFCC and PLP based models for accent recognition. In particular, LPC-derived cepstral coefficients consisting of 12 cepstral coefficients and the energy term without the delta and delta-delta coefficients yield the best performance. The better performance of LPCCs for accent

Table 8.13: Performance of MFCC, PLP and LPCC features on accent detection

Feature Kind	Coefficients	Accuracy (%)
MFCC	_E	63.80
	_0	82.57
	_0_D	83.77
	_0_D_A	81.83
PLP	_E	61.87
	_0	82.97
	_0_D	83.37
	_0_D_A	83.73
LPCC	_E	**90.33**
	_E_D	88.37
	_E_D_A	87.10

recognition may be attributed to the fact that LPC-based techniques nicely model the speech production process of the vocal tract which is highly variable both across languages and regional accents. Since speakers with foreign accents usually introduce some acoustic and phonological features from their native languages into the speech production process, the accent of a speaker can be robustly estimated using a speech production model trained on the native speech data of the target accent groups.

Similarly, in (Wong and Sridharan, 2001) it has been shown that LPCC-based features consistently outperformed MFCCs in language identification task. It has also been suggested in (Arslan and Hansen, 1997) that mel-scale based analysis is not particularly suitable for accent detection.

Further observation in Table 8.13 shows that there is a drastic gain in performance when the 0^{th} order cepstral coefficient is used as energy term instead of the log energy in MFCC and PLP. The 0^{th} coefficient represents the average energy in the speech frame and we consistently observed that it is more useful than the log energy in different applications. With LPCC, the dynamic features – namely, delta and delta-delta coefficients do not appear to be useful for accent detection.

It's interesting to observe in the confusion matrix in Table 8.14 that German-accented speakers are identified at a much better rate (96.25%) than native speakers (85.28%).

This further confirms our hypothesis that accent-related information can be effectively captured from native speech data.

Table 8.14: Accent recognition confusion matrix using LPCC_E

	Native	German-accented
Native	85.28%	14.72%
German-accented	3.75%	96.25%

8.5.2 Accent Adaptation Using Accented Data

The performance of the acoustic models trained on native-US English data degrades significantly when used with actual users with German accent as can be seen in Section 8.5.2.2. Therefore, in order to obtain robust acoustic models that can perform well with accented speakers, we adapt the speaker-independent and gender-dependent models trained on native data to the vocal characteristics of German-accented speakers.

It has been reported in (He and Zhao, 2001) that triphones trained on native speech are not appropriate for use with non-native speech. In fact, our preliminary experiments also revealed that monophones outperform triphones in recognizing non-native speech. Therefore, in the experiments described in the following sections, we used the context-independent monophone models built as described in Section 8.1 as the seed models.

8.5.2.1 Data Description

The enrollment set consisting of 600 English utterances recorded over the telephone from 10 male and 10 female native German speakers (i.e., 30 utterances from each speaker) using prompts drawn from the English training-set. The test-set consists of 600 English utterances recorded over the telephone from 5 male and 5 female native German speakers (i.e., 60 utterances from each speaker) using prompts drawn from the English test-set (i.e., Set-1).

8.5.2.2 Baseline Performance

A summary of the results of the SI, and gender-dependent monophone models trained on native US-English data on the German-accented English test-set is presented in Table 8.15.

Table 8.15: The performance of the SI and gender-dependent seed models on accented speech

Acoustic Model	Accuracy (%)
Speaker-Independent (SI)	61.40
Gender-Dependent (Female)	60.05
Gender-Dependent (Male)	62.29

As can be observed, the performance of the native models on accented speech is very poor compared to the performance of the models on native speech described in Section 8.2.9.

8.5.2.3 Accent Adaptation Experiments

We obtained significant boost in performance by using the standard speaker adaptation techniques; namely, MLLR, MAP and MLLR followed by MAP, where we use German-accented English speech as adaptation data to adapt the native English SI and gender-dependent acoustic models. For MLLR, optimal performance was obtained with 42 regression classes where both means and diagonal covariances are transformed. Figure 8.4 summarizes the results obtained.

As can be observed in Figure 8.4 MLLR alone resulted in 11.01% absolute (28.52% relative) WER reduction for the SI model. On the gender-dependent models, we obtained 9.69% absolute (24.26% relative) and 17.44% absolute (46.25% relative) WER reduction for female and male models, respectively. Comparable improvement could also be achieved with two iterations of MAP adaptation. However, the best results were achieved by applying three iterations of MAP adaptation on the MLLR transformed models where 14.5% absolute (37.56% relative), 13.47% absolute (33.72% relative), and 20.19% absolute (53.54% relative) WER reduction with SI, female and male models are obtained, respectively. It can also be observed that the accent-adapted male models perform much better than the accent-adapted female models. Our speculation based on the analysis of the recorded data is that the male speakers spoke with more natural accent, hence more

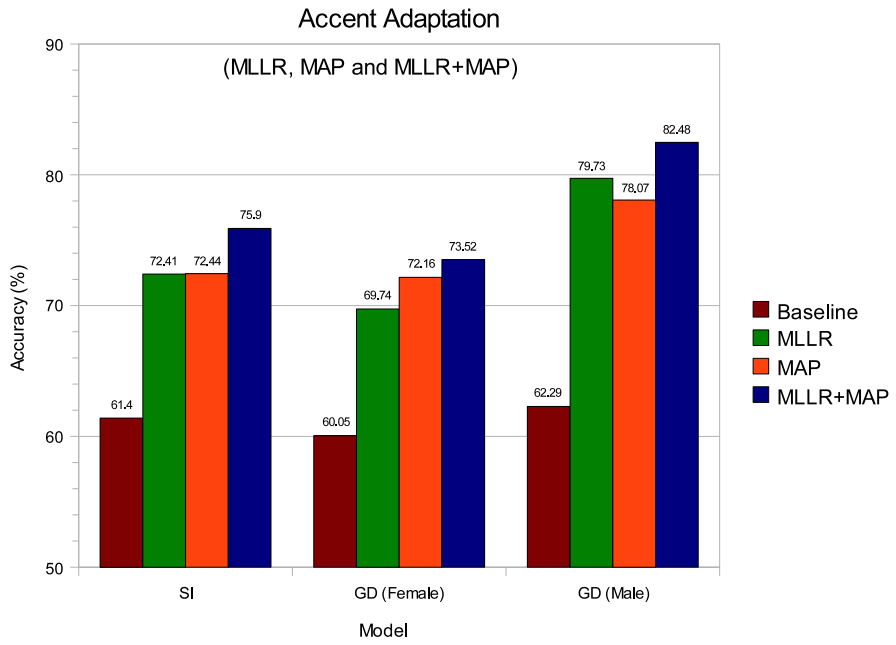

Figure 8.4: Speech recognition performance improvement for non-native speakers due to MLLR, MAP and MLLR+MAP adaptation

accent-related information could be obtained. On the other hand, most of the female speakers tried to imitate native speakers, hence were less accented than the male speakers.

8.5.3 Cross-language Accent Adaptation

Motivated by the successful utilization of native speech data from two target accent groups (German and native US-English) to train an accent recognizer that detects accent from a spoken utterance as described in Section 8.5.1, we hope to obtain performance gain by using native German speech data as enrollment set to adapt models trained on native US-English speech. This is referred to as cross-language accent adaptation. Cross-language accent adaptation is helpful in cases where it is hard to collect accented speech in a given application domain. Often native speech data from a target accent group is more available than accented speech data. In this section, we show the use of native German speech data to adapt speaker-independent and gender-dependent models trained on native US-English data to the German accent.

8.5.3.1 Data for Cross-language Accent Adaptation

The enrollment data for cross-language accent adaptation consist of 600 German utterances recorded over the telephone from 10 male and 10 female native German speakers using prompts drawn from the transcriptions of the test-set in the ©ERBA distribution. The evaluation-set consists of 600 German-accented English utterances described in Section 8.5.2.1.

In order to use cross-language accent adaptation, we first constructed an approximate mapping between the phoneme sets of German and English. We then built an auxiliary pronunciation dictionary that defines the pronunciation of the German words in the adaptation set with English phonemes.

8.5.3.2 Cross-language Accent Adaptation Experiments

The enrollment data is force-aligned using the English SI monophone model and the auxiliary pronunciation dictionary to produce the monophone transcription of the enrollment data. For MLLR, two global transforms (one for silence and another for speech models) where both means and covariances are transformed give improved result. Figure 8.5 summarizes the results obtained.

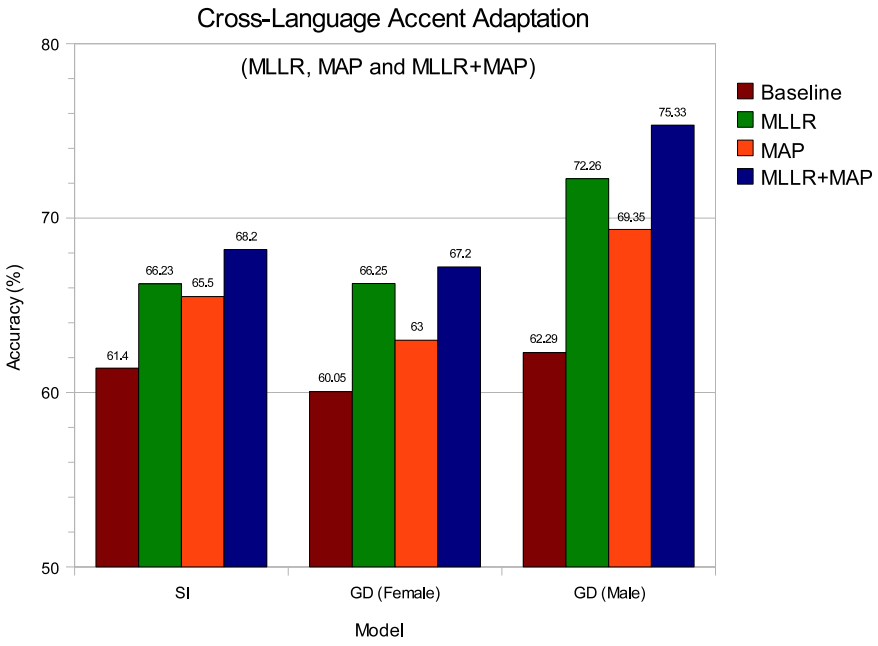

Figure 8.5: Speech recognition performance improvement for non-native speakers using cross-language accent adaptation

As can be seen in Figure 8.5, when MAP is applied on MLLR transformed means and covariances, we obtain 6.8% absolute (17.62% relative), 13.04% absolute (34.58% relative), and 7.15% absolute (17.90% relative) WER reduction for the SI, male and female models, respectively. Although the performance gain is remarkable, the improvement we could get using cross-language approach is relatively less than we achieved using accented data as demonstrated in Section 8.5.2. One explanation is that the phoneme mapping is neither one-to-one nor accurate as some German phonemes do not have a counterpart in English and vice versa.

8.5.4 Channel Adaptation

The acoustic models for the German system were trained and evaluated on simulated telephone quality speech. Experiments show that these models do not perform well on actual telephone speech. Therefore, we use maximum likelihood linear regression adaptation technique to adapt the models to the telephone channel using some telephone recorded speech data with the hope of improving performance of the adapted models on telephone speech.

8.5.4.1 Data Description

We recorded 300 German utterances from 5 male and 5 female native German speakers over the telephone using prompts drawn from the training set of the German speech corpus (ERBA) for the purpose of channel adaptation.

The 600 German-accented English utterances from 10 male and 10 female speakers described in Section 8.5.2.1 are used to evaluate how well the speaker-independent (SI) and gender-dependent models trained on simulated telephone-quality speech perform on actual telephone speech.

8.5.4.2 Effect of Channel Adaptation

Table 8.16 summarizes the performance of the monophone speaker-independent and gender-dependent models on actual telephone speech before and after channel adaptation.

As can be seen, there is apparent performance gain after MLLR adaptation. The MLLR adaptation (with 42 regression classes and transformation of both mean and diagonal covariance parameters) may have captured some effects of the telephone channel

Table 8.16: Performance gain due to channel adaptation

Acoustic Model	Before (Accuracy (%))	After (Accuracy (%))
SI	76.94	79.87
GD (Male)	78.72	82.03
GD (Female)	79.00	80.30

that were not captured with the simulation procedure we described in Section 5.3.4. The actual performance of the gender-dependent models in real-time use will be discussed in Section 8.7.

8.6 Spoken Language Understanding

In the previous sections, we investigated various approaches to build robust speech recognition models that decode a given acoustic signal into a sequence of words which is hopefully close to the correct transcription of the spoken utterance. However, the raw output of a speech recognizer can merely serve any purpose in a spoken dialog system without a process that converts the output of the speech recognizer into a meaningful sequence of semantic concepts that connote what is meant from what might have been said. In this section, we describe the experiments conducted and the results obtained using different but interrelated semantic concept labeling approaches discussed in Chapter 6.

8.6.1 Data Description

The semantic model for the domain of airline travel planning was trained on the transcriptions of 8000 utterances drawn from the merger of set-2, set-3, set-4 and set-5 of the data described in Section 8.2.1. The evaluation test-set consists of 1000 selected from the transcriptions of set-1. The selection mainly excluded repetitions of short utterances such as "yes", "no", etc. An interesting feature of the data in the DARPA 2001 Communicator Evaluation Corpus is that it consists of spontaneous utterances – consisting of filled pauses, repetitions, repairs, false starts, ungrammatical utterances, etc. Table 8.17 describes the training and test-sets used for the domain of airline travel planning.

As can be seen, there are 79 distinct out-of-vocabulary words (OOVs) in the test-set. In general, there are 139 occurrences of these OOVs in the test-set.

Table 8.17: Description of data for the airline travel planning domain (Communicator)

Set	No. of utterances	No. of unique words	Avg. No. of words per utterance
Training	8000	914	4.04
Test	1000	579 (79 OOVs)	8.98

On the other hand, the model for the train inquiries domain was trained on the transcriptions of 8000 utterances drawn from the 10,000-utterance training set described in Section 8.3.1. The transcriptions of the first 900 utterances from the remaining 2000 were added to the 100-utterance test-set described in Section 8.3.1 to form a 1000-sentence evaluation set. The utterances in the domain of train inquiries are read, relatively long, well-structured, and grammatically well-formed sentences. Table 8.18 describes the data used to build and evaluate the German semantic model.

Table 8.18: Description of data for train information inquiries domain (ERBA)

Set	No. of utterances	No. of unique words	Avg. No. of words per utterance
Training	8000	920	12.26
Test	1000	829 (9 OOVs)	11.76

8.6.2 Performance Measures

The performance of the semantic models is evaluated using precision, recall and F-measure. Precision (P) is the percentage of correctly labeled concepts out of all labeled concepts given by the system. Recall (R) is the percentage of correctly identified concepts actually present in the reference annotation. By correct we mean that both the boundaries of the concept and the label are correct.

$$Precision = \frac{Number\ of\ correctly\ labeled\ concepts}{Total\ number\ of\ labeled\ concepts\ given\ by\ the\ system}$$

$$Recall = \frac{Number\ of\ correctly\ labeled\ concepts}{Total\ number\ of\ labeled\ concept\ chunks\ in\ the\ reference\ annotation}$$

To illustrate the two measures we use the following example. Suppose that the utterance "I'd like to fly on Air Canada" is tagged by a model as:

```
(I'd like to fly) DUMMY
(on) ON
(Air) AIRLINE_NAME
(Canada) COUNTRY
```

And the reference annotation consists of:

```
(I'd like to fly) DUMMY
(on) ON
(Air Canada) AIRLINE_NAME
```

Since two out of the four labeled output are correct, the precision is 50% while the recall is 66.67% since two of the three chunks in the reference are correctly identified.

F-measure (van Rijsbergen, 1975) is a weighted harmonic mean of precision and recall as defined by Equation 8.1.

$$F = \frac{2PR}{P+R} \tag{8.1}$$

8.6.3 The Flat-Concept Model

It has been described in Section 6.5 that the first step in semantic modeling involves identifying the relevant entities, events, attributes and relations within the domain of discourse using prior domain knowledge and example utterances. These identified semantic classes represent the semantic concepts in the application domain and each semantic class constitutes a set of lexical items used in the domain. Therefore, the words in the system's vocabulary are classified into the identified set of semantic classes such that all words belonging to a semantic class are initially equiprobable.

In HMM-based semantic concept labeling, the hidden states correspond to the semantic classes (tags or labels) in a given application domain while the observation set corresponds to the set of words in the lexicon of the system. The task of the required model

is to determine the most likely sequence of semantic labels that could have generated the sequence of words in a recognized utterance.

In Section 6.5.1, it has been noted that the flat-concept model cannot capture the hierarchical relationship of words across states. However, for relatively simple application domains, it can give adequate performance as can be seen in the next sections.

8.6.3.1 Initial Flat-Concept Models

We start with an ergodic model where all transitions to and from any state (semantic class) are equally likely including self-loops. The only restriction is that a one-step transition from the entry to the exit state of the global HMM is prohibited to prevent non-emitting loops as described in Section 6.5. As can be seen in Table 8.19, this very basic model, where little domain-specific constraints are imposed, gives a modest performance which can be considered as a baseline for the flat-concept model.

Table 8.19: Performance of the ergodic initial models

Data	P (%)	R (%)	F-Measure (%)
Communicator	71.49	70.44	70.96
ERBA	80.26	83.64	81.92

This model is too unconstrained and it is often useful to introduce some informative structures by prohibiting arbitrary and unlikely state transitions based on prior domain knowledge and training examples. This enables us to train the model more efficiently on semantically unannotated training data. We use the term "tuning" to refer to the process of introducing constraints in the model. This is efficiently done with the model compiler and the modeling language introduced in Section 6.5 using the keywords "except", "none", "only", "high", "low" and "all" .

After the required tuning of model parameters, the performance of the flat-concept models is given in Table 8.20.

As can be seen in Tables 8.19 and 8.20, introducing prior knowledge-based constraints into the model definition results in a substantial gain in performance – 13.78% and 12.26% absolute improvement in F-measure for the Communicator and ERBA tasks, respectively.

Table 8.20: Performance of the tuned flat-concept initial models

Data	P (%)	R (%)	F-Measure (%)
Communicator	89.47	80.49	84.74
ERBA	94.61	93.76	94.18

The performance can further be improved with EM training. In fact, the tuning of initial model parameters is very important because the EM algorithm heavily depends on these.

8.6.3.2 Trained Flat-Concept Models

It was found that the best performance for the English model (Communicator) was obtained after six iterations of training as shown in Figure 8.6 while a single iteration of the EM algorithm was sufficient for the German models (ERBA). This phenomenon where we achieve the best performance in the first few training iterations is referred to as early maximum (Elworthy, 1994).

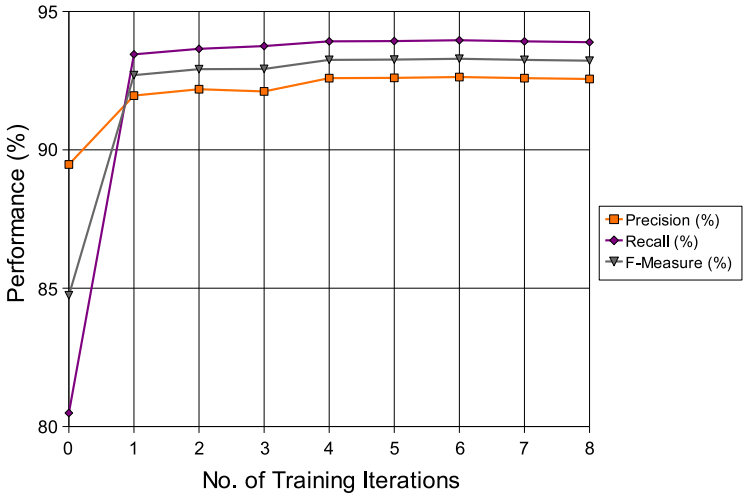

Figure 8.6: Performance of the flat-concept model as a function of number of training iterations (in the Communicator domain)

As can be seen in Figure 8.6, the major performance boost is achieved in the first iteration, afterwards minor gains are obtained until the sixth iteration and starts to de-

cline gradually. After training and smoothing of transition and emission probabilities as described in Section 6.4, the performance of the models is shown in Table 8.21.

Table 8.21: Performance of the flat-concept models after training and smoothing

Data	P (%)	R (%)	F-Measure (%)
Communicator	92.63	93.96	93.29
ERBA	94.55	95.46	95.00

As can be seen, in Table 8.21, the performance of the flat-concept model is significantly improved after training and smoothing and is quite satisfactory for both application domains. However, in order to achieve this level of performance, a substantial number of preliminary testing on the training data had to be performed to tune the model so as to resolve many sources of ambiguities.

Furthermore, as described in Section 6.5.1 the flat-concept model provides a fragmented output where each word is labeled with a corresponding atomic semantic label and, therefore, is less informative as can be noted in the following example in German[1].

Listing 10 Example output of the flat-concept semantic model

```
(ich möchte gerne) DUMMY (am) ON (sechs) DAY_OF_MONTH (und) CONNECTIVE
(zwanzigsten) DAY_OF_MONTH (zweiten) MONTH (um) AT (neun) HOUR_OF_DAY
(Uhr) HOUR (drei) MINUTES (und) CONNECTIVE (zwanzig) MINUTES (die) DUMMY
(schnellste) MODIFIER (Direktverbindung) TRAIN_CONNECTION (von) FROM
(Düsseldorf) CITY_1 (nach) TO (Magdeburg) CITY_1 (Neustadt) CITY_2
```

In the above example, the phrase "sechs und zwanzigsten zweiten"[2] represents a DATE concept and the phrase "neun Uhr drei und zwanzig"[3] represents a TIME concept, etc. Obviously, it would be more informative if these are labeled as DATE and TIME, respectively, while encapsulating the low-level details inside.

Finally, as can be clearly seen in Figure 8.7, the gain in performance after training (and smoothing) for the Communicator application domain is much more substantial than for the ERBA domain. This is due to the occurrence of more number of unseen observations and out of vocabulary words in the Communicator domain than in the ERBA domain

[1] Translation of the German utterance: I would like the fastest direct connection from Düsseldorf Airport to Magdeburg Neustadt on February twenty sixth at 9:23 a. m.

[2] twenty sixth of February

[3] 9:23 a.m.

which are properly handled by the smoothing technique described in Section 6.4. More discussion on this is given in the next section.

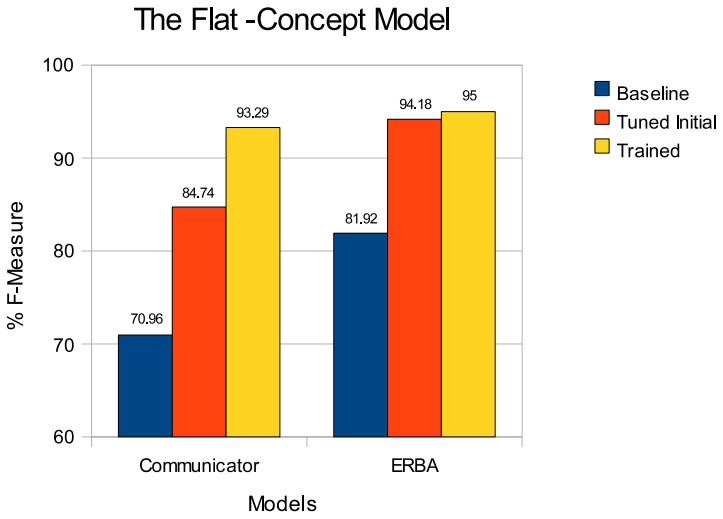

Figure 8.7: Performance of the the baseline, tuned and trained flat-concept models for Communicator and ERBA application domains

8.6.4 The Medium-level Hierarchical Model

In order to combat the problems associated with the flat-concept modeling approach, we organize semantically related concepts into higher-level concepts such as DATE, TIME, FLIGHT_INFO, etc. as described in Section 6.6. We refer to this model as medium-level hierarchical model as it captures the context within semantically related concepts. The experiments that follow describe the performance of the medium-level hierarchical models for both airline travel planning and train information inquiries domains.

8.6.4.1 Initial Medium-level Hierarchical Models

The performance of the initial medium-hierarchical models after introducing the necessary constraints in the model definition as described in Section 6.6 is depicted in Table 8.22.

Table 8.22: Performance of the medium-level hierarchical initial models

Data	P (%)	R (%)	F-Measure (%)
Communicator	96.77	83.61	89.71
ERBA	96.73	95.10	95.91

As can be observed, tuning of model parameters alone can result in a working model with acceptable performance. However, a model that solely relies on engineering the model could require a lot of human effort to tune the model parameters which could otherwise be learnt from the training data. Moreover, the model so built could be brittle in the face of unseen transitions and out-of-vocabulary (OOV) words. Therefore, we aim at introducing only the obvious and most important constraints and let the EM algorithm refine the model parameters during training.

It can also be noted in Table 8.22 that the recall for the airline travel planning domain, is quite low. This is because the test-set consists of a significant number of utterances that could not be parsed due to unseen observations in the training data. This is, in turn, attributed to the inherent data sparseness problem and the inevitability of OOV words in spontaneous spoken utterances. To make the model robust to unseen observations and OOV words, we use the smoothing technique discussed in Section 6.4 to adjust the probabilities of observations to obtain reasonable probabilities for unseen data.

8.6.4.2 Trained Medium-level Hierarchical Models

Using the EM training algorithm and the smoothing technique described in Section 6.4, the model parameters are refined and all the sentences including those containing unseen observations could be effectively parsed. Only a single iteration of the EM algorithm was used as further iterations were found to be counterproductive. The results obtained after training and smoothing are summarized in Table 8.23.

Table 8.23: Performance of the medium-level hierarchical model after training and smoothing

Data	P (%)	R (%)	F-Measure (%)
Communicator	96.82	96.64	96.73
ERBA	96.96	96.68	96.82

In addition to the additional performance gain after training and smoothing, the resulting model is quite robust to unseen observations and can correctly label out-of-vocabulary words using the surrounding context. Moreover, the output is semantically more informative as can be observed in the following examples.

The output of the medium-level hierarchical model on the example utterance given in Listing 10 is shown in Listing 11 which is much more structured and informative than the output of the flat-concept model.

Listing 11 Example output of the medium-level hierarchical semantic model in German

```
(ich möchte gerne) DUMMY (am) ON (sechs und zwanzigsten zweiten) DATE
(um) AT (neun Uhr drei und zwanzig) TIME (die) DUMMY (schnellste) MODIFIER
(Direktverbindung) CONNECTION (von) FROM (Düsseldorf Flughafen) LOCATION
(nach) TO (Magdeburg Neustadt) LOCATION
```

Listing 12 shows an example in the domain of airline travel planning. Given that the departure location Berlin Tegel was not seen in the training data, the sentence "Do you have a flight from Berlin Tegel to Washington Dulles Airport on September the twenty ninth in the morning?" would be labeled as:

Listing 12 Example output of the medium-level hierarchical semantic model for the domain of Airline Travel Planning

```
(Do you have a) QUESTION (flight) FLIGHT_INFO (from) FROM
(oov[Berlin] oov[Tegel]) CITY (to) TO
(Washington Dulles Airport) AIRPORT (on) ON
(September the twenty ninth ) DATE (in the morning) PERIOD_OF_DAY
```

As can be seen, the model is robust in that the OOV information consisting of two adjacent words (i.e. Berlin Tegel) could be correctly labeled using the surrounding context and the resulting tagged output is structured and semantically appealing.

8.6.5 The Hierarchical Model

The hierarchical model extends the medium-level structure further by grouping semantically as well as hierarchically related concepts together so as to improve the ambiguity

resolution ability and the predictive power of the model and obtain more structured output by a hierarchical organization of concepts into higher-level structures.

The model for each application domain is constructed and tuned as described in Section 6.7. Then, we train the tuned initial hierarchical model of each application domain with the EM algorithm and apply the smoothing method described in Section 6.4. The best performance was observed after only one iteration of training on the tuned initial models for both application domains. This is because with good initial estimates, the EM training improves performance only for a few number of iterations – the pattern Elworthy (1994) termed "early maximum". The experiments that follow describe the performance of the hierarchical models evaluated at two different levels of hierarchy.

8.6.5.1 Trained Hierarchical Models

The performance of the hierarchical model is evaluated at two levels of detail. First, we measure how well the model identifies the structured units of information such as REQUEST, ARRIVAL_LOC, DEPARTURE_TIME, etc. without considering the low-level details in each structure such as CITY_P1, CITY_P2, HOUR, MINUTES, etc. Tables 8.24 shows the performance of the hierarchical model after training and smoothing on the high-level tag-set for the two application domains.

Table 8.24: Performance of the hierarchical model on structured (high-level) tag-set

Model	P (%)	R (%)	F-Measure (%)
Communicator	95.07	96.25	95.66
ERBA	96.06	96.27	96.16

Second, in order to compare the performance of the hierarchical model and the flat-concept model using the same tag-set, we generate a detailed output using the hierarchical model, take only the low-level concepts, resolve name differences of similar concepts between the two models, and evaluate the performance using the reference annotation of the flat-concept model. Table 8.25 depicts the result on the tag-set used in the flat-concept model.

As can be observed, the hierarchical model outperforms the flat-concept model by about 2.99% and 3.98% absolute in F-measure (compare with Table 8.21) in the airline travel planning and train inquiries domains, respectively as can be seen in Figure 8.8.

Table 8.25: Performance of the hierarchical model on low-level tag-set

Model	P (%)	R (%)	F-Measure (%)
Communicator	96.08	96.48	96.28
ERBA	98.88	99.08	98.98

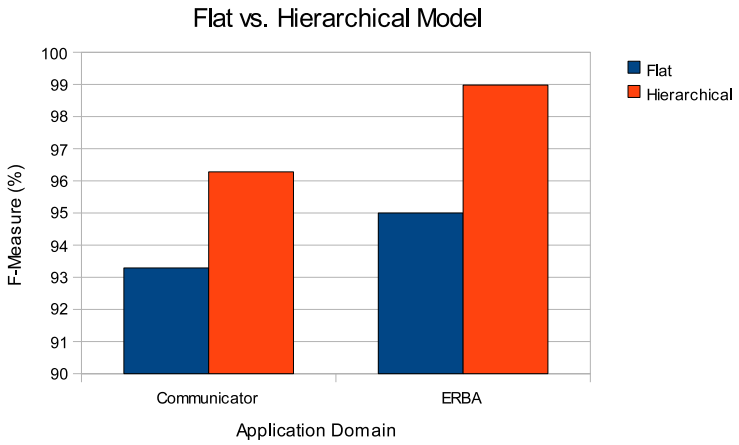

Figure 8.8: Comparison of the performance of the flat-concept and hierarchical models in F-measure

The very high performance of the German semantic model is mainly because both the training and test data are very structured utterances and contain less sources of ambiguity; hence, it can be modeled by the hierarchical approach with a high degree of accuracy.

8.6.5.2 Example Outputs of the Hierarchical Model

The hierarchical model has the virtue of providing semantically labeled information at several levels of detail as required. For instance, given the utterance:

```
"leaving San Francisco on November eleventh traveling to Boston
leaving in the afternoon on flight eleven seventy"
```

The model can produce a higher-level output as shown in Listing 13.

Listing 13 Communicator (Level 1): Structured (high-level) output

```
(leaving San Francisco) DEPARTURE_LOC
(on November eleventh)DATE_INFO
(traveling) DUMMY (to Boston) ARRIVAL_LOC
(leaving in the afternoon) DEPARTURE_TIME
(on) ON (flight eleven oov[seventy])FLIGHT_INFO
```

As can be noted, the hierarchical model provides additional higher-level information such as ARRIVAL_LOC, DEPARTURE_TIME, etc. which we could not directly obtain using either the medium-level hierarchical or the flat-concept model. Moreover, the model is robust in that unknown words could be correctly labeled as in the example above where "seventy" was not seen in the training data but was correctly labeled using the surrounding context.

The model can also produce a detailed low-level output as shown in Listing 14.

Listing 14 Communicator (Level 2): Detailed (low-level) output

```
((leaving) DEPARTURE
(((San) CITY_P1 (Francisco) CITY_P2) CITY_MACRO) LOCATION_MACRO) DEPARTURE_LOC
(((on) ON
((November) MONTH (eleventh) DAY_OF_MONTH) DU_MACRO) DATE_MACRO) DATE_INFO
(traveling) DUMMY
((to) TO (((Boston) CITY_P2)CITY_MACRO)LOCATION_MACRO) ARRIVAL_LOC
((leaving) DEPARTURE (((in the) DUMMY
(afternoon) PERIOD_OF_DAY)POD_MACRO) TIME_MACRO) DEPARTURE_TIME (on) ON
((flight) FLIGHT_QUALIFIER (eleven oov[seventy]) FLIGHT_NUMBER) FLIGHT_INFO
```

As a remark, the hierarchical model could correctly label 66.9% (93 out of 139 occurrences) of OOVs in the airline travel planning domain.

Given the utterance (in the domain of train information inquiries[1]):

```
"Welches ist die schnellste Zugverbindung zwischen Kobern Gondorf
 und Esslingen frühestens übermorgen um acht Uhr fünfzehn"
```

The model can produce a detailed output as shown in 15.

[1] Translation: Which is the fastest train connection between Kobern Gondorf and Esslingen at the earliest of the day after tomorrow at eight fifteen a. m.

Listing 15 ERBA (Level 2) Detailed (low-level) output

```
((welches) QUESTION (ist) DUMMY) REQUEST
((die) DUMMY
(schnellste) MODIFIER (Zugverbindung) TRAIN_CONNECTION)CONNECTION_INFO
((zwischen) BETWEEN ((Kobern) CITY_1 (Gondorf) CITY_2) CITY_MACRO
(und) CONNECTIVE ((Esslingen) CITY_1)CITY_MACRO) ROUTE
(((frühestens) MODIFIER (übermorgen)DAY_OF_WEEK) DATE_MACRO) DATE_INFO
(((um) AT
((acht)HOUR_OF_DAY (Uhr) HOUR (fünfzehn) MINUTES) TIME_MACRO)) TIME_INFO
```

A graphical representation of the example in Listing 15 where the leaf nodes represent the low-level semantic concepts is shown in Figure 8.9.

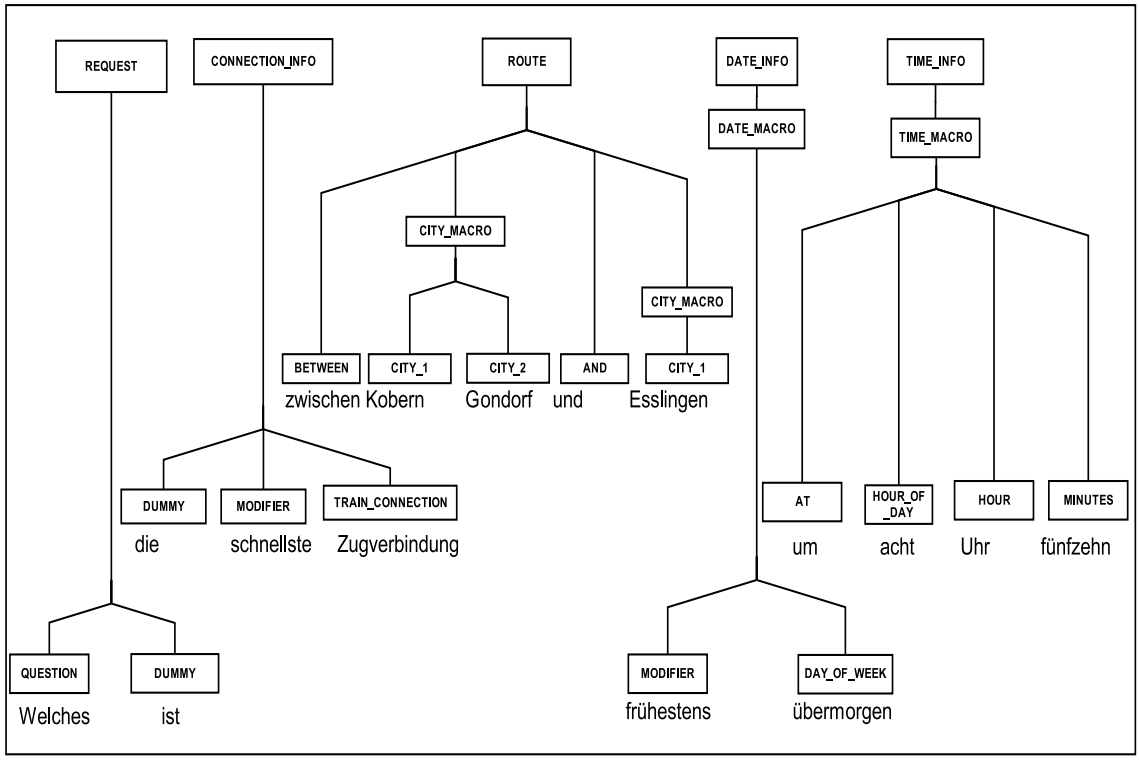

Figure 8.9: Example graphical representation of a detailed output of the hierarchical model

8.6.6 Model Complexity

A semantic labeling task would be very simple, if every word belongs to one and only one semantic class. However, a word can have multiple senses in different contexts. For instance, the word "to" has different senses in "I would like to", "to Frankfurt", "quarter to ten", etc. and thus should be labeled accordingly. Although one can think of various other ways to gauge the complexity of a model or a task, we measure complexity in two ways:

- The performance of an unconstrained model that defines only the semantic classes and the lexical items in each class
- The average number of possible labels that a model can assign to a word in the lexicon of the application.

It has been shown in Section 8.6.3.1 that an unconstrained ergodic flat-concept model gives 70.96% and 81.92% in F-measure showing that the German task is relatively easier than the English one because the Communicator corpus consists of spontaneous utterances as opposed to ERBA which consists of well-structured, read utterances.

The average number of labels that any word in the vocabulary of a given application domain can assume in the flat and the hierarchical models is given in Table 8.26.

Table 8.26: Average number of possible labels for a word in the flat and the hierarchical models

Model	Communicator (English)	ERBA (German)
Flat	1.3	1.1
Hierarchical	1.73	1.54

It can be observed in Table 8.26 that the English task is relatively more ambiguous than the German task. It can also be noted that the hierarchical model is more complex as different senses of a word in different contexts are introduced. For instance, phrases "September *the* nineteenth", "early *in the* morning", "twenty third *of* May", "*at* nine a. m.", "*at* j. f. k. *airport*", "*airport* hotel", etc. consist of context-sensitive versions of words such as $in, the, of, at, airport$, etc. that must be labeled according to the context of their use.

The values in Table 8.26 do not consider the higher level context; i.e., they do not distinguish between arrival, departure and other cities. However, the task requirement of the hierarchical model is higher as we would like to know whether a given date, time, city, etc. is ARRIVAL_DATE, DEPARTURE_TIME, ARRIVAL_LOC, etc. using the surrounding context. Hence, the complexity of the hierarchical model can be measured by computing the average number of times that a word which is tied to a sub-network occurs in different sub-structures. For instance, a city can occur in about five super-concepts in the domain of train information inquiries – in ARRIVAL_LOC, in DEPARTURE_LOC, in CONNECTION when the city is a connecting station, in ROUTE when referring to possible routes between two cities or in LOCATION when there is no further contextual information. This means every city can have five different possible labels in different contexts. Therefore, as a measure of the complexity of the hierarchical models, we compute the average number of possible tags that a tied word can have for the two application domains. This value is 5.61 for train inquiries domain and 2.92 for airline travel planning. This measure tells only the complexity of the model and does not mean that the German task is harder than the English task.

8.7 Evaluation of the Demonstration System

In the previous sections, we discussed the speech recognition and spoken language understanding components of our telephone-based spoken dialog system. We applied the acoustic, language and semantic models discussed above in real-time test settings where test subjects are invited to interact with the system to perform predefined information inquiry tasks. In the following section, we present the evaluation procedure used and discuss the results obtained.

Since user satisfaction is inherently subjective, the core of the evaluation procedure is based on collecting judgements from test users using a list of questions compiled based on the SASSI (Hone and Graham, 2001) questionnaire and the ITU-T Rec. P.851 (ITU_T Rec. P.851) recommendation. Complementary information are obtained from logged interaction parameters.

8.7.1 Example Interaction

We prepared two test-cases for each user (for a total of 20 test subjects) – one in English in the domain of airline travel planning and the other in German in the domain of train information inquiries. The first expected spoken input from the user is a choice of the preferred language – English or German. This input is used to recognize the preferred language and to estimate the gender of the speaker from the same acoustic input so as to load the corresponding semantic model and gender-dependent acoustic model for the rest of the dialog. Listing 16 shows an example interaction in the domain of airline travel planning.

Listing 16 Typical example interaction

```
System: Hello! My name is KEY. I provide service in English and German.
        Which one do you prefer?
User: English
System: Welcome to the Airline travel planning system of University
        Magdeburg! What is your travel plan?
User: I want to fly from San Francisco to Washington D. C. on December
      eleven.
System: Would you like to fly from Toronto to Washington D. C.?
User: no from San Francisco.
System: Would you like to fly from San Francisco to Washington D. C.?
User: yes
System: At what time do you want to fly?
User: three  p. m.
System: Would you like to fly on December eleven at three  p. m.?
User: yes
System: Do you have any airline preference?
User: Delta airlines
System: Do you want to fly on Delta Airlines?
User: yes
System: Just a second! I have flight four hundred and ninety five from San
        Francisco to Washington D. C. on Delta Airlines. Check-in time is
        at three  p. m. This is the only available flight. Please choose
        change for another flight, book to book the flight, or exit to quit.
User:   exit
System: Thank you for calling! Good-bye.
```

8.7.2 Subjective Evaluation

The questionnaire (in Appendix B) contains 40 questions; 38 of which are rated on a 5-point Likert scale shown in Figure 8.10. The value 5 corresponds to the most positive and 1 to the most negative response on the scale. Some of the statements were expressed positively and others negatively in order to minimize the possible bias of responses. "Strongly Agree" is the most positive response for positive statements while "Strongly Disagree" is the most positive response for negative statements as disagreement to a negative question expresses a positive attitude.

Figure 8.10: A 5-point Likert scale

For analysis purposes, we group the 38 questions into eight major categories based on what the questions are intended to elicit – task efficiency, speech input and output quality, reliability, cooperativity, dialog efficiency, user satisfaction, task ease and acceptability. The last two questions ask the users which of the two services they liked more and to rate their overall impression about the system on a continuous scale. In the following sections we present and discuss the results of the evaluation in terms of these categories.

8.7.2.1 Task Efficiency

Task efficiency consists of factors such as task success, completeness and clarity of the provided information, suitability of the system for the task and efficiency. The questions that are intended to elicit these information and the percentage of respondents by response category for each question are given in Table 8.27.

As can be seen in Table 8.27, the majority of the respondents agree on the completeness and clarity of the information provided by the system. However, it is not obvious to draw conclusions from the other two questions. Although more respondents show a positive attitude to these questions, the percentage of undecided respondents for each question is significant. Nevertheless, as can be seen in Figure 8.11 that depicts the mean of the responses for each question and the average of the means, one can see that task efficiency is well on the positive side of the scale. The bold horizontal line in Figure 8.11 shows the average of the means (3.66).

Table 8.27: Task efficiency: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
1(a): The system did exactly what I requested	0	45	30	25	0
1(b): The information provided by the system was clear	40	40	10	10	0
1(c): The provided information was complete	50	40	5	5	0
1(d): The system could efficiently provide information inquiry services	0	35	35	25	5

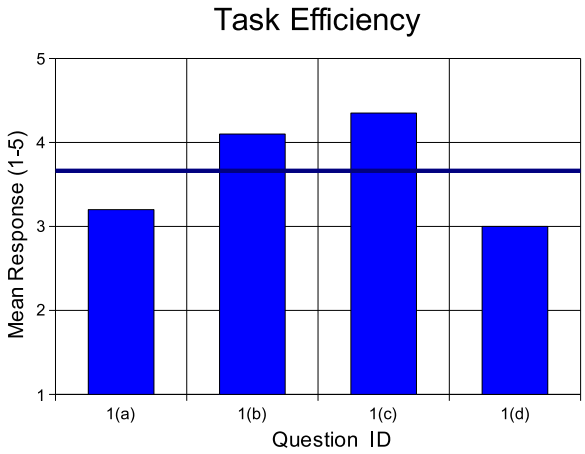

Figure 8.11: Task efficiency: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.2 Speech Input and Output Quality

Speech input and output quality comprises of factors that have to do with speech recognition performance, understanding ability of spoken requests, naturalness of synthesized speech and the speed of interaction. The questions that are intended to elicit these information and the percentage of respondents by response category for each question are given in Table 8.28.

Table 8.28: Speech input and output quality: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
2(a): I felt well understood by the system	0	5	35	40	20
2(b): I had to concentrate to acoustically understand the system	5	35	10	35	15
2(c): The system voice sounded natural	5	45	15	25	10
2(d): The system reacted too slowly	0	45	25	30	0

As can be noted, 60% of the respondents disagree with the statement "I felt well understood by the system". This question entails the performance of both speech recognition and speech understanding components of the system. Since the spoken language understanding unit takes the speech recognition result as its input, the output depends on the quality of the recognized input. From the logged information, we could see that the SLU unit (semantic model) labeled its input almost always correctly, however, there were significant number of recognition errors mainly in the German system. As can be seen from the objective evaluation in Section 8.7.3 and our observation, the performance of the system for the German service was suboptimal for reasons to be discussed in Section 8.7.2.10, while the performance of the system for the English service was impressive in most cases. The difference in performance of the two services seems to be the reason why a significant percentage (35%) of respondents are undecided on this question. Moreover, most users performed the English airline travel planning task first and the German train information inquiries task last which was in some cases unpleasant. The phenomenon called "recency effect", where the last experience has greater influence on the overall impression, might be the reason why some respondents disagree with the first question even when the English interaction was quite good.

As can be seen in Figure 8.12 which depicts the mean of responses for each question on a 5-point Likert scale, the average of the means for the speech input and output quality

category is 2.85 which is on the negative side of the scale. This can be attributed mainly to the weakness of the speech recognition model used for German service.

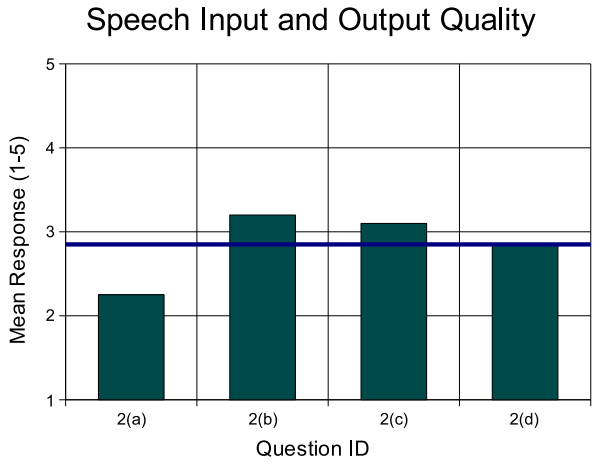

Figure 8.12: Speech input and output quality: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.3 Reliability

We define reliability as consisting of factors that are related to the ability of a system to perform the required task(s), frequency of errors and consistency of the system's behavior in different situations. The two questions that are intended to elicit these information and the percentage of respondents by response category for each question are given in Table 8.29

Table 8.29: Reliability: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
3(a): The system made many errors	10	40	30	20	0
3(b): The system is unreliable	5	5	35	40	15

As can be seen in Table 8.29, 50% of the respondent believe that the system made many errors while 30% are undecided. On the other hand, 55% of the respondents believe that the system is reliable while 35% are undecided. This sounds conflicting but it somehow tells the fact that the users could successfully complete the task even with recognition errors. The disparity in performance of the two services in English and German, may explain the high percentage of undecided respondents for the two questions. The average of the means, however, is 3.1 which is slightly on the positive side of the scale as can be seen in Figure 8.13.

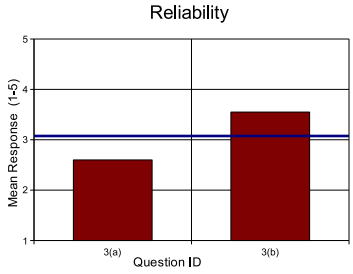

Figure 8.13: Reliability: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.4 Cooperativity

We group factors relating to ease of recovering from errors, reversibility of actions, helpfulness in case of confusion, error tolerance and human-like behavior into the category cooperativity. The questions that are intended to elicit these information are given in Table 8.30 along with the percentage of respondents by response category.

As can be seen in Table 8.30, most respondents (75%) agree that the system behaved in a cooperative way but 45% agree and 40% disagree on whether the system allowed them to easily recover from errors. On the other hand, 55% of the respondents believe that the system's behavior is not human-like while 30% of respondents are undecided on this question.

As can be seen in Figure 8.14 which depicts the mean of responses for each question on a 5-point Likert scale, the average of the means is 3.1 which is slightly on the positive side of the scale.

Table 8.30: Cooperativity: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree
4(a): I was able to recover easily from errors	10	35	15	40	-
4(b): The system behaved in a cooperative way	20	55	10	10	5
4(c): The system reacted like a human	-	15	30	35	20

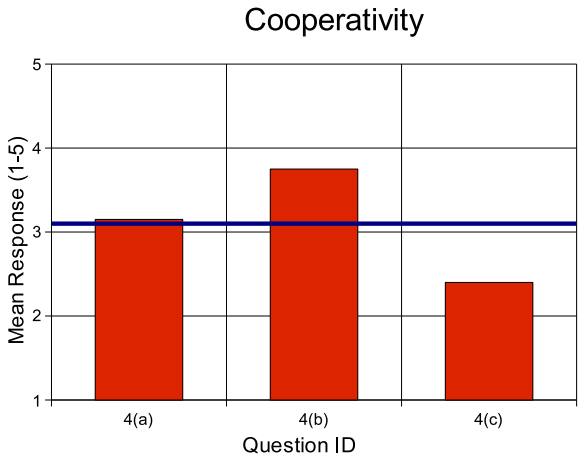

Figure 8.14: Cooperativity: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.5 Dialog Efficiency

We define dialog efficiency as a parameter that comprises of factors such as dialog flow, dialog symmetry, controllability, transparency of the system behavior, etc. The questions that are intended to elicit these information and the percentage of respondents by response category for each question are given in Table 8.31.

As can be observed, most respondents (95%) believe that they were not lost in the dialog flow while the remaining 5% are undecided, and 85% of the respondents always knew what the system expected from them. 65% of the respondents believe that the dialog was balanced between the system and themselves. 55% of the respondents judge

Table 8.31: Dialog efficiency: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
5(a): I got easily lost in the dialog flow	0	0	5	55	40
5(b): The dialog was irregular	0	50	20	25	5
5(c): I could direct the dialog as I wanted	0	20	30	30	15
5(d): The dialog was too long	5	5	35	45	10
5(e): The dialog quickly led to the desired aim	0	10	55	25	10
5(f): The dialog was balanced between me and the system	5	60	15	20	0
5(g): I always knew what to say to the system	5	55	10	30	0
5(h): I felt in control of the interaction with the system	5	25	40	25	5
5(i): I was not always sure what the system expected from me	0	10	5	50	35

the dialog length positively while 35% are undecided. On the other hand, 50% of the respondents judge the dialog as irregular while 20% are undecided.

As can be seen in Figure 8.15, which shows the mean of responses for each question on a 5-point Likert scale, the average of the means for the questions in dialog efficiency category is 3.31 which is well on the positive side of the scale.

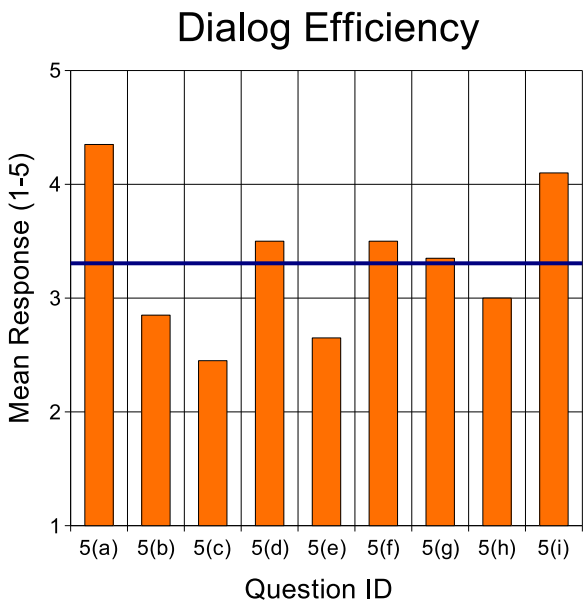

Figure 8.15: Dialog efficiency: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.6 User Satisfaction

We define user satisfaction as a set of factors that are related to the usefulness of the system, the pleasantness of the interaction, the friendliness of the system and the conformance of the system to user expectations. The questions that are believed to influence or are directly related to user satisfaction are listed in Table 8.32 with the percentage of respondents by response category.

As can be seen in Table 8.32, 75% of the respondents believe that the system is useful; 85% have a positive opinion about the friendliness of the system. 55% think the interaction was fun while 25% are undecided. 55% of the respondents voted that they are satisfied with the system while 20% are undecided. However, there seems to be room for improvement in terms of pleasantness of the system, and conformance of the systems reaction to user expectations.

As a remark, user satisfaction heavily depends on user attitude. Some users have a huge expectation which cannot be met by any current spoken dialog system which follows their natural spontaneous daily experience in human-to-human communication. However,

Table 8.32: User satisfaction: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
6(a): The interaction with the system was pleasant	0	40	10	40	10
6(b): I felt relaxed	5	25	35	30	5
6(c): The interaction was fun	15	40	25	15	5
6(d): The system is useful	25	50	10	10	5
6(e): The system is friendly	40	45	5	5	5
6(f): The system always reacted as expected	0	10	25	65	5
6(g): Overall, I am satisfied with the system	0	55	20	20	5

due to the limitations of the various technologies that constitute a spoken dialog system, there is often noticeable difference in performance between man-machine and human-to-human communication. On the other hand, some users underestimate a spoken dialog system and tend to speak only single-word utterances, very loudly which could be counterproductive as it may introduce recognition errors (due to so-called Lombard effect). As a result the interaction takes longer. Some think it is weird to talk to a machine with full sentences, etc. All these factors may lead to less user satisfaction. Nevertheless, it is interesting to see that the system is rated positively as can be seen in Figure 8.16 that shows the mean of responses for each question in this category on Likert scale. The average of the means of responses for this category is 3.26 which is on the positive side of the scale.

8.7.2.7 Task Ease

We define task ease as comprising of factors such as ease of use, learnability, flexibility, cognitive demand and comfort of interaction. The questions that are intended to elicit these information and the percentage of respondents by response category are shown in Table 8.33.

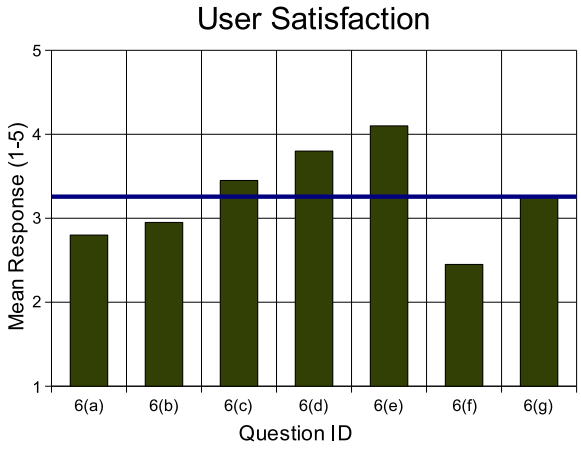

Figure 8.16: User satisfaction: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

Table 8.33: Task ease: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
7(a): The system is difficult to use	0	15	5	60	20
7(b): It is easy to learn to use the system	50	35	5	10	0
7(c): Information inquiry via speech was comfortable	10	50	15	10	15
7(d): The system is too inflexible	5	35	35	25	0
7(e): A high level of concentration is required when using the system	5	55	30	10	0

As can be observed in Table 8.33, 80% of the respondents find the system easy to use and 85% rate the system as easy to learn. 60% of the respondents think it is comfortable to do information inquiry via speech while 35% of the respondents don't think so. In

terms of flexibility and cognitive demand, the system is more or less rated on the negative side of the scale. In general, as can be seen in Figure 8.17 the average of the means of responses in these category on a 5-point Likert scale is 3.33 which is well on the positive side of the scale.

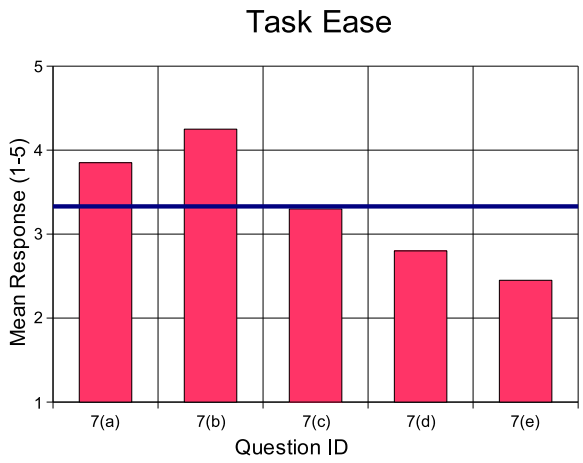

Figure 8.17: Task ease: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.8 Acceptability

We define acceptability as a set of factors that are related to helpfulness of the system for information inquiry, if it would be preferred to other methods, if the users would like to use the system again, etc. The questions in Table 8.34 are intended to elicit these information.

As can be seen in Table 8.34, 50% of the users believe that the system is helpful for information inquiry services, while 30% are undecided. 35% of the users would use the system again in the future while 40% of the respondents are undecided. On the other hand, a significant percentage of the respondents (60%) prefer to do information inquiry in a different way, while 35% are undecided. This clearly indicates that we could not beat the graphical user interface (GUI) yet with which most of the respondents are very familiar. However, this is understandable in the sense that it is not easy to take users away from their comfort zone – away from the system which they use on a daily basis.

Table 8.34: Acceptability: percentage of respondents by response category

Question	Strongly Agree (%)	Agree (%)	Undecided (%)	Disagree (%)	Strongly Disagree (%)
8(a): The system is not helpful for information inquiry services	0	20	30	30	20
8(b): I prefer to do information inquiry in a different way	20	40	35	5	0
8(c): I would use the system again in the future	5	30	40	20	5

As can be seen in Figure 8.18, the average of the means of responses in this category is 2.95 which is slightly to the negative side of the scale mainly because most users prefer to do information inquiry in a different way.

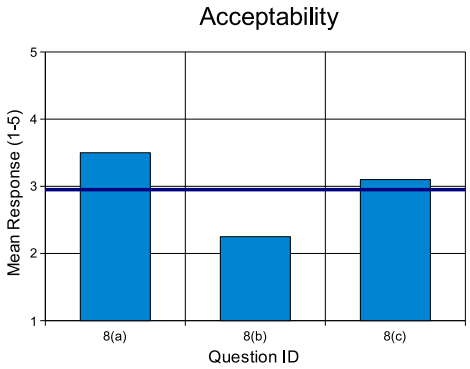

Figure 8.18: Acceptability: mean of responses for each question on a 5-point Likert scale where 5 corresponds to the most positive and 1 to the most negative response

8.7.2.9 Overall Impression

Finally, the users are asked to rate their overall impression about the system after performing airline travel planning task in English and train information inquiry task in German. This parameter was evaluated by the test subjects on a continuous rating scale from "bad"

to "excellent" (0-100) as shown in Figure 8.19. The categories on the continuous scale correspond to: bad (0-20), poor (20-40), fair (40-60), good (60-80), and excellent (80-100) (Xie et al., 2007).

Figure 8.19: A continuous rating scale

As can be seen in Figure 8.20, 55% of the test subjects rated their impression as "good", 30% rated it as "fair", while 15% of the users rated it as poor. In general, the mean rating is 56.45% which is in the range of fair to good. We believe this is a good impression for a first round evaluation.

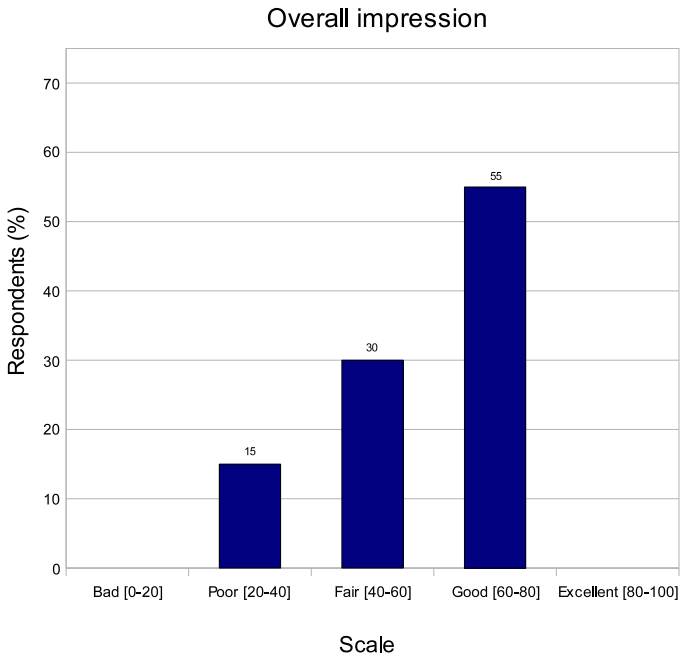

Figure 8.20: Overall impression of the interactions with the KEY system: on a continuous rating scale from "bad" to "excellent" (0-100)

8.7.2.10 Comparison of the English and the German Services

In order to elicit which of the two services is preferred, we included one question asking which of the two services was more likeable. As a result, 65% of the test subjects (13

subjects) liked the English system more while 10% of the test subjects (2 subjects) preferred the German service, 20% of the subjects (4 subjects) rated both services as equally good, while 5% (1 subject) didn't like both. Even though English is not the preferred language of the users, the system performed better with English models mainly because the acoustic model was robust enough to work with the typical accent of native German speakers.

On the other hand, the techniques we applied to use microphone recorded data for telephone-based spoken dialog system proved ineffective. We believe that there are three likely reasons for the suboptimal performance of the German acoustic model in real-time despite the reasonable performance reported in Section 8.3.2. First, the corpus used for the intended purpose was collected in a quiet office environment which is considerably different from the test setting. Even though, we used cepstral mean normalization to compensate for channel mismatch and additive noise, the performance was still poor. Second, the corpus contains little of the most commonly used dialog utterances such as "ja" and "nein" and hence are under-represented and were often misrecognized. Third, our system doesn't allow barge-in for the reasons that will be described in section 8.7.4 and when the user takes the turn too early before the system is finished with its turn, the input could be ignored or misrecognized. Besides, some of the grammar files used for the German system are quite big and take a while (about a second after the system prompt is finished) before the recognizer is ready for taking the spoken input. These issues need to be investigated further.

8.7.3 Objective Evaluation

In addition to the information we obtained using subjective evaluation, we obtain complementary information from interaction parameters extracted from the logged interaction data while the test subjects interacted with the system. The important information extracted for this purpose from the transcription of the recorded interaction are: average number of system turns, average number of user turns, average number of words per system turn, average number of words per user turn, average number of user correction turns, task completion rate and gender recognition error rate for each service (English and German).

Table 8.35: Interaction parameters

	Service	
Parameter	English	German
Total number of test calls	20	20
Avg. no. of system turns	13.45	19.8
Avg. no. of user turns	12.45	18.85
Avg. no. of words per system turn	12.73	9.65
Avg. no. of words per user turn	4.19	3.47
Avg. no. of user correction turns	2.95	7.85
Task completion rate	95%	80%
Gender recognition rate	90%	80%

As can be seen in Table 8.35, the average number of system and user turns for the German system is higher than that of the English service even though the task is relatively simpler. It can also be observed that the average number of user correction turns is quite high in the German version which is 7.85 as opposed to that of the English service which has an average correction turn of 2.95. This tells that the German acoustic model is not good enough for real-time use in a telephone-based spoken dialog system. The average number of words per user utterance is 4.19 and 3.47 for the German and English services, respectively. In fact, some of the confirmation user turns are single-word utterances such as "yes" or "ja".

The English service had a task completion rate of 95% while the German service had a task completion rate of 80%. The gender recognition system could correctly identify the gender of most of the female test subjects correctly but failed to correctly detect the gender of some male subjects from the first single-word utterance in which case the users were asked to hang up and dial again.

In general, the performance of the English service is observed to be much better than the German service.

8.7.4 Known Limitation

The main limitation of our telephone-based spoken dialog system is its inability to support barge-in due to the undesirable effect of acoustic echo where the system prompts played

through the earpiece of a telephone handset is captured by the microphone of the same handset. As a result, the prompt is captured as audio input by the telephony interface component and streamed to the speech recognizer. In order to deal with this effect, we turn the speech recognizer off when prompts are played and turn it on when input from the user is expected. This works fine but at the expense of barge-in and introduces a little bit of latency as the system prompt should finish before the speech recognizer is ready to process the next input.

There are echo cancelation algorithms that basically analyze outgoing audio data as well as incoming audio data in order to filter echo signals from incoming audio streams. This obviously introduces latency and takes some serious amount of system resources. Therefore, most of the time, echo cancelation is done in hardware such as server-grade ISDN boards, etc. Hence, the problem we are facing could be solved by either implementing an echo cancelation algorithm (which has its drawbacks, as discussed above) or using ISDN boards with echo cancelation capabilities.

8.8 Summary

In this chapter we described the data used, the methods employed, the experiments conducted and the results obtained in the various experiments. Analyses and discussions of the results were also given in the respective sections. The experiments conducted include automatic speech recognition, automatic gender identification, accent recognition, accent adaptation, spoken language understanding and evaluation of the demonstration system. The results obtained are promising while rooms for improvement and areas for further investigation have been identified.

Chapter 9

Conclusions and Recommendations

9.1 Conclusions

This thesis aimed at investigating the feasibility of building a robust, multi-domain, and multi-lingual telephone-based spoken dialog system framework that possesses sufficient robustness to carry out successful spoken language interactions in different application domains. To this end, we built the various components of a telephone-based spoken dialog system; namely, a telephony interface component, a speech recognition engine, a gender recognizer, a grammar component and a spoken language understanding unit. The required acoustic, language and semantic models are also carefully built. These are then integrated within a VoiceXML framework – ©OptimTalk (OptimSys, 2006). The integrated system can be used for multiple application domains and/or languages by switching between the required recognition resources in real-time. A complete description of the system is presented in Chapter 4 with some implementation details left out.

Robustness can be achieved through proper acoustic, language and semantic modeling, where robustness in acoustic modeling is of paramount importance for the overall success of a spoken dialog system. Human-speech recognition ability is little affected by channel mismatch, inter-speaker and intra-speaker variability, background noise, etc. However, these have a serious negative influence on the performance of an automatic speech recognizer. A robust speech recognizer should be able to cope with these problems and provide the required service even when the spoken input is unforeseen or degraded. An approach to obtain some level of robustness is to use domain and user-group depen-

dent acoustic models. To this end, we carefully built acoustic models while investigating better features and feature parameters for telephone-based spoken dialog system.

We built a Gaussian mixture model (GMM) based gender recognizer using Mel-frequency cepstral features (MFCC) with the 0^{th} coefficient as the energy term and the dynamic features which gave better performance than PLP and LPCC based systems. We opted for cepstral features for gender recognition because the most salient cue for distinguishing adult male and female speech (i.e. the fundamental frequency (F0)) is either missing or weak in telephone speech due to the band-limiting effect of the telephone channel. The gender recognizer is used to estimate the gender of a speaker from a spoken utterance so that gender-dependent acoustic models which perform better than a speaker-independent model could be used.

Since the target users of the system are native German speakers, the acoustic model that is trained on native English speech data is tailored to the particular vocal characteristics of German-accented English speakers. The use of a few number of maximum a posteriori (MAP) adaptation on top of maximum likelihood linear regression (MLLR) transformed models gives a tremendous boost in performance for each gender group. Moreover, multiple transforms where both mean and diagonal covariance are transformed is found to be more productive than a single global transform in our setup. We also demonstrated the feasibility of training an accent recognizer on native speech data of the target accent groups. The model trained on native German and US-English data can detect accent from a spoken English utterance with high accuracy using linear predictive coding cepstral coefficients (LPCC) and the energy term. Motivated by this success, we further investigated the use of cross-language accent adaptation where native German speech data is used to adapt the English acoustic model. This resulted in remarkable performance gain.

On the other hand, although widely used, simulating telephone quality speech from microphone recorded data by introducing the obvious effects of the telephone channel and using a small amount of telephone recorded data to further adapt a model trained on "simulated" data to the telephone channel did not seem to be effective. Despite the performance gain we observed after channel adaptation on telephone recorded test-set, the resulting model under-performed in real-time tests. This may be attributed to the considerable channel and acoustic mismatch between the training and test environments that could not be handled by the approaches we employed and the under-representation

of some frequently used dialog utterances in the training corpus. Further investigation in this regard is required.

In terms of language modeling, we tried to strike a balance between the conflicting requirements of usability and naturalness of interactions. Naturalness and freedom of expression may hinder usability and task completion (Pieraccini and Huerta, 2005) due to more recognition errors. Allowing users to say anything at any point in a dialog is too luxurious and is prohibitive as it entails more speech recognition errors and it makes error recovery difficult. Therefore, we made one reasonable assumption; i.e., telephone-based interactions are often task-oriented. Consequently, in order to execute a dialog about a task, a set of well-structured operations are required. It follows that it is practical to use domain and dialog-state dependent language models or grammars instead of a universal language model (or grammar) for the whole system.

The use of dialog state-specific language models instead of grammars in order to give more freedom to users resulted in relatively poor recognition performance mainly because of the insufficiency of training data to train bigram language models for each dialog state as the training data had to be split into a number of subsets. Therefore, we finally resorted to using comprehensive dialog state-specific grammars that can more or less give the required freedom without compromising usability. The users are allowed to provide more than one information at a time and can use universal commands like "start over" or "help" at any point.

Regarding semantic modeling, we described three different but interrelated HMM-based approaches to semantic concept labeling; namely, flat-concept, medium-level hierarchical and hierarchical models. We started with the flat-concept approach and incrementally extended it to encode more context at different levels of hierarchy by grouping semantically and hierarchically related low-level concepts into higher level structures using prior domain knowledge and training examples. The hierarchical models offer better ambiguity resolution ability, more predictive power and produce semantically richer information than the flat-concept model. Moreover, the hierarchical models are robust in that out-of-vocabulary words could be more correctly labeled using the surrounding context and can gracefully ignore semantically irrelevant speech recognition errors. This allows us to focus on content-bearing concepts to easily infer the meaning of what might have been said. Besides, the hierarchical models can robustly handle noisy input due to the natural phenomena of spontaneous speech such as hesitations, false starts, filled pauses,

etc. that introduce undesirable noise. In addition, the hierarchical models are easily extensible to include new requirements and can produce output at different levels of detail as required.

All the described approaches can be readily trained on unlabeled data with relatively less human supervision. The required additional human effort to design the proposed hierarchical models is obviously much less than the laborious and error-prone semantic annotation of the training data which would also require a detailed analysis of the application domains to define semantic labels and organize them into hierarchically structured concepts. To keep the human effort low, we implemented a model compiler that allows us to easily tune a model based on example sentences and prior domain knowledge. The success of our modeling approach relies mainly on the use of a priori commonplace domain knowledge to build an informed initial model that can further be trained using the EM algorithm. In order to account for unseen observations and out-of-vocabulary words we smooth transition and emission probabilities. The hierarchical model outperforms the flat-concept model and has been successfully used in our demonstration system.

Finally, once the required models are built and optimized, they are plugged into the telephone-based spoken dialog system framework and the performance of the system as a whole is evaluated with actual test users. Since user satisfaction is inherently subjective, the core of the evaluation method is based on collecting judgements from test users using a list of questions compiled based on the de-facto standards SASSI questionnaire and the ITU-T Rec. P.851. We also obtained complementary information from logged interaction parameters. We analyzed the responses of the test-users for the various questions in eight major categories based on what the questions are intended to elicit; namely, task efficiency, speech input and output quality, reliability, cooperativity, dialog efficiency, user satisfaction, task ease, and acceptability. The system is rated mostly on the positive side of a 5-point Likert scale. The overall impression of the test users after using the system for the two application domains in the two languages was evaluated on a continuous rating scale from "bad" to "excellent" (0-100). 55% of the users rated their impression as good while 30% rated it as fair which makes a big majority (85%). 15% of the test users rated their impression as poor. It is noteworthy, however, that there was clear disparity in the quality of the two services – as the German acoustic models were unsuitable for the task.

9.2 Recommendations

Future work could focus on a number of issues concerning acoustic, language and semantic modeling. An interesting extension of the semantic model we proposed in this thesis could be to introduce a feature that will enable the system to automatically learn out-of-vocabulary (OOV) words and unseen transitions as and when they occur. In order to avoid learning the wrong information in the case of wrong semantic labeling of OOV words, a confidence measure could be computed to gauge the likelihood of the OOV word belonging to the hypothesized semantic concept. If the computed value is less than a predefined threshold, the system could ask the user to confirm if the hypothesized semantic concept for the new word is correct before updating the model. Otherwise, the system could automatically update the model without the intervention of the user. Using confidence measures at acoustic level may also be useful to reduce the number of necessary confirmation turns.

One drawback of the approach we used in building the proposed semantic modeling, is that the classification of vocabulary items into the identified set of semantic classes is done manually. An automatic approach to do this is desirable to minimize the human effort involved in this regard.

The correct identification of a speaker's gender is indispensable so that acoustic models tailored to each gender group can be used to achieve improved performance. Hence, the performance of the gender recognizer we built in this thesis may be further improved using other features in addition to cepstral features to obtain performance as close to that of humans as possible. Besides, a nice to have feature in a spoken dialog system is online speaker adaptation where the first few utterances of a speaker are used to adapt the acoustic model for a duration of a dialog. This is typically useful for long interactions.

Appendix A

List of Semantic Classes

Airline Travel Planning Domain

Listing 17 List of low-level semantic classes in the domain of airline travel planning

```
AIRLINE_NAME, AIRLINE_QUALIFIER, AIRPORT_QUALIFIER, AIRPORT_NAME,
AIRPORT_TYPE, AMOUNT_OF_MONEY, CONNECTIVE, BETWEEN, CAR, CAR_TYPE,
PICK_UP, DROP_OFF, RENTAL, RENTAL_COMPANY, CITY_P1, CITY_P2,
CITY_P3, CITY_QUALIFIER, COMMAND, COUNTRY, PREFERENCE, DAY_OF_MONTH,
DATE_QUALIFIER, DAY_OF_ WEEK, MONTH, YEAR, DEPARTURE, ARRIVAL, DUMMY,
FINISHED, FLIGHT_QUALIFIER, FLIGHT_CLASS, FLIGHT_NUMBER, FLIGHT_TYPE,
AT, FROM, TO, HOTEL_QUALIFIER, HOTEL_TYPE, HOTEL_ROOM, ROOM_TYPE,
ON, IN, HOUR_OF_DAY, MINUTES, AMPM, PERIOD_OF_DAY, TIME_QUALIFIER,
ID, ID_NUMBER, USER_NAME, ITINERARY, PLACE_INDICATOR, NEXT, MODIFIER,
OPTION, PRICE, FARE_CLASS, RETURN, QUANTITY, RESERVATION, REQUEST,
STATE, SPELT_CITY, SPELT_AIRPORT, QUALIFIER, SEGMENT, YES, NO, PLANE,
NUMBER, TICKET, QUESTION, TRAVEL, TRAVEL_TYPE, INFORMATION, STREET
```

Train Inquiries Domain

Listing 18 List of low-level semantic classes in the domain of train inquiries

```
CITY_PRE, CITY_1, CITY_2, CITY_POST, PLACE_INDICATOR, IN, BETWEEN,
DIRECTION, DEPARTURE, DEPARTURE_INFO, ARRIVAL, ARRIVAL_INFO, FROM, TO,
BACK, DAY_OF_MONTH, MONTH, HOLIDAY, DAY_OF_WEEK, DAY_TYPE, ON, NUMBER_OF,
DAYS, WEEK, MINUTES, HOUR, HOUR_OF_DAY, TIME_SPECIFIER, PERIOD_OF_DAY,
AROUND, QUALIFIER, TIME_QUALIFIER, SERVICE, QUESTION, MODIFIER, TRAVEL,
AT, TICKET, TRAIN, TRAIN_CLASS, TRAIN_TYPE, PRICE_ TYPE, DELAY, DUMMY,
TIME_FREQUENCY, TRAIN_CONNECTION, TRAIN_CHANGE, IN_TRAIN_SERVICE, YES, NO,
CONNECTIVE
```

Appendix B

The Questionnaire

Adapted from (Möller et al., 2007)

1. Task Efficiency
   (a) The system did exactly do what I requested.
   ("Das System tat genau das, was ich von ihm verlangte.")
   ☐ strongly agree ☐ agree ☐ undecided
   ☐ disagree ☐ strongly disagree
   (b) The information provided by the system was clear.
   ("Die vom System gelieferten Informationen waren klar und deutlich.")
   ☐ strongly agree ☐ agree ☐ undecided
   ☐ disagree ☐ strongly disagree
   (c) The provided information was complete.
   ("Die gelieferten Informationen waren vollständig.")
   ☐ strongly agree ☐ agree ☐ undecided
   ☐ disagree ☐ strongly disagree
   (d) The system could efficiently provide information inquiry services.
   ("Mit dem System lassen sich gewünschte Informationen effizient erfragen")
   ☐ strongly agree ☐ agree ☐ undecided
   ☐ disagree ☐ strongly disagree

2. Speech Input and Output Quality
   (a) I felt well understood by the system.
   ("Ich fühlte mich gut vom System verstanden.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree
   (b) I had to concentrate to acoustically understand the system.
   ("Ich musste mich konzentrieren, um das System akustisch zu verstehen.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree
   (c) The system voice sounded natural.
   ("Die Stimme des Systems klang natürlich.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree
   (d) The system reacted too slowly.
   ("Das System reagierte zu langsam.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree

3. Reliability
   (a) The system made many errors.
   ("Das System machte viele Fehler.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree
   (b) The system in unreliable.
   ("Das System ist unzuverlässig.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree

4. Cooperativity
   (a) I was able to recover easily from errors.
   ("Ich konnte auftretende Fehler leicht beheben.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree

(b) The system behaved in a cooperative way.
("Das System verhielt sich kooperativ.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(c) The system reacted like a human.
("Das System reagierte wie ein Mensch.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

5. Dialog Efficiency

(a) I got easily lost in the dialog flow.
("Ich konnte leicht den Gesprächsfaden verlieren.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(b) The dialog was irregular.
("Das Gespraĺch verlief holprig.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(c) I could direct the dialog as I wanted.
("Ich konnte das Gespräch wie gewünscht lenken.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(d) The dialog was too long.
("Das Gespräch war zu lang.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(e) The dialog quickly led to the desired aim.
("Das Gespräch führte schnell zum gewünschten Ziel.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(f) The dialog was balanced between me and the system.
("Die Gesprächsanteile waren gleich verteilt zwischen mir und dem System.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(g) I always knew what to say to the system.
("Ich wusste zu jeder Zeit, was ich dem System sagen konnte.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(h) I felt in control of the interaction with the system.
("Ich hatte has Gefühl, das ich die Kontrolle über das System hatte, während ich es benutzte.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(i) I was not always sure what the system expected from me.
("Ich wusste nicht immer, was das System von mir verlangte.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

6. User Satisfaction

(a) The interaction with the system was pleasant.
("Die Interaktion mit dem System war angenehm.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(b) I felt relaxed.
("Ich fühlte mich entspannt.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(c) The interaction was fun.
("Die Interaktion hat Spaß gemacht.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(d) The system is useful.
("Das System ist nützlich.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(e) The system is friendly.
("Das System ist freundlich.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(f) The system always reacted as expected.
("Das System reagierte immer wie erwartet.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(g) Overall, I am satisfied with the system.
("Ich bin insgesamt mit dem System zufrieden.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

7. Task Ease

(a) The system is difficult to use.
("Das System lässt sich nur schwer bedienen.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(b) It is easy to learn to use the system.
("Die Benutzung des Systems lässt sich leicht erlernen.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(c) Information inquiry via speech was comfortable.
("Die Anfrage von Informationen mittels Sprache war komfortabel.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(d) The system is too inflexible.
("Das System ist zu unflexibel.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

(e) A high level of concentration is required when using the system.
("Ich musste mich sehr auf die Interaktion mit dem System konzentrieren.")
□ strongly agree □ agree □ undecided
□ disagree □ strongly disagree

8. Acceptability
   (a) The system is not helpful for information inquiry services.
   ("Das System ist nicht hilfreich für Informationen Anfrage Dienstleistungen.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree
   (b) I prefer to do information inquiry in a different way.
   ("Ich würde die Informationen lieber auf eine andere Weise beschaffen.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree
   (c) I would use the system again in the future.
   ("Ich würde das System in Zukunft wieder benutzen.")
   □ strongly agree □ agree □ undecided
   □ disagree □ strongly disagree

9. Which of the two services did you like more?
   □ The English Airline Travel Planning System.
   □ The German Train Information Inquiry System.
   □ Both are equally good.
   □ I didn’t like both.

10. Overall impression of the interaction with the KEY system.
    ("Gesamteindruck der Interaktion mit dem KEY System.")

Authored Publications

List of Authored Publications

[1] Kinfe Tadesse Mengistu, Mirko Hannemann, Tobias Baum, and Andreas Wendemuth. Hierarchical HMM-based Semantic Concept Labeling Model. In: *Proceedings of IEEE Workshop on Spoken Language Technology (SLT)*, pages 57-60, 2008.

[2] Kinfe Tadesse Mengistu and Andreas Wendemuth. Accent and Channel Adaptation for Use in a Telephone-based Spoken Dialog System. In: Lecture Notes In Artificial Intelligence, Vol. 5246, *Proceedings of the* 11^{th} *International Conference on Text, Speech, and Dialogue (TSD)*, pages 403-410, 2008.

[3] Kinfe Tadesse Mengistu and Andreas Wendemuth. Towards User Group Dependent Acoustic Models. In: *Proceedings of 19. Konferenz Elektronische Sprachsignalverarbeitung (ESSV)*, pages 64-71, 2008.

[4] Kinfe Tadesse Mengistu, Mirko Hannemann, Tobias Baum and Andreas Wendemuth. Using Prior Domain Knowledge to Build Robust HMM-Based Semantic Tagger Trained on Completely Unannotated Data. In: *Proceedings of the Workshop on Prior Knowledge for Text and Language (PKTL)*, pages 31-36, 2008.

[5] Kinfe Tadesse Mengistu, Marcel Katz and Andreas Wendemuth. Acoustic Modeling for a Speaker-Independent Telephone-Based Spoken Dialog System. In: *Proceedings of the* 12^{th} *International Conference on Speech and Computer (SPECOM)*, pages 469-474, 2007.

[6] Kinfe Tadesse Mengistu, Martin Schafföner and Andreas Wendemuth. Gender Recognition and Gender-Based Acoustic Model Adaptation for Telephone-Based Spoken Dialog System. In: *Proceedings of 18. Konferenz Elektronische Sprachsignalverarbeitung (ESSV)*, pages 68-75, 2007.

[7] Kinfe Tadesse Mengistu and Andreas Wendemuth. Telephone-Based Spoken Dialog System Using HTK-based Speech Recognizer and VoiceXML. *Proceedings of the* 33^{rd} *German Annual Conference on Acoustics (DAGA)*, pages 625-626, 2007.

[8] Kinfe Tadesse Mengistu, Mirko Hannemann and Andreas Wendemuth. Using More Context in HMM-Based Semantic Tagging Model. To appear in: *Proceedings of Research Workshop on Emotion-, Speech-, and Face Recognition with Advanced Classifiers*, A. Wendemuth and H.-G. Meier (eds.), 2009.

Co-authored Publication

[1] Bogdan Vlasenko, Björn Schuller, Kinfe Tadesse Mengistu, Gerhard Rigoll and Andreas Wendemuth. Balancing Spoken Content Adaptation and Unit Length in the Recognition of Emotion and Interest. In: *Proceedings of the Interspeech 2008 Incorporating SST 2008*, pages 805-808, 2008.

References

The number(s) at the end of each reference refer to the page(s) where the reference has been cited in the thesis

W. Abdulla and N. Kasabov. Improving Speech Recognition Performance Through Gender Separation. *Artificial Neural Networks and Expert Systems International Conference*, pages 218–222, 2001. 16

J. Allen and C. R. Perrault. Analyzing Intentions in Dialogues. *Artificial Intelligence*, 15 (3):143–178, 1980. 105

J. F. Allen. *A Plan-Based Approach to Speech Act Recognition.* PhD thesis, University of Toronto, 1979. 103, 105

J. F. Allen, B. W. Miller, E. K. Ringger, and T. Sikorski. A Robust System for Natural Spoken Dialogue. *Proceedings of the 1996 Annual Meeting of the Association for Computational Linguistics*, pages 62–70, 1996. 11

L. D. Alsteris and K. K. Paliwal. Evaluation of the Modified Group Delay Feature for Isolated Word Recognition. *International Symposium on Signal Processing and Applications*, pages 715–718, 2005. 123

D. E. Appelt. *Planning English Sentences*. University Press, Cambridge, 1985. 105

L. M. Arslan and J. H. L. Hansen. Frequency Characteristics of Foreign Accented Speech. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 1123–1126, 1997. 18, 135

L. M. Arslan and J. H. L. Hansen. Language accent classification in American English. *Speech Communication*, 18:353–367, 1996. 18

B. S. Atal. Effectiveness of Linear Prediction Characteristics of the Speech Wave for Automatic Speaker Identification and Verification. *The Journal of the Acoustical Society of America*, 55(6):1304–1312, 1974. 26, 57

B. S. Atal and S. L. Hanauer. Speech Analysis and Synthesis by Linear Prediction of the Speech Wave. *The Journal of the Acoustical Society of America*, 50(2):637–655, 1971. 26, 57

H. M. Aust, F. S. Oerder, and V. Steinbiss. The Philips Automatic Train Timetable Information System. *Speech Communication*, 17:249–262, 1995. 10

J. L. Austin. *How to do Things with Words*. Oxford Unviersity Press, Oxford, 1962. 99

C. Baber. Developing Interactive Speech Technology. In C. Baber and J. M. Noyes, editors, *Interactive Speech Technology: Human Factors Issues in the Application of Speech Input/Output to Computers*, pages 1–18. Taylor & Francis, Inc., Bristol, PA, 1993. 98

L. Bahl, F. Jelinek, and R. Mercer. A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, PAMI-5(2):179–190, 1983. 118

R. J. Baken and R. F. Orlikoff. *Clinical Measurement of Speech and Voice*. Singular Publishing Group, Thompson Learning, San Diego, CA, 2^{nd} edition, 2000. 16

L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970. 61

S. Bennacef, L. Devillers, S. Rosset, and L. Lamel. Recent Developments in Spoken Language Sytems for Information Retrieval. *ESCA Workshop on Spoken Dialog Systems*, pages 17–20, 1995. 10

K. Berkling, M. Zissman, J. Vonwiller, and C. Cleirigh. Improving Accent Identification Through Knowledge of English Syllable Structure. *Proceedings of the* 5^{th} *International Conference on Spoken Language Processing*, pages 89–92, 1998. 18

N. O. Bernsen and L. Dybkjær. The DISC Project. *ELRA Newsletter*, 2(2):6–8, 1997. 10

D. Bobrow, R. Kaplan, M. Kay, D. Norman, H. Thompson, and T. Winograd. GUS, a Frame-Driven Dialog System. *Artificial Intelligence*, 8(2):155– 173, 1977. 9

T. H. Bui. Multimodal dialogue management - state of the art. Technical Report TR-CTIT-06-01, Centre for Telematics and Information Technology, University of Twente, 2006. 105

H. Bunt. Context and dialogue control. *THINK Quarterly*, 3:19–31, 1994. 99

E. Charniak. Immediate-Head Parsing for Language Models. *Proceedings of the* 39^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 116–123, 2001. 20, 77

C. Chelba and F. Jelinek. Structured Language Modeling. *Computer Speech and Language*, 14(4):283–332, 2000. 20, 77

H. H. Clark and E. R. Schaefer. Contributing to Discourse. *Cognitive Science*, 13:259–294, 1989. 99

P. R. Cohen and H. J. Levesque. Rational Interaction as the Basis for Communication. In P. R. Cohen, J. Morgan, and M. E. Pollack, editors, *Intentions in Communication*. M.I.T. Press, Massachusetts, 1990. 105

P. R. Cohen and C. R. Perrault. Elements of a Plan Based Theory of Speech Acts. *Cognitive Science*, 3(3):177–212, 1979. 103, 105

K. H. Davis, R. Biddulph, and S. Balashek. Automatic Recognition of Spoken Digits. *Journal of the Acoustical Society of America*, 24(6):627–642, 1952. 12

S. B. Davis and P. Mermelstein. Comparison of Parametric Representation for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions onf Acoustics, Speech, Signal Processing*, 1980. 26, 57

J. Dowding, R. Moore, F. Andry, and D. Morgan. Interleaving syntax and semantics in an efficient bottom-up parser. *Proceedings of* 32^{nd} *Annual Meeting of the Association for Computational Linguistics*, pages 110–116, 1994. 19, 76

B. Edgar. *The VoiceXML Handbook: Understanding and Building the Phone-Enabled Web*. CMP Books, New York, 2001. 30, 32

D. Elworthy. Does Baum-Welch Re-estimation Help Taggers? *Proceedings of the* 4^{th} *Conference on Applied Natural Language Processing*, pages 53–58, 1994. 145, 150

H. Erdogan, R. Sarikaya, Y. Gao, and M. Picheny. Semantic Structured Language Models. *Proceedings of the International Conference on Spoken Language Processing*, pages 933–936, 2002. 20, 77

Z. Fang and Z. Guoliang. Integrating the Energy Information into MFCC. *Proceedings of the* 6^{th} *International Conference on Spoken Language Processing*, 1:389–392, 2000. 123

G. Ferguson and J. F. Allen. TRIPS: An Integrated Intelligent Problem-Solving Assistant. *In Proceedings of the* 15^{th} *National Conference on Artificial Intellegence*, pages 567–572, 1998. 11

G. Fielding and P. Hartley. The Telephone: a Neglected Medium. In A. Cashdan and M. Jordan, editors, *Studies in Communication*. Basil Blackwell, Oxford, 1987. 104

S. Fine, Y. Singer, and N. Tishby. The Hierarchical Hidden Markov Model: Analysis and Applications. *Machine Learning*, 32(1):41–62, 1998. 20, 77

P. Fung and W. K. Liu. Fast Accent Identification and Accented Speech Recognition. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 221–224, 1999. 17

M. J. F Gales. Maximum Likelihood Linear Transformations for HMM-based Speech Recognition. *Computer Speech and Language*, 12(2):75–98, 1998. 72

J. L. Gauvain and C.-H. Lee. Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains. *IEEE Transactions on Speech and Audio Processing*, 2:291–298, 1994. 14, 73

J. Glass, G. Flammia, D. Goodine, M. Phillips, J. Polifroni, S. Sakai, S. Seneff, and V. Zue. Multilingual Spoken-language Understanding in the MIT Voyager system . *Speech Communication*, 17:1–18, 1995. 11

B. Gold and N. Morgan. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. John Wiley and Sons, Inc, New York, 2000. 26, 57

B. F. Green, A. K. Wolf, C. Chomsky, and K. Laughery. Baseball: an Automatic Question Answerer. *Computers and Thoughts*, pages 207–216, 1963. 9

J. H. L. Hansen and L. M. Arslan. Foreign Accent Classification Using Source Generator Based Prosodic Features. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 836–839, 1995. 14, 18

H. Harb and L. Chen. Voice-based Gender Identification in Multimedia Applications. *Journal of Intelligent Information Systems*, 24(2):179–198, 2005. 16

X. He and Y Zhao. Model Complexity Optimization for Non-native English Speakers. *Proceedings of Eurospeech*, 2:1461–1464, 2001. 136

Y. He and S. Young. Semantic Processing using the Hidden Vector State Model. *Computer Speech and Language*, 19(1):85–106, 2005. 19, 20, 77

C. T. Hemphill, J. Godfrey, and G. R. Doddington. The ATIS Spoken Language Systems Pilot Corpus. *Proceedings of DARPA Speech and Natural Language Workshop*, pages 96–101, 1990. 3, 10

H. Hermansky. Perceptual Linear Predictive (PLP) Analysis of Speech. *The Journal of The Acoustical Society of America*, 87(4):1738–1752, 1990. 26, 57, 59

J. M. Hillenbrand, L. A. Getty, M. J. Clark, and K. Wheeler. Acoustic Characteristics of American English Vowels. *Journal of the Acoustical Society of America*, 97:3099–3111, 1995. 15, 67

K. S. Hone and R. Graham. Subjective Assessment of Speech-System Interface Usability. *Proceedings of the* 7^{th} *European Conference on Speech Communication and Technology*, pages 2083–2086, 2001. 32, 112, 155

C. Huang, E. Chang, and T. Chen. Accent Issues in Large Vocabulary Continuous Speech Recognition. Technical Report MSR-TR-2001-69, Microsoft Research China, 2001a. 14, 18, 66

X. Huang, A. Acero, and H-W. Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall PTR, 1^{st} edition, 2001b. 28, 64, 73, 123

ISO9241-110. ISO 9241-110 Ergonomics of Human-System Interaction Part 110: Dialogue Principles, 2006. 107

F. Itakura. Minimum Prediction Residual Applied to Speech Recognition. *IEEE Transactions on Acoustics, Speech, Signal Processing*, 1:67–72, 1975. 13

F. Itakura and S. Saito. Analysis Synthesis Telephony based upon the Maximum Likelihood Method. *Proceedings of the* 6^{th} *International Congress on Acoustics*, pages C17–C20, 1968. 57, 59

ITU_T Rec. P.851. Subjective Quality Evaluation of Telephone Services Based on Spoken Dialog Systems, 2003. 32, 112, 155

F. Jelinek. Continuous Speech Recognition Using Statistical Methods. *Proceedings of IEEE*, 64(4):532–556, 1976. 61, 62

F. Jelinek and R. L. Mercer. Interpolated Estimation of Markov Source Parameters From Sparse Data. *In Proceedings of the Workshop Pattern Recognition in Practice*, pages 381–397, 1980. 80

B. H. Juang and L. R. Rabiner. Automatic Speech Recognition: Brief History of the Technology. *Encyclopedia of Language and Linguistics*, 2005. 13

J.-C. Junqua and J.-P. Haton. *Robustness in Automatic Speech Recognition: Fundamentals and Applications*. Kluwer Academic Publishers, Norwell, MA, 1995. 3, 25, 60, 61, 63

D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. Pearson Education, 2008. 56, 64, 65, 99, 118

S. M. Katz. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401, 1987. 80

C. J. Leggetter and P. C. Woodland. Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density Hidden Markov Models. *Computer Speech and Language*, 9:171–185, 1995a. 14, 72

C. J. Leggetter and P. C. Woodland. Flexible Speaker Adaptation Using Maximum Likelihood Linear Regression. *Proceedings of Eurospeech*, pages 1155–1158, 1995b. 71, 72

E. Levin, S. Narayanan, R. Pieraccini, K. Biatov, E. Bocchieri, G. Di Fabbrizio, W. Eckert, S. Lee, A. Pokrovsky, M. Rahim, P. Ruscitti, and M. Walker. The AT&T-DARPA Communicator Mixed-initiative Spoken Dialog System. *Proceedings of the* 6^{th} *International Conference on Spoken Language Processing*, 2:122–125, 2000. 10

S. C. Levinson. *Pragmatics*. Cambridge University Press, 1983. 98

G. J. Lidstone. Note on the General Case of the Bayes-Laplace Formula for Inductive or a Posteriori Probabilities. *Transactions of the Faculty of Actuaries*, 8:182–192, 1920. 80

C. E. Linke. A Study of Pitch Characteristics of Female Voices and Their Relationship to Vocal Effectiveness. *Folia Phoniatrica*, 25:173–85, 1973. 15, 67

S. E. Linville and H. B. Fisher. Acoustic Characterisics of Perceived versus Actual Vocal Age in Controlled Phonation by Adult Females. *Journal of the Acoustical Society of America*, 78:40–48, 1985. 15, 67

L. Liporace. Maximum likelihood estimation for multi-variate observations of markov sources. *IEEE Transactions on Information Theory*, 28(5):729–734, 1982. 63

W. K. Liu and P. Fung. MLLR-Based Accent Model Adaptation without Accented Data. *Proceedings of the* 6^{th} *International Conference on Spoken Language Processing*, pages 738–741, 2000. 15

B. T. Lowerre. *The Harpy Speech Recognition System*. PhD thesis, Ph.D. Thesis, Carnegie Melon University, 1976. 66

J. Makhoul. Linear Prediction: a Tutorial Review. *Proceedings of IEEE*, 63(5):561–580, 1975. 59

M. F. McTear. *Spoken Dialogue Technology: Toward the Conversational User Interface*. Springer Verlag, London, 2004. 9, 10

K. T. Mengistu, M. Hannemann, T. Baum, and A. Wendemuth. Using Prior Domain Knowledge to Build Robust HMM-Based Semantic Tagger Trained on Completely Unannotated Data. *Proceedings of the Workshop on Prior Knowledge for Text and Language (PKTL)*, pages 31–36, 2008a. xi, 89

K. T. Mengistu, M. Hannemann, T. Baum, and A. Wendemuth. Hierarchical HMM-based Semantic Concept Labeling Model. *Proceedings of IEEE Workshop on Spoken Language Technology (SLT)*, pages 57–60, 2008b. xii, 90, 92

F. Metze, J. Ajmera, R. Englert, U. Bub, F. Burkhardt, J. Stegmann, C. Müller, R. Huber, B. Andrassy, J. G. Bauer, and B. Littel. Comparison of Four Approaches to Age and Gender Recognition for Telephone Applications. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1089–1092, 2007. 17

S. Miller, R. Bobrow, R. Ingria, and R. Schwartz. Hidden Understanding Models of Natural Language. *Proceedings of the Association of Computational Linguistics*, pages 25–32, 1994. 19, 20, 76, 77

S. Möller. *Quality of Telephone-Based Spoken Dialogue Systems*. Springer, New York, 2005. 111

S. Möller, P. Smeele, H. Boland, and J. Krebber. Evaluating Spoken Dialogue Systems According to De-facto Standards: A Case Study. *Computer Speech & Language*, 21: 26–53, 2007. 111, 112, 181

R. D. Mori and F. Brugnara. *Survey of the State of the Art in Human Language Technology: HMM Methods in Speech Recognition*. Cambridge University Press, New York, 1997. 27

T. Murry and S. Singh. Multidimensionsl Analysis of Male and Female Voices. *Journal of the Acoustical Society of America*, 68:1294–1300, 1980. 15, 67

K. Nagata, Y. Kato, and S. Chiba. Spoken Digit Recognizer for Japanese Language. *NEC Research and Development*, 1963. 13

H. J. Nock, M. J. F. Gales, and S. J. Young. A Comparative Study of Methods for Phonetic Decision-Tree State Clustering. *Proceedings European Conference on Speech Communication and Technology*, pages 111–114, 1997. 124

L. Norskog. SoX: Sound Exchange. `http://sox.sourceforge.net/`; last accessed February 27, 2009, 1995. 117

OptimSys. OptimTalk Technology. OptimSys, s.r.o., `http://www.optimsys.eu/`; last accessed February 27, 2009, 2006. 29, 36, 39, 173

E. den. Os, B. Lou, L. Lamel, and P Baggia. Overview of the ARISE Project. *Proceedings of EUROSPEECH*, pages 1527–1530, 1999. 10

D. O'Shaughnessy. *Speech Communications: Human and Machine*. IEEE Press, 2^{nd} edition, 2000. 121

K. K. Paliwal. On the use of Filter-Bank Energies as Features for Robust Speech Recognition. *International Symposium on Signal Processing and Applications*, 2:641 – 644, 2005. 124

E. S. Parris and M. J. Carey. Language Independent Gender Identification. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 685–688, 1996. 16

J. Peckham. A New Generation of Spoken Dialogue Systems: Results and Lessons from the SUNDIAL Project. *Proceedings of the* 3^{rd} *European Conference on Speech Communication and Technology*, pages 33– 40, 1993. 10

R. Pieraccini and J. Huerta. Where Do We Go From Here? Research and Commercial Dialog Systems. 6^{th} *SIGdial Workshop on Discourse and Dialogue*, pages 1–10, 2005. 175

R. Pieraccini and E. Levin. A Learning Approach to Natural Language Understanding. *Proceedings of the NATO Advanced Study Institute on New Advances and Trends in Speech Recognition and Coding*, 1:261–279, 1993. 11, 19, 76, 81

S. D. Pietra, M. Epstein, S. Roukos, and T. Ward. Fertility Models for Statistical Natural Language Understanding. *Proceedings of the* 35^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 168–173, 1997. 81

P. Price. Evaluation of Spoken Language System: The ATIS Domain. *DARPA Speech and Natural Language Workshop*, pages 91–95, 1990. 19

L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):275–285, 1989. 61, 62

L. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, New Jersey, 1993. 59, 61, 62, 125

A. I. Rudnicky, C. Bennett, A. W. Black, A. Chotimongkol, K. Lenzo, A. Oh, and R. Singh. Task and Domain Specific Modelling in the Carnegie Mellon Communicator System. *Proceedings of the* 6^{th} *International Conference on Speech and Language Processing*, 2, 2000. 10

H. Sacks. *Lectures on Conversation*. Blackwell, Oxford, 1992. 98

H. Sacks, E. Schlegoff, and G. Jefferson. A Simple Systematics for the Organization of Turn Taking for Conversation. In J. Schenkin, editor, *Studies in the Organization of Conversational Interaction*. Academic Press, New York, 1974. 98

J. Sakai and S. Doshita. The Phonetic Typewriter, Information Processing 1962. *Proceedings of IFIP Congress*, 1962. 13

H. Sakoe and S. Chiba. Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE Transactions on Acoustics, Speech, Signal Processing*, 1:43–49, 1978. 13

E. A. Schegloff and H. Sacks. Opening up Closings. *Semiotica*, 8:289–327, 1973. 98

J. R. Searle. The Classification of Illocutionary Acts. *Language in Society*, pages 1–24, 1976. 98, 105

S. Seneff. TINA: A Natural Language System for Spoken Language Applications. *Computational Linguistics*, 18(1):61–86, 1992. 11, 19, 76

S. Seneff and J. Polifroni. Dialogue Management in the Mercury Flight Reservation System. *Satellite Dialogue Workshop, ANLP-NAACL*, pages 11–16, 2000. 3, 11

S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V Zue. GALAXY-II: a reference architecture for conversational system development. *Proceedings of the* 5^{th} *International Conference on Spoken Language Processing*, pages 931–934, 1998. 10

C. Sharma and J. Kunins. *VoiceXML: Strategies and Techniques for Effective Voice Application Development with VoiceXML 2.0*. John Wiley and Sons, New York, 2002. 30

R. Sharp, E. Bocchieri, C. Castillo, S. Parthasarathy, C. Rath, M. Riley, and J. Rowland. The Watson Speech Recognition Engine. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 4065–4068, 1997. 10

B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 3^{rd} edition, 1997. 106

S. Slomka and S. Sridharan. Automatic Gender Identification Optimised for Language Independence. *Proceedings of IEEE Region 10 Annual Conference. Speech and Image Technologies for Computing and Telecommunications*, pages 145–148, 1997. 16

R. Stalnaker. Assertion. *Pragmatics: Syntax and Semantics*, 9:315–332, 1978. 99

S. S. Stevens. On the Psychophysical Law. *The Psychological Review*, 64(3):153–181, 1957. 59

C. Teixeira, I. Trancoso, and A. Serralheiro. Accent Identification. *Proceedings of the* 4^{th} *International Conference on Spoken Language Processing*, 3:1784–1787, 1996. 17

H. Ting, Y. Yingchun, and W. Zhaohui. Combining MFCC and Pitch to Enhance the Performance of the Gender Recognition. *Proceedings of the* 8^{th} *International Conference on Signal Processing* , pages 16–20, 2006. 16

L. M. Tomokiyo. *Recognizing Non-native Speech: Characterizing and Adapting to Non-native Usage in Speech Recognition*. PhD thesis, Carnige Mellon University, 2001. 66

L. M. Tomokiyo and A. Waibel. Adaptation Methods for Non-native Speech. *Proceedings of the Workshop on Multilinguality in Spoken Language Processing*, 2001. 14, 15

D. Traum and S. Larsson. Information State and Dialogue Management in the TRINDI Dialogue Move Engine Toolkit. *Natural Language Engineering*, 6:323–340, 2000. 103, 104

A. M. Turing. Computing Machinery and Intelligence. *Mind*, 59(236):433– 460, 1950. 13

C. J. van Rijsbergen. *Information Retrieval*. Buttersworth, 1975. 143

V. M. Velichko and N. G. Zagoruyko. Automatic Recognition of 200 Words. *International Journal of Man-Machine Studies*, 2:223, 1970. 13

A. J. Viterbi. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967. 61, 79

W3C. Voice Browser Call Control: CCXML Version 1.0. W3C Recommendation `http://www.w3.org/TR/ccxml/`;last accessed February 27, 2009, 2007. 29

W3C. Voice Extensible Markup Language (VoiceXML) 2.0 Specification. W3C Recommendation `http://www.w3.org/TR/voicexml20/`; last accessed February 27, 2009, 2004. 29, 108

W. Wahlster. Verbmobil - Translation of Face to Face Dialogs. *Proceedings of the Machine Translation Summit IV*, pages 127–135, 1993. 10

M. Walker and S. Whittaker. Mixed-initiative in Dialogue: an Investigation into Discourse Segmentation. *Proceedings of the* 28^{th} *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 70–78, 1990. 100

M. Walker, C. Kamm, and D. Litman. Towards Developing General Models of Usability with PARADISE. *Natural Language Engineering*, 6, 2000. 112

M. Walker, J. Aberdeen, and G. Sanders. 2001 Communicator Evaluation, 2003. 116, 133

M. A. Walker, A. I. Rudnicky, J. Aberdeen, E. O. Bratt, J. S. Garofolo, H. Hastie, A. N. Le, B. Pellom, A. Potamianos, R. Passonneau, R. Prasad, S. Roukos, G. A. Sanders, S. Seneff, and D. Stallard. DARPA Communicator Evaluation: Progress from 2000 to 2001. *Proceedings of the 7^{th} International Conference on Spoken Language Processing*, pages 273–276, 2002. 3, 10

Y. Wang, L. Deng, and A. Acero. Spoken Language Understanding: An Introduction to Statistical Framework. *IEEE Signal Processing Magazine*, 22(5):16–31, 2005. 20, 77

Z. Wang, T. Schultz, and A. Waibel. Comparison of Acoustic Model Adaptation Techniques on Non-native Speech. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 540–543, 2003. 14, 15

W. Ward and S. Issar. Recent Improvements in the CMU Spoken Language Understanding System. *Proceedings of the ARPA Human Language Technology Workshop*, pages 213–216, 1996. 10, 19, 76

W. Ward and B. Pellom. The CU Communicator System. *IEEE Automatic Speech Recognition and Understanding*, pages 341–344, 1999. 11

A. Wendemuth. *Grundlagen der Stochastischen Sprachverarbeitung*. Oldenbourg Wissenschaftsverlag, Munich, 2004. 61, 62

T. Winograd. *Understanding Natural Language*. Academic Press, New York, 1972. 9

E. Wong and S. Sridharan. Comparison of Linear Prediction Cepstrum Coefficients and Mel-Frequency Cepstrum Coefficients for Language Identification. *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 95–98, 2001. 135

W. A. Woods. *Language Processing for Speech Understanding*. Computer Speech Processing, Prentice-Hall International, Englewood Cliffs, NJ, 1983. 19

W. A. Woods, B. L. Nash-Webber, and R. M. Kaplan. The Lunar Sciences Natural Language System: Final Report. *BBN Report 2378*, pages 207–216, 1972. 9

K. Wu and D. G. Childers. Gender Recognition From Speech. Part I: Coarse Analysis. *Journal of the Acoustical Society of America*, 90(4):1828–1840, 1991. 15

M. Xie, D. Lindbergh, and P. Chu. From ITU-T G.722.1 to ITU-T G.722.1 Annex C: A New Low-Complexity 14kHz Bandwidth Audio Coding Standard. *Journal of Multimedia*, 2(2):65–76, 2007. 169

S. Young. Large Vocabulary Continuous Speech Recognition: a Review. *IEEE Signal Processing Magazine*, 13(5):45–57, 1996. xi, 60, 61

S. Young. ATK: An Application Toolkit for HTK Version 1.6.1. `http://mi.eng.cam.ac.uk/research/dialogue/ATK_Manual.pdf`; last accessed February 27, 2009, 2007. 27, 47, 48

S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. A. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. The HTK Book. Revised for HTK Version 3.4. `http://htk.eng.cam.ac.uk/prot-docs/htkbook.pdf`; last accessed February 27, 2009, 2006. 26, 27, 36, 49, 57, 61, 62, 63, 64, 72, 73, 118, 124

Y. Zeng, Z. Wu, T. Falk, and W. Y. Chan. Robust GMM-based Gender Classification Using Pitch and RASTA-PLP Parameters of Speech. *Proceedings of* 5^{th} *IEEE International Conference on Machine Learning and Cybernetics*, pages 3376–3379, 2006. 16

V. Zue, J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni, and S. Seneff. PEGASUS: a Spoken Dialogue Interface for Online Air Travel Planning. *Speech Communication*, 15:331–340, 1994. 3, 11

V. Zue, S. Sene, J. Glass, J. Polifroni, C. Pao, T. J. Hazen, and L. Hetherington. JUPITER: a Telephone-based Conversational Interface for Weather Information. *IEEE Trans. on Speech and Audio Processing*, 8:85–96, 2000. 11