Domain Action Classification and Argument Parsing for Interlingua-Based Spoken Language Translation

Chad T. Langley

October 2003

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
http://www.lti.cs.cmu.edu

Submitted in partial fulfillment of the requirements
for the degree Doctor of Philosophy

Thesis Committee:
Alon Lavie, Chair
Lori Levin
Alex Waibel
Jaime Carbonell
Kevin Knight, USC/ISI

Abstract

Interlingua-based machine translation systems divide the machine translation process into two phases: analysis and generation. Source language input is converted into an interlingua representation during the analysis phase, and target language output is produced from the interlingua representation during the generation phase. For spoken language translation, the analysis phase typically includes an automatic speech recognizer that transforms a source language acoustic signal into text and an analyzer that produces interlingua representations for the source language text.

We describe an analyzer that combines phrase-level parsing using handwritten grammars and machine learning techniques to transform spoken task-oriented utterances into a shallow semantic interlingua representation called Interchange Format. The representation consists of four components: a speaker tag that indicates the role of the speaker, a domain-independent speech act that represents speaker intention, a sequence of domain-dependent concepts that captures the semantic focus of an utterance, and a set of arguments that encode specific details from an utterance. Since the speech act and concept sequence, collectively referred to as the domain action, are essentially flat category labels, machine learning techniques are especially appropriate for the task of domain action classification.

Our analyzer operates in three stages, first using a robust parser and handwritten phrase-level semantic grammars to extract arguments from utterances. The argument grammars are easier to develop and less domain-dependent than domain action grammars. Since utterances are represented in the Interchange Format as sequences of meaningful segments, our analyzer next identifies segment boundaries using an automatic boundary detector. Finally, domain actions are assigned to each semantic segment using automatic classifiers. In order to ensure that the representations produced are valid, we define a strategy for identifying the best legal representation using the domain action classification output and the Interchange Format specification.

We demonstrated the feasibility of our approach by incorporating our analyzer into the NESPOLE! system, where our analyzer provides analysis for the English and German translation engines. We conducted extensive evaluations of segmentation and domain action classification performance as well as end-to-end translation quality using English and German data from the NESPOLE! Travel & Tourism and Medical Assistance domains. We also examined the effects of different machine learning techniques, input feature sets, classification task definitions, and training data on classification performance. We found that domain action classification could be performed with high accuracy and that word-level and argument parse information could be combined to provide superior performance over either type of information alone. We also examined the robustness and portability of our analyzer compared to an analyzer that used only handwritten domain action grammars. Our approach improved robustness to unseen and automatically recognized inputs. Applying machine learning techniques to domain action identification also improved portability by reducing development time compared to the purely grammar-based approach while providing comparable translation quality.

Acknowledgements

There are a great many people who have my sincere gratitude for the help that they gave me in a variety of ways during the course of my work on this thesis.

I would first like to thank my advisor, Alon Lavie, for all of his advice, guidance, and support during the course of this work. I would also like to thank the rest of my thesis committee: Lori Levin, Alex Waibel, Jaime Carbonell, and Kevin Knight for their helpful comments and suggestions on the preliminary draft of this dissertation and during my defense.

Many thanks also go to all of the colleagues with whom I worked in the NESPOLE! and C-STAR projects. Thanks are especially due to Dorcas Alexander, Donna Gates, and Kay Peterson who, among other things, annotated data and wrote the analysis and generation grammars used in the translation systems described in this dissertation. I am also grateful for their efforts in the end-to-end performance evaluations. Their insights and advice regarding the Interchange Format interlingua and the requirements for analysis in translation systems based on Interchange Format were invaluable in the completion of this work. I would like to thank Michael Bett, Zachary Sloane, and Robert Isenberg for grading the translations produced for the portability experiments. I would also like to thank Roldano Cattoni for many useful chats regarding the NESPOLE! system in general and analysis in particular.

I would like to thank all of the students, faculty, and staff at the Language Technologies Institute for providing a fun and productive environment in which to study and conduct research on language technologies.

Although there are too many to list individually, I would like to thank all of my friends and family for providing support and encouragement in many ways, both large and small. Whether it was from a distance or in day-to-day interactions, each of them contributed in some way to the success of this work.

Finally, I would like to thank my wife, Suzette, for all of her patience and love throughout the course of this work. If it weren't for her support and sacrifices, I would not have made it to the end of this long road.

Contents

List of Tables

List of Figures

Chapter 1 Introduction

1.1 Introduction

Opportunities for multilingual interaction and information exchange are greater now than ever before. Activities such as personal travel, business partnerships, military operations, humanitarian aid efforts, and even simply browsing the web give people from all walks of life who speak a wide variety of languages the opportunity to interact. In many of these situations, spoken language is the most natural medium for people to communicate. In any case, whether the interaction is spoken or written, the ability to communicate in real time, or at least nearly real time, is important. This makes the prospect of developing systems that can automatically perform fast and reliable speech-to-speech machine translation extremely important and useful.

For systems that support a large number of languages, interlingua-based machine translation approaches are particularly attractive. An interlingua abstracts away from the details of any specific language and allows for easy translation between any pair of languages supported by the system. For each source language, an analyzer that converts the source language into the interlingua is required. For each target language, a generator that converts interlingua representations into the target language is required. Given an analyzer and a generator for each supported language, an interlingua-based system simply connects the source language analyzer with the target language generator to perform translation between any language pair.

It is easy to see that the analyzer is a critical component in interlingua-based machine translation systems. The analyzer must accurately transform source language input into the interlingua representation. In the context of a speech-to-speech translation system, this means that the analyzer must be robust to speech recognition errors, spontaneous speech effects, and ungrammatical inputs such as those described in [Lavie, 1996b]. Furthermore, the analyzer must be able to produce analyses in real time (or at least near real time) so as not to slow down the dialogue.

In addition to accuracy, speed, and robustness, the portability of the analyzer is also an important consideration. Despite continuing improvements in speech recognition and translation technologies, restricted domains of coverage are still generally necessary in order to achieve

reasonably accurate machine translation. Porting translation systems to new domains or even expanding the coverage in an existing domain can be extremely difficult and time-consuming. Nevertheless, there is always a demand for translation in new domains. In some cases, there may be a good deal of time available to develop a system for a new domain, but translation may also be needed for a new domain with very little notice. Thus, one of the primary motivations for developing the hybrid analysis approach described in this dissertation was to improve the portability of the analyzer. This goal is accomplished in our analysis approach by replacing the most labor intensive aspect of grammar development and maintenance, writing rules for parsing full domain actions, with automatic domain action classifiers.

Analysis approaches that use handwritten grammars for parsing may provide highly accurate analyses because knowledge about a particular domain may be incorporated into the grammar rules. However, a very large amount of effort by expert human grammar writers is typically required to develop an effective grammar, even for limited domains. Furthermore, no matter how comprehensive a grammar is designed to be, it is generally infeasible for the grammar to completely cover a domain, even for clean input. For spoken language, where inputs are noisy and people tend to follow grammatical conventions less strictly, the problem is further exacerbated. The degradation of a parser using handwritten grammars on inputs that deviate from the grammars can be very sharp, although robust parsing techniques can help to smooth the degradation. On the other hand, machine learning approaches tend to degrade gracefully in the face of noisy input. Machine learning techniques are also typically able to generalize beyond the input on which they have been trained, unlike grammar-based parsers, which must adhere strictly to the rules contained in the grammar. However, since machine learning approaches may not be able to incorporate as much domain knowledge as grammar-based approaches, they may be less accurate on clearly in-domain inputs that would be covered by a grammar. Furthermore, machine learning approaches may require a large amount of training data in order to achieve acceptable performance.

In this dissertation, we develop an analyzer that uses a combination of grammar-based parsing methods and machine learning techniques to take advantage of the benefits of each. The main purpose of the analyzer is to transform spoken task-oriented utterances into a shallow semantic interlingua representation for speech-to-speech machine translation. The hybrid analyzer performs argument parsing by using a robust parser and handwritten phrase-level

semantic grammars to extract low-level interlingua arguments from input utterances. The phrase-level grammars used by the analyzer are easier to develop and less domain-dependent than semantic grammars for full domain actions. The analyzer then uses automatic classification techniques to divide an utterance into semantic segments and to determine the high-level domain action that represents a speaker's intention for each semantic segment. This domain action classification task can be performed with high accuracy and improves the robustness of the analyzer to unseen in-domain inputs and inputs from an automatic speech recognizer. Furthermore, applying machine learning techniques to the task of domain action classification improves portability by eliminating the most labor-intensive stage of grammar writing and reducing data annotation requirements.

1.2 Thesis Statement

An analysis method that includes argument parsing with phrase-level semantic grammars, automatic identification of semantic segments, and automatic domain action classification can provide fast, robust, and accurate analysis for task-oriented interlingua-based spoken language translation. This approach can provide translation performance comparable to manually developed domain action grammars while improving the portability of the analyzer by reducing the effort required for grammar development and data annotation.

1.3 Thesis Contributions

The main contributions of this thesis are the following:

- Development and evaluation of a novel approach to analysis of spoken language for real-time interlingua-based machine translation using phrase-level grammars and domain action classifiers
- Development and evaluation of a reliable method for identifying semantic dialogue unit boundaries in speaker turns
- Empirical comparison of alternative classification approaches, feature sets, and task definitions for domain action classification
- Empirical comparison of the analysis approach with handwritten domain-action-level grammars demonstrating the improved portability and robustness of our approach over the strictly grammar-based approach

- Demonstration of the feasibility of the analysis approach through the full integration of our approach in the English and German translation servers for the NESPOLE! speech-to-speech machine translation system

1.4 Organization of the Dissertation

The remainder of this dissertation is organized as follows. The following section describes the NESPOLE! speech-to-speech translation project, which provided the context within which our analysis approach was developed, and the last section of Chapter 1 describes previous related work. Chapter 2 describes the architecture and components of our hybrid analysis approach and discusses our motivation for using grammars and machine learning as we did. Chapter 3 presents the evaluation of the automatic semantic dialogue unit boundary classifier. Chapter 4 presents a comprehensive evaluation of automatic domain action classification, including a comparison of several machine learning approaches, an assessment of different definitions of the domain action classification task, an examination of different input feature sets, and an evaluation of the strategy used to ensure that our analyzer produces legal interlingua representations. Chapter 5 describes the evaluation of the portability of our analysis approach, including an assessment of the training data requirements and a comparison with a purely handwritten grammar approach. Finally, Chapter 6 summarizes the results presented in this dissertation and discusses directions for future work.

1.5 Research Context

The hybrid analysis approach described in this dissertation was developed primarily within the context of the NESPOLE! translation system ([Metze *et al.*, 2002], [Guerzoni *et al.*, 2003], [NESPOLE!]). The NESPOLE! system is a human-to-human speech-to-speech interlingua-based machine translation system. The NESPOLE! system performs online near-real-time translation in restricted domains using a shallow semantic task-oriented interlingua.

1.5.1 Predecessors of the NESPOLE! Machine Translation System

The NESPOLE! translation system is among the latest in a long line of multilingual domain-limited interlingua-based speech-to-speech translation systems that have been developed at the Interactive Systems Labs at Carnegie Mellon University and Universität Karlsruhe based on the

JANUS translation framework. Many of the ideas and components in the NESPOLE! system evolved from the systems developed and experience gained during the course of these previous research projects.

The earliest JANUS systems ([Waibel *et al.*, 1992], [Osterholtz *et al.*, 1992], [Woszczyna *et al.*, 1993]) were designed for translation of read speech in a very limited conference registration domain. During the course of research efforts such as the first C-STAR consortium and the Enthusiast project ([Qu *et al.*, 1996a], [Qu *et al.*, 1996b]), the JANUS machine translation framework was expanded to support translation of spontaneous speech in the appointment scheduling domain ([Woszczyna *et al.*, 1994], [Suhm *et al.*, 1994], [Suhm *et al.*, 1995], [Lavie *et al.*, 1996], [Lavie *et al.*, 1997c], [Waibel *et al.*, 1997]). More recently the domain of coverage of the JANUS the translation system was expanded from appointment scheduling to travel planning ([Lavie *et al.*, 1997a]). Two recent predecessors of the NESPOLE! translation system that were based on the JANUS-III translation system ([Levin *et al.*, 2000a]) and supported translation in the travel planning domain were the C-STAR II system from the second phase of the C-STAR consortium ([C-STAR]) and LingWear ([Fügen *et al.*, 2001]).

In each system, translation was performed through a task-oriented interlingua. Each system used the same underlying translation approach adapted for the specific domain and environment of the system. There were four main steps in the translation process. First, spoken input was passed through an automatic speech recognizer. Output from speech recognition was then passed through an analysis engine that produced an interlingua representation for the input utterance. Next, target language text was generated from the interlingua representation. Finally, the target language text was synthesized into speech.

The LingWear system ([Fügen *et al.*, 2001]) was designed to provide foreign language support on a wearable platform. The system provided automatic speech-to-speech translation, navigational assistance, and local tourism information. Users interacted with the LingWear system via spoken input or via a touch-sensitive screen. In translation mode, spoken source language input was passed through a JANUS translation module to produce synthesized speech in the target language. By sharing the microphone and selecting the appropriate input language, a foreign traveler and a local inhabitant could carry out short conversations. LingWear allowed input in English and German and produced translations into English, German, and Japanese.

The C-STAR II translation system was the most recent predecessor to the NESPOLE! translation system and had a large influence on the design of the NESPOLE! system. The C-STAR II system was primarily developed by an international consortium of research labs including the four partners from the NESPOLE! project as well as ATR in Japan and ETRI in Korea. The system supported translation in the travel planning domain between English, German, Italian, French, Japanese, and Korean. The travel planning domain primarily focused on dialogues about making reservations and payments for hotels, transportation, and events. The main difference between the architecture of the C-STAR II system and the architecture of the NESPOLE! system described in the following section is the way in which users interact with the translation servers. In the C-STAR II system, the user was required to run the complete translation system for their language locally on their own machine. In the NESPOLE! system, a dedicated remote server performs the processing required for translation.

1.5.2 The NESPOLE! Machine Translation System

The NESPOLE! translation system ([Lavie *et al*., 2002], [Metze *et al*., 2002], [Guerzoni *et al*., 2003]) is designed to provide human-to-human speech-to-speech machine translation using an interlingua-based approach similar to that used in the JANUS-III system. The general goal of the system is to provide translation over the Internet to facilitate communication for e-commerce and e-service applications between common users in real-world settings. The NESPOLE! system supports translation in two domains between clients who speak English, German, or French and agents who speak Italian ([Taddei *et al*., 2003]).

The first domain addressed in the NESPOLE! system is the Travel & Tourism domain. This domain is an extension of the C-STAR II travel planning domain, which focused mainly on making and paying for reservations for accommodations and transportation. In the Travel & Tourism domain, a client who is traveling to a foreign country may connect with an agent at a local tourism bureau who does not speak the same language in order to get information about the region to which they are traveling. In this domain, we imagine that a traveler is browsing the website of the tourism agency and decides to contact the agency for more detailed information. Specifically, in the NESPOLE! system a client may obtain information about a winter or summer vacation in the Trentino region of northern Italy. For example, users could discuss information regarding vacation packages, accommodations (e.g. hotels, campsites, etc.), tourist attractions

(e.g. parks, lakes, castles, museums, etc.), or recreational activities (e.g. skiing, swimming, etc.). Users could also inquire about the locations of particular hotels or attractions and about directions for traveling between two sites. Figure 1 shows an example of what a typical dialogue in the Travel & Tourism domain might look like.

A: APT Trentino. Good Morning.
C: Good morning. I would like to take a vacation in Val di Fiemme.
A: I'll show you a map of Val di Fiemme.
C: Thank you.
A: Can you see it?
C: Yes I can.
A: Do you have a specific place in mind?
C: Yes, I would like to stay in Cavalese.
A: There are many hotels in Cavalese. I'm sending you a map of Cavalese.
C: Yes, thank you. I need a two star hotel.
A: I suggest the Hotel Lagorai. I will indicate the hotel with a blue circle.
C: Is the hotel close to a bus stop?
A: Yes. I will show the bus stop with a red square.
C: I want to go downhill skiing. Can I get to the ski area by bus from Cavalese?
…

Figure 1: Dialogue example from the NESPOLE! Travel & Tourism domain

The second domain supported by the NESPOLE! system is the Medical Assistance domain. The Medical Assistance domain was added as an experiment to examine the portability of the NESPOLE! translation approach. In this domain, a traveler in a foreign country (i.e. the client) who is not feeling well can connect with a local doctor who does not speak the same language (i.e. the agent) to get a preliminary diagnosis and medical advice. We imagine that the traveler would access the NESPOLE! translation system through a terminal at their hotel rather than calling the doctor directly. In the Medical Assistance domain, a client may describe health-related personal information (e.g. age, medications, allergies, medical risk factors, etc.) as well as information related to the symptoms they are experiencing (e.g. specific symptoms, when the symptoms started, duration, severity, etc.). The doctor may perform a medical interview to elicit relevant information from the patient. The doctor may then describe their diagnosis and make appropriate recommendations (e.g. over-the-counter medication, an office visit, immediate

medical attention, etc.). The dialogues supported in the NESPOLE! Medical Assistance domain focus primarily on flu-like symptoms and/or chest pain. Figure 2 shows an example of part of a typical dialogue in the Medical Assistance domain.

A: Good morning. May I help you?
C: Good morning. I'm not feeling very well.
A: What kind of problems are you having?
C: I have a pain in my chest.
A: How long have you been having this pain?
C: Since this morning.
A: What symptoms are you having?
C: I have a strong pain in my chest and I also have rapid heartbeats.
A: Wait a moment. I will send you an image. Can you see the image?
C: Yes, I see the image.
A: Where do you have the pain exactly? Can you indicate it on the image?
C: I have the pain in the middle of my chest. Here.
[THE PATIENT INDICATES THE LOCATION ON THE IMAGE]
A: Do you have high blood pressure?
C: No. I don't have high blood pressure, but I have diabetes. I take insulin.
…

Figure 2: Dialogue example from the NESPOLE! Medical Assistance domain

The NESPOLE! project was a collaboration among four research labs and two industrial partners in four countries. The English and German translation resources were developed in the Interactive Systems Laboratories at the Language Technologies Institute at Carnegie Mellon University in Pittsburgh and Universität Karlsruhe in Germany. The Italian translation resources were developed at Istituto Trentino di Cultura, Istituto per la Ricerca Scientifica e Tecnologica (ITC-irst) in Trento, Italy. The French translation resources were developed at Université Joseph Fourier in Grenoble, France. The video conferencing resources that connected the users and the translation systems were developed by AETHRA, a telecommunications company in Ancona, Italy. Finally, Azienda per la Promozione Turistica di Trento (APT), the Trentino tourism board, provided support for developing the requirements for the Travel & Tourism domain, data collection, and system testing.

One of the guiding principles in the design of the NESPOLE! architecture ([Lavie *et al.*, 2001]) was to create a system that imposed minimal hardware and software requirements on the end users. Under the NESPOLE! architecture, a user is only required to have a computer with an Internet connection, a microphone, speakers, and a video camera. The only software required for connecting to the NESPOLE! system is videoconferencing software such as Microsoft® NetMeeting and a small downloadable NESPOLE! Whiteboard component. All of the language processing required for translation is performed on a dedicated server (or servers) rather than on the user's computer. After a connection to the NESPOLE! system initiated, all of the processing required for translation is completely transparent to the user.

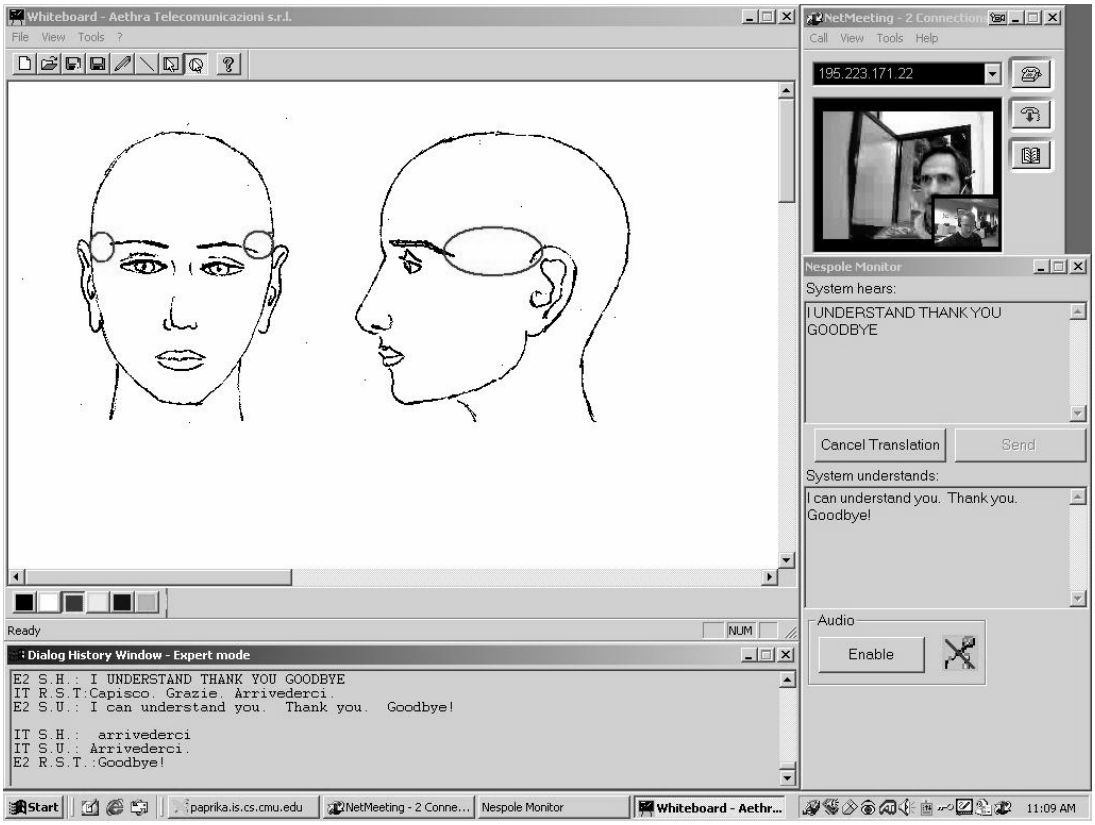

Figure 3: The NESPOLE! user interface

An example of the user interface for the NESPOLE! translation system ([Taddei *et al.*, 2002]) is shown in Figure 3. The interface consists of four main components. Microsoft®

NetMeeting (upper right) provides the videoconferencing connection. The Whiteboard (upper left) allows for a multimodal aspect to the communication between the client and agent by supporting the sharing of images and free-hand drawings or annotations. In the example interface shown in Figure 3, the agent (a doctor) shows the client a picture of a head, and the client indicates the location of their headache by circling it on the image. The interface also includes a Dialogue History Window (lower left) that allows the users to view the inputs, paraphrases, and translations from previous turns. Finally, the interface includes the NESPOLE! Monitor (lower right), which allows the user to interact with the translation system. The NESPOLE! system uses a push-to-talk mechanism for recording user speech, and the user initiates recording by clicking on the *Enable* button in the NESPOLE! Monitor window. The NESPOLE! Monitor window shows the user the result of speech recognition for their most recent turn in the *System hears:* box and the paraphrase produced by generating from the interlingua representation for the utterance back into the source language in the *System understands:* box. The NESPOLE! Monitor window also displays progress meters (not shown in Figure 3) to indicate the progress of the system while translation is being performed. Finally, the *Cancel Translation* button allows a user to signal that an error occurred during their turn and that the most recent translation should be ignored. The user may then either edit the text of the automatic speech recognition output or speak again.

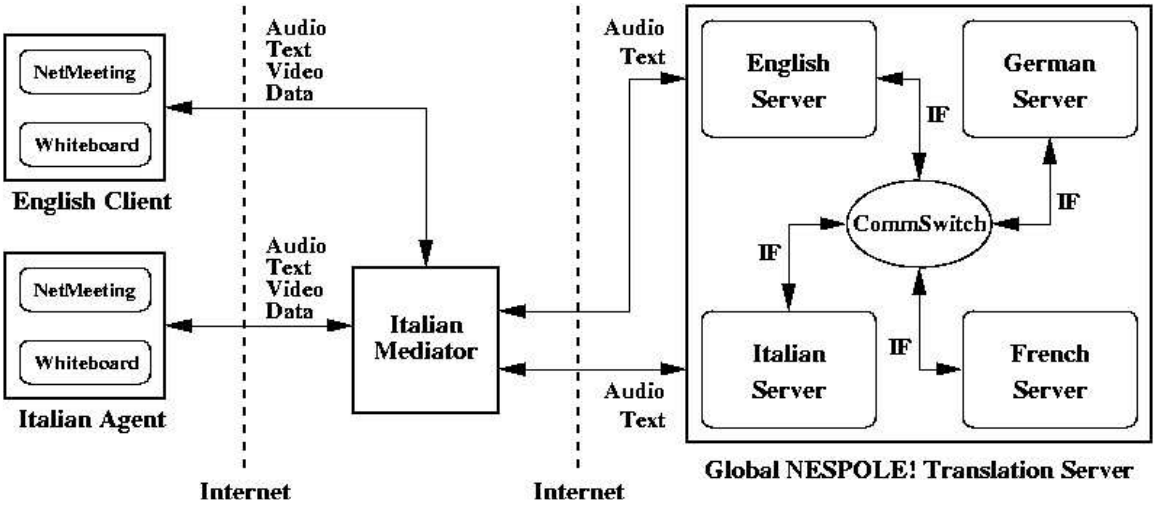

Figure 4: Global architecture of the NESPOLE! machine translation system

The general architecture of the NESPOLE! translation system ([Lavie *et al.*, 2001]) is shown in Figure 4. In a typical scenario using the NESPOLE! system, a user (i.e. client) would initiate a connection over the Internet to the system. This might be accomplished by clicking on a link on the web page or using a terminal at the user's hotel. Upon initiating the connection, the user is first connected with a component called the Mediator ([NESPOLE!-D8, 2001], [Lavie *et al.*, 2002]). The Mediator is responsible for managing the videoconferencing connection between the client and the agent and for passing speech and text between the users and the global translation server. After receiving a connection from a client, the Mediator contacts the agent and establishes connections to the appropriate language-specific translation servers. The Mediator is also responsible for transmitting spoken (or typed) input from the users to the translation server and synthesized speech (along with textual paraphrases and translations) from the translation server to the users.

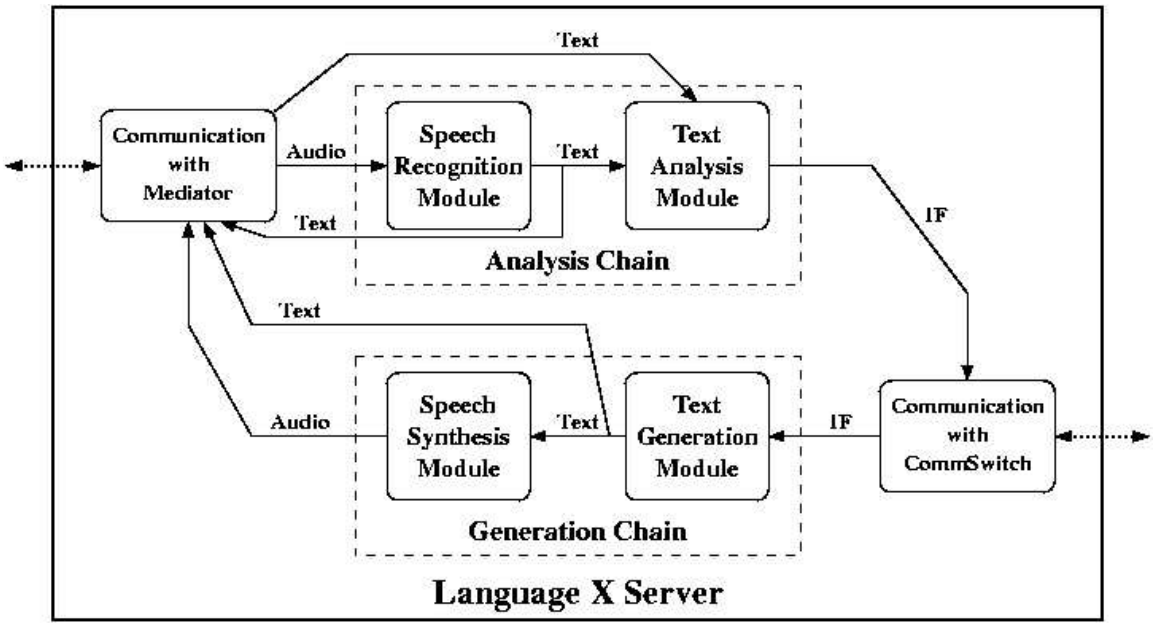

Figure 5: Internal architecture of the language-specific servers in the NESPOLE! translation system

Although the Global NESPOLE! Translation Server may be thought of conceptually as a single dedicated server, it is actually composed of a language-specific server for each language

supported by the system as well as a simple Communication Switch (*CommSwitch*) for passing messages between the language-specific servers. Figure 5 shows the general architecture of the language-specific servers ([Lavie *et al*., 2001]). Each server contains an analysis chain comprised of a *Speech Recognition Module*, which transforms spoken input into text, and a *Text Analysis Module*, which transforms text into the interlingua representation. Each server also contains a generation chain comprised of a *Text Generation Module*, which transforms interlingua representations into text, and a *Speech Synthesis Module*, which transforms text into speech.

The specific implementation of the components within each language-specific server varies since each server was developed by a different partner in the NESPOLE! project. Because there was a close partnership between the research groups at Carnegie Mellon University and Universität Karlsruhe, the English and German servers are very similar. The *Speech Recognition Module* in both servers is implemented using the JANUS Recognition Toolkit ([Soltau *et al*., 2001a], [Soltau *et al*., 2001b]). The hybrid analysis approach described in this dissertation serves as the *Text Analysis Module* for both servers. The *Text Generation Module* in each server is implemented using a unification-based generator ([Han and Lavie, 2003]) based on the GenKit system ([Tomita and Nyberg, 1988]). The *Speech Synthesis Module* is implemented using the Festival system ([Black *et al*., 2001]), and the German *Speech Synthesis Module* uses voices created at Oregon Graduate Institute ([Macon *et al*., 1998], [Macon *et al*., 1997]).

Translation of an English input utterance into Italian output in the NESPOLE! system takes place as follows. First, the English client speaks an utterance. The Mediator relays the spoken input to the English translation server where automatic speech recognition takes place. The best hypothesis from the speech recognizer is then passed to the analyzer for conversion to the interlingua representation and back to the English client via the Mediator. The interlingua representation is passed from the English translation server to the Italian translation server via the Communication Switch. The interlingua representation is also passed to the English generation module so that a paraphrase can be produced and sent back to the English client via the Mediator. When the Italian translation server receives the interlingua representation, the generator is used to produce Italian text for the utterance, and the speech synthesis module produces speech for the text. Both the text and the synthesized speech are passed from the Italian translation server to the Mediator and from there to the Italian agent.

1.5.3 Interlingua-Based Machine Translation

There have been many approaches to machine translation over the years ([Hutchins and Somers, 1992], [Dorr *et al.*, 1999]). One dimension along which machine translation approaches vary is the extent to which intermediate representations are used to perform translation. At one end of the spectrum are direct translation approaches. These approaches translate the words from the source language directly into words in the target language without using any intermediate representation. Such approaches generally focus on properties specific to the language pair to perform translation. In the middle of the spectrum are a variety of transfer-based approaches that convert natural language into some form of structured intermediate representation that is specific to the particular source language. The structured form often includes information at the syntactic or semantic level. Transfer-based approaches transform source language structures into corresponding target language structures, and the translation into words in the target language is then generated from the target language structures. Finally, at the opposite end of the spectrum from direct approaches are approaches that use a language-independent intermediate representation called an interlingua to perform translation. As in transfer-based approaches, the first step in translation is to transform the source language input into a structured intermediate representation. However, since the interlingua abstracts away from language-specific details, there is no need to perform any transformations on the structure before generating the target language. Thus, in interlingua-based approaches, the source language is converted directly into the interlingua, and the target language is generated directly from the interlingua.

Interlingua-based approaches to translation offer several advantages that are particularly useful for systems that support a large number of languages ([Levin *et al.*, 2003a]). Figure 6 illustrates how interlingua-based translation is used in a multilingual system. Because an interlingua representation abstracts away from the details of any specific language, translation between any pair of languages supported by the system is very straightforward. For each source language supported by the translation system, an analyzer that converts source language inputs into the interlingua representation is required. Likewise, for each target language, a generator that converts interlingua representations into the target language is required. The languages in the top half of Figure 6 represent source language analyzers, and the languages in the bottom half of Figure 6 represent target language generators. Given an analyzer and a generator for each supported language, translation between any pair of supported languages is relatively simple.

The interlingua-based translation system simply uses a source language analyzer to produce interlingua representations from source language input and then uses a target language generator to produce target language output from the interlingua representations.

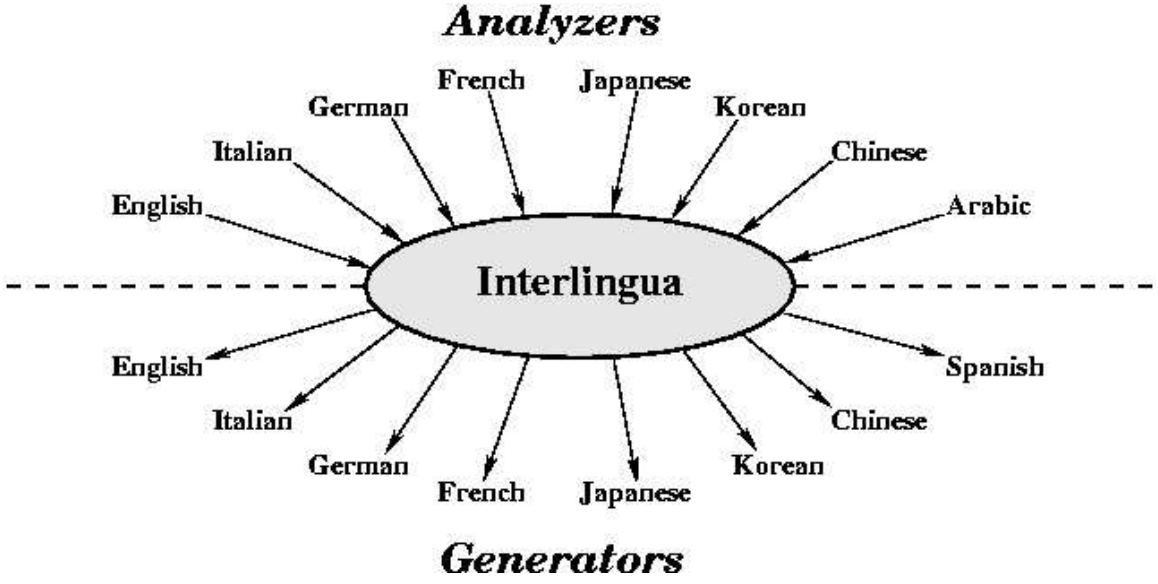

Figure 6: Interlingua-Based Machine Translation

Interlingua-based approaches to machine translation also make adding new source and target languages to a translation system relatively easy. In order to add a source language, only an analyzer for the new language must be provided, regardless of the number of languages already supported by the system. Translation from the source language into any existing target language is then accomplished by connecting the new analyzer to any of the existing generators. Similarly, a new target language can be added by providing a generator for the language. Then the analyzer for any existing source language is connected with the new generator to perform translation into the target language. For example, suppose that the six leftmost languages in Figure 6 (English, Italian, German, French, Japanese, and Korean) were already present in a translation system. Adding all-ways translation to and from Chinese simply requires the addition of a Chinese analyzer and generator with no additional modifications to the system. Alternatively, suppose that Arabic input were of interest, but there was no need for Arabic

output. In that case, only an Arabic analyzer would have to be added to the system. Similarly, if Spanish output were needed, a Spanish generator could be added.

In contrast, adding a new language to a translation system that uses a direct or transfer-based approach can be considerably more difficult because such approaches require resources specific to each language pair and possibly specific to the direction of translation as well. For example, consider adding Chinese to a transfer-based system that already supports English, Italian, German, French, Japanese, and Korean. As in the interlingua-based system, a Chinese analyzer and generator must be provided. In addition, structure mapping rules or procedures must be provided for transforming Chinese source structures into structures for each target language. If the structure mapping resources are not symmetric, then they must also be provided from each existing source language into Chinese. For each new language added to a translation system, this process becomes more difficult. Whereas the addition of a new language (source and target) to an interlingua-based system requires only the addition of a single analyzer and a single generator, the addition of a new language to a direct or transfer-based system with N languages may require the addition of up to $2N$ new components. Furthermore, all-ways translation for N languages in an interlingua-based system requires $2N$ components: an analyzer and generator for each language. A direct or transfer-based system for the same N languages would require on the order of N^2 components.

Interlingua-based translation approaches also offer advantages from the perspective of system development. Because the source language is converted directly into the interlingua and the target language is produced directly from the interlingua, there are no language-pair dependent connections in an interlingua-based system. This means that those developing the analyzer and generator for each language only need to know one language (plus the interlingua). Furthermore, interlingua-based systems allow for each analyzer and generator for to be developed independently of other analyzers and generators. Different analysis and generation methods can be used for each language, and the components can be developed by native speakers of each language who may or may not be fluent in any of the other languages supported by the system. This is especially important for consortiums such as NESPOLE! and C-STAR in which different research groups develop the analysis and generation modules for each language.

One additional advantage that interlingua-based translation systems offer for human-to-human translation is the ability to produce a paraphrase of a user's input by "translating" back

into the source language through the interlingua. Assuming that the generators for each target language perform at roughly the same level, this allows the user to receive some verification of the quality of the translation. Although not a perfect indicator of translation quality, such paraphrases can give users a reasonable idea as to whether or not the translation of their utterance will be understandable. If the paraphrase is not acceptable, the translation system may provide some mechanism for the user to signal the unacceptable translation and try their turn again.

1.5.4 The Interchange Format Interlingua

As mentioned previously, spoken language translation in the NESPOLE! system is performed via a task-oriented interlingua representation. The interlingua representation used in the NESPOLE! system is called Interchange Format ([Levin *et al.*, 2003a], [Levin *et al.*, 2003b], [Levin *et al.*, 2002]). An earlier version of the Interchange Format interlingua was also used in the C-STAR II and LingWear translation systems ([Levin *et al.*, 1998], [Levin *et al.*, 2000b]). The Interchange Format is a shallow semantic representation for utterances in task-oriented dialogues in limited domains. The aim of the Interchange Format representation is to capture speaker intention rather than the literal meaning or predicate-argument structure of an utterance. Thus, the representation abstracts away from language-specific syntax and idiosyncrasies while preserving the meaning of the original input. Such a representation can be particularly effective for domains containing many formulaic phrases that are indicative of a speaker's intention but that do not translate directly into other languages. Figure 7 contains a list of English phrases that might indicate a speaker's intention to make a suggestion. While attempts to translate the literal meaning of the phrases into other languages may be met with varying degrees of success, the Interchange Format representation simply encodes that the speaker's intent was to suggest something. It is left for the target language generator to decide how to best express a suggestion.

i suggest
i would like to recommend
have you considered
how about
it is usually a good idea to
you would have to
what you should maybe do is
i would consider taking
i should think you will enjoy

Figure 7: Examples of English phrases for making a suggestion

Although the NESPOLE! translation system only supports input in Italian, English, German, and French, the additional languages of the C-STAR consortium and the NESPOLE! user group (Chinese, Japanese, Korean, Arabic, Spanish, and Catalan) were also taken into consideration in designing the Interchange Format. The Interchange Format was also designed to balance the need for enough expressive power to adequately convey the intent and meaning of utterances with the need for a representation that was simple enough to be used and reproduced reliably by multiple research teams ([Levin *et al.*, 2002]).

The Interchange Format represents speaker intention at the level of semantic dialogue units, which are semantic segments for which speaker intention can be identified. A single utterance may contain multiple semantic dialogue units. For example, the utterance "*Hello. I would like information about Val di Fiemme. I want to go skiing there.*" would contain three semantic dialogue units, one for each sentence in the utterance. As this example shows, semantic dialogue units often correspond to the sentences in an utterance, but there is no restriction that a single sentence must correspond to a single semantic dialogue unit. Each semantic dialogue unit in an utterance is assigned an Interchange Format representation. The Interchange Format representation for a semantic dialogue unit is composed of four key elements: a speaker tag, a speech act, an optional sequence of concepts, and an optional set of arguments. The speech act and concept sequence are collectively referred to as the domain action. The general form of an Interchange Format representation is shown in Figure 8, and Figure 9 contains examples of utterances along with their corresponding Interchange Format representations.

```
speaker : speech act +concept* argument*
```

Figure 8: General form of an Interchange Format representation

Yes.
```
c:affirm
```

Hello.
```
c:greeting (greeting=hello)
```

How may I help you?
```
a:offer+help
```

I would like information about Val di Fiemme.
```
c:give-information+disposition+information-object
  (disposition=(who=i, desire),
   info-object=(information,
                object-topic=name-val_di_fiemme_area))
```

Of course there aren't hotels in the park.
```
a:negate-give-information+existence+accommodation
  (accommodation-spec=(hotel, quantity=plural)
   location=(park, identifiability=yes))
```

Do you have any shortness of breath with that chest pain?
```
a:request-information+experience+health-status
  (experiencer=you,
   health-status=(breathlessness, object-ref=any),
   co-health-status=(chest_pain, identifiability=distant))
```

Are you still there?
```
a:dialog-request-present (who=you)
```

Figure 9: Examples of Interchange Format representations

The first component of an Interchange Format representation is the speaker tag. The speaker tag encodes the role of the speaker of an utterance in a dialogue. Two speaker tags are used in the Interchange Format: 'a' for agent and 'c' for client. The agent would be a representative at a tourism bureau for the NESPOLE! Travel & Tourism domain and a health care provider in the NESPOLE! Medical Assistance domain. The client would be a traveler in both domains. The speaker tag is included because the generation for some otherwise identical Interchange Format representations may be different depending on the role of the speaker in the dialogue.

The second main element of the Interchange Format representation is the speech act. Speech acts are generally domain-independent and are used to represent the main intention of the speaker in uttering a particular semantic dialogue unit. For example, `give-information`

would be used when the speaker is providing information to the listener (e.g. "*I would like information about Val di Fiemme.*" in Figure 9), and `request-information` would be used when the speaker is asking the listener to provide some information (e.g. "*Do you have any shortness of breath with that chest pain?*" in Figure 9). The speech act is a mandatory component of the Interchange Format representation, and in some cases (e.g. "*Yes.*" in Figure 9), a semantic dialogue unit may be assigned a speech act with no concepts or arguments. Most semantic dialogue units are only assigned one speech act, but the Interchange Format defines three speech acts (`verify-`, `negate-`, and `request-verification-`) that may be combined with other speech acts to modify the intention represented by the speech act (e.g. "*Of course there aren't hotels in the park.*" in Figure 9). The Interchange Format also defines a set of "meta" speech acts that are used for communicating about the status of the dialogue or the translation system (e.g. "*Are you still there?*" in Figure 9).

The third major component of the Interchange Format representation is the concept sequence. A concept sequence is created compositionally from an inventory of domain-dependent concepts and may contain zero or more concepts. The degree of domain specificity of individual concepts varies somewhat. Figure 9 contains examples of concepts that are quite domain-specific (e.g. `+accommodation` for the Travel & Tourism domain and `+health-status` for the Medical Assistance domain) as well as concepts that are relatively domain-independent (e.g. `+disposition` and `+existence`). While the speech act represents the general intention expressed in a semantic dialogue unit, the concepts capture the semantic focus of the semantic dialogue unit. Together the speech act and concept sequence form a domain action that is used to represent a speaker's intention to achieve some domain-dependent activity. [Levin *et al.*, 2003b] argues that domain actions, rather than speech acts alone, provide an appropriate level of representation of speaker intention for achieving good quality translation.

The final element of the Interchange Format representation is the set of arguments. The set of arguments for a particular semantic dialogue unit may contain zero (e.g. "*How may I help you?*" in Figure 9) or more (e.g. "*Do you have any shortness of breath with that chest pain?*") arguments. The order of the arguments in an Interchange Format representation does not carry any meaning, so a target language generator is free to generate text for the arguments in whatever manner and order is appropriate for the language. Interchange Format arguments use a simple feature-value representation to encode specific details contained in a semantic dialogue

unit. Arguments are represented as an argument name followed by the "=" symbol followed by a value. Argument values may be atomic (e.g. the value of the `greeting=` argument for "*Hello.*" in Figure 9) or complex with multiple values and/or subarguments (e.g. the value of the `info-object=` argument for "*I would like information about Val di Fiemme.*" in Figure 9).

The Interchange Format specification ([NESPOLE!-IF-Spec], [Levin *et al.*, 2003a]) defines the set of components used in the Interchange Format as well as the ways in which the components are allowed to combine to form valid representations. The Interchange Format specification consists of three parts: the domain action specification, the argument specification, and the value specification. The domain action specification defines the set of legal speech acts and concepts and specifies how the speech acts and concepts may be combined to form valid domain actions. In addition, the domain action specification defines the set of top-level arguments licensed by each speech act and concept. In order for an argument to appear in an Interchange Format representation for a semantic dialogue unit, it must be licensed by at least one of the components of the domain action. Figure 10 shows an example of the definition for the `+disposition` concept from the NESPOLE! domain action specification. The domain action specification contains a similar definition for every speech act and concept sequence in the Interchange Format.

```
DEF: +disposition
   continuations:
      ($bottom$ +accommodation +action +activity +admission +airport
       +arrival +attraction +budget +cancellation +change +checkin
       +checkout +click +completion +concept +connection +contain
       +contained-in +currency +decrease +delay +departure +deposit
       +describe +directions +display +drop-off +effect +equipment
       +event +eventuality +exclusion +exclusion-from +exit
       +experience +explain +facility +feasibility +feature +food
       +general-action +give-medical-procedure +goto +health-status
       +hear +help +improve +inclusion +inclusion-in +increase
       +indicate +inform +information-object +knowledge +learn
       +learn-about +meal +medical-procedure +meeting +numeral
       +object +obtain +occurrence +order +package +payment +person
       +pick-up +plan +price +proceed +purchase +read
       +receive-medical-procedure +recommendation +rent +repeat
       +reservation +restaurant +room +search +seat +send +service
       +skill +stay +tour +transportation +trip +vehicle +view
       +worsen +write)
   arguments: (<disposition=> <manner=>)
```

Figure 10: Definition of the `+disposition` concept from the NESPOLE! Interchange Format specification

The set of concepts that are allowed to immediately follow `+disposition` in a concept sequence are listed in the `continuations:` section of the definition shown in Figure 10. The special symbol `$bottom$` in the continuations list indicates that the `+disposition` concept may legally occur as the last concept in a concept sequence. The set of arguments licensed by `+disposition` is listed in the `arguments:` section of the definition. Consider the Interchange Format representation for the utterance "*I would like information about Val di Fiemme.*" in Figure 9. The definition shown in Figure 10 indicates that the `+information-object` concept is allowed to follow the `+disposition` concept in the concept sequence. Similarly, the definition of the `give-information` speech act includes the `+disposition` concept in its continuation list, and the continuation list for the `+information-object` concept contains the `$bottom$` symbol. As shown in Figure 10, the `disposition=` argument is licensed by the `+disposition` concept. The `+info-object=` argument is licensed by the `+information-object` concept.

The domain action specification will be particularly important for the analysis approach described in this dissertation. As described in Sections 2.5.4 and 4.6, it will allow us to ensure that that the Interchange Format representations produced by our analysis approach are valid and will support a mechanism for searching for alternative representations when the best output of the analyzer is invalid. In addition to the domain action specification, the Interchange Format specification includes an argument specification and a value specification. The argument specification defines the set of legal arguments, including all top-level arguments and subarguments. In a manner similar to the domain action specification, the argument specification defines the set of values that an argument can take as well as the set of subarguments allowed for the argument. The value specification defines sets of values to which definitions in the argument specification or the value specification may refer. The main purpose of the value specification is for convenience so that sets of values that are referred to by multiple definitions only need to be explicitly listed once. We do not use the argument and value specifications in our analysis approach under the assumption that the handwritten argument grammars used in our analyzer will produce legal argument representations.

1.5.5 NESPOLE! Data Collection and Corpora

Development of the Interchange Format specification and the translation modules used in the NESPOLE! translation system requires the availability of a corpus of representative dialogues from the domain to be covered. The NESPOLE! databases for the Travel & Tourism domain and the Medical Assistance domain served as the primary source for the training and test data used in the experiments described in this dissertation. In addition to the NESPOLE! data, the database from the C-STAR II travel planning domain and the Babylon medical domain were used to supplement the NESPOLE! data when possible. Although the dialogues in those databases do not exactly match the domains of the NESPOLE! system, they are generally similar enough to provide coverage of some aspects of the NESPOLE! domains.

The data collection protocols used for building the NESPOLE! databases are described in [Burger *et al*., 2001] and [Burger *et al*., 2003]. Monolingual and bilingual conditions were used for recording the dialogues in the NESPOLE! Travel & Tourism domain database. The monolingual condition consisted of dialogues between a client and an agent at a tourism agency both speaking the same language. The clients were students or professionals who were native speakers of the language in which the dialogue was spoken and the agents were real representatives from APT, the Trentino tourism board, who were native Italian speakers and bilingual in the language of the dialogue. For each dialogue, the client was given a description of their role in a domain scenario and some goals to accomplish. They then had a conversation with an agent to accomplish whatever goals were set for them in whatever way they found most natural. The client and agent communicated with each other directly in the client's native language using the Microsoft® NetMeeting videoconferencing software. For the English dialogues, for example, a native-English-speaking client in the United States (at Carnegie Mellon) connected with a bilingual (Italian-English) agent at APT in Italy and held a conversation according to the scenario that they were given. Monolingual data was collected in this manner for English, German, French, and Italian and forms the bulk of the data in the NESPOLE! database. The setup for the bilingual data collection condition was similar. However, rather than communicating directly using Microsoft® NetMeeting, the client and agent each communicated in their native language using the NESPOLE! system to translate their dialogue. Bilingual data was collected for English and German. The C-STAR II travel planning data that

was used to supplement the NESPOLE! Travel & Tourism database was originally collected in a monolingual role-playing condition with both speakers at the same site.

The NESPOLE! Medical Assistance database was collected using only a monolingual protocol. The protocol was similar to the one used for the Travel & Tourism domain except that both the client and agent were native speakers of the language. All of the speakers that participated in the Medical Assistance domain data collection were doctors, nurses, or medical students. Data was collected for English, German, and Italian in the Medical Assistance domain, and some of the data was manually translated into other NESPOLE! languages. The English NESPOLE! Medical Assistance database was also supplemented with data from the Babylon medical domain. The Babylon data consisted mainly of individual utterances in which patients described a variety of symptoms and health concerns.

All of the NESPOLE! dialogues were recorded and manually transcribed, at which point they were suitable for training the speech recognizers used in the NESPOLE! servers. However, in order to be useful for developing the analyzers and generators, the dialogues had to be tagged with Interchange Format representations. Each speaker turn in the transcribed dialogues was first divided into semantic dialogue units. Then each semantic dialogue unit was annotated with its corresponding Interchange Format representation. The annotation of the NESPOLE! database was an effort that continued over the course of the project. All of the dialogues in the database were segmented into semantic dialogue units. However, the databases contain some dialogues that were fully annotated with Interchange Format representations, some dialogues for which only one side of the dialogue was tagged, and some dialogues for which only fragments of the dialogue were tagged. This was especially true early in the annotation process since tagging was prioritized based on usefulness (e.g. client side utterances for English, semantic dialogue units containing novel expressions of concepts, utterances of native speakers, etc.). Additionally, some of the dialogues in the NESPOLE! database were manually translated into other NESPOLE! languages. Although the translations were clearly not native utterances in a dialogue setting, they could be translated by non-Interchange Format experts and increased the amount of tagged data available for the new language without requiring additional annotation effort. In addition to the recorded dialogues, the NESPOLE! database also contains some individual utterances that were manufactured during the course of developing the Interchange Format in order to provide initial data for defining the specification and/or to provide examples of certain aspects of the

specification. Including data from all of the NESPOLE! languages and domains, the NESPOLE! database contains over 12,000 semantic dialogue units in dialogues that have been completely annotated with Interchange Format representations. More than 2000 semantic dialogue units in completely annotated dialogues from the C-STAR II database have also been tagged with the NESPOLE! Interchange Format. The database also contains additional annotated semantic dialogue units in partially annotated dialogues and manufactured data.

```
e709wb.7.0  comments: DATA from e709_1_0006_ITAGOR_00

e709wb.7.1  olang ITA  lang ITA  Prv CMU   " sono molti posti "
e709wb.7.1  olang ITA  lang GER  Prv CMU   "es gibt viele Orte"
e709wb.7.1  olang ITA  lang FRE  Prv CLIPS ""
e709wb.7.1  olang ITA  lang ENG  Prv CMU   "there are many places"
e709wb.7.1                  IF  Prv CMU  a:give-information+existence+object (object-
spec=(place, quantity=many))
e709wb.7.1  comments: Tagged by dmg

e709wb.7.2  olang ITA  lang ITA  Prv CMU   " in cui lei puo sciare in Val-di-Fiemme . "
e709wb.7.2  olang ITA  lang GER  Prv CMU   "wo Sie Ski fahren k~onnen in Val-di-Fiemme."
e709wb.7.2  olang ITA  lang FRE  Prv CLIPS ""
e709wb.7.2  olang ITA  lang ENG  Prv CMU   "in which you could ski in Val di Fiemme"
e709wb.7.2                  IF  Prv CMU  a:give-information+feasibility+activity (who=you,
feasibility=feasible, activity-spec=(skiing, location=name-val_di_fiemme_area))
e709wb.7.2  comments: Tagged by dmg

e709wb.7.3  olang ITA  lang ITA  Prv CMU   " le mostro una mappa . "
e709wb.7.3  olang ITA  lang GER  Prv CMU   "ich werde Ihnen eine Karte zeigen."
e709wb.7.3  olang ITA  lang FRE  Prv CLIPS ""
e709wb.7.3  olang ITA  lang ENG  Prv CMU   "I will show you a map"
e709wb.7.3                  IF  Prv CMU  a:give-information+display+information-object (who=i,
to-whom=you, info-object=(map, identifiability=no))
e709wb.7.3  comments: Tagged by dmg
```

Figure 11: Excerpt from the NESPOLE! Travel & Tourism database

Figure 11 contains an excerpt from the NESPOLE! Travel & Tourism database that shows the annotation of a single speaker turn. Each line in the database begins with a code that includes a dialogue ID, the number of the speaker turn in the dialogue (i.e. a turn ID), and the number of the semantic dialogue unit within the turn (i.e. a semantic dialogue unit ID). The example shown in Figure 11 was the *7*th speaker turn in dialogue *e709wb*, and the utterance contained three semantic dialogue units. The first line in the example contains a comment (indicated by the semantic dialogue unit ID 0) that identifies the source recording of the original utterance. For each semantic dialogue unit, the database contains the transcription of the original recording as well as any manual translations. The *olang* field indicates the language in which the semantic

dialogue unit was originally spoken, and the *lang* field indicates the language of the text contained in a particular line. The *Prv* field indicates where the text was transcribed or translated. Thus, in the example shown in Figure 11, the turn was originally spoken in Italian and then translated into English and German. The turn was transcribed and translated at Carnegie Mellon. In addition to the transcribed and translated text, the database includes the Interchange Format representation for each semantic dialogue unit. The Interchange Format representations are contained in the lines labeled *IF*, and the *Prv* field again indicates where the semantic dialogue unit was tagged. When a semantic dialogue unit has not been tagged, the *IF* line is simply left blank. The database may also include alternative Interchange Format representations if more than one representation may be reasonably assigned to a semantic dialogue unit as well as comments from the annotators.

1.6 Related Work

1.6.1 Other Spoken Language Translation Systems

In addition to the interlingua-based spoken language translation systems mentioned in Section 1.5.1, a number of other large-scale research projects involving international collaboration among multiple partners have focused on the translation of spoken language in limited domains. A few recent examples of such research efforts include Verbmobil ([Wahlster, 2000]), the Spoken Language Translator ([Rayner *et al*., 2000]), and the EuTrans project ([Amengual *et al*., 2000]). Although these examples by no means represent an exhaustive list of past or ongoing spoken language translation research, they provide a glimpse of alternative approaches and domains that have been used successfully in building speech-to-speech translation systems.

The Verbmobil project ([Wahlster, 2000]) was a very large-scale research effort that ran in two phases from 1993 to 2000 and involving a large consortium of academic and industrial partners from Europe, North America, and Asia. The Verbmobil system supported speech-to-speech translation over mobile phones among German, English, and Japanese in three limited domains. The Verbmobil domains included appointment scheduling, travel planning, and personal computer maintenance. Verbmobil combined a broad range of techniques for shallow through deep processing to perform translation ([Bub *et al*., 1997]). A multi-engine architecture was used to incorporate the results of five different translation engines: semantic transfer, dialogue-act-based translation, case-based translation, substring-based translation, statistical

translation. A multi-engine architecture was also used within the semantic transfer system to combine the outputs of three parsers that provided analyses at different levels of linguistic detail: a shallow chunk parser, a probabilistic LR parser, and a chart parser for deep analysis with Head-driven Phrase Structure Grammar. The different parsing and translation engines used and combined manually developed resources and automatic learning techniques to varying degrees. The dialogue-act-based translation engine is particularly relevant since it was the approach most similar the approach used in the NESPOLE! project. Dialogue-act-based translation in the Verbmobil system ([Reithinger and Engel, 2000]) was intended to provide shallow robust translations in the face of noisy spoken input. The dialogue acts used in the Verbmobil system are described in [Jekat *et al*., 1995] and [Alexandersson *et al*., 1998]. In the dialogue-act-based translation module, dialogue acts were identified using statistical language models as described in [Reithinger and Klesen, 1997] and summarized in Section 1.6.4. After the dialogue act for an utterance was identified, manually developed finite state transducers were used to representations for objects such as temporal expressions, locations, etc. Finally, a set of sentence templates and finite state transducers was used to generate target language text.

The Spoken Language Translator ([Rayner *et al*., 2000]) research project ran in three phases from 1992 to 1999. The research partners on the project included SRI International, Telia Research, the Swedish Institute of Computer Science, the University of Geneva, and the Technical University of Crete. The Spoken Language Translator supported speech-to-speech translation among English, Swedish, and French in the domain of air travel planning. Translation was performed using a combination of a shallow and deep processing. Shallow processing involved a simple direct lexical translation of words and/or phrases, and deep processing involved translation via structural transfer using the SRI Core Language Engine ([Alshawi, 1992]). Structural analysis was performed using grammars in the unification formalism of the Core Language Engine. A set of handwritten grammars was initially developed to provide general coverage of important constructions in each source language. Using a small corpus specially constructed to be representative of a new domain, domain-specific coverage shortfalls of the handwritten grammar could be easily identified and repaired ([Rayner and Carter, 1997]). Automatic learning techniques were also used in two ways to improve parsing efficiency and translation quality ([Rayner and Carter, 1996]). First, all possible parses for utterances in a domain-specific training corpus were produced using the Core Language Engine, and the correct

parse for each utterance was identified. The corpus of parsed utterances was used to estimate the likelihood that parse constituents would contribute to a correct parse. Then during parsing of new utterances, constituents that were unlikely to be part of a correct analysis were eliminated. Using only the correct parses from the training corpus, explanation-based learning methods were used to specialize the general grammar for the domain. During specialization, sets of grammar rules that were frequently applied in combination were merged into a single rule, resulting in a non-recursive grammar that produced shallower parse trees of limited depth. The combination of grammar specialization and constituent pruning resulted in much faster parsing and higher translation quality without significant loss of coverage ([Rayner and Carter, 1996]).

The EuTrans project ([Amengual *et al.*, 2000], [Pastor *et al.*, 2001]) was a five year research effort starting in 1996 that involved the collaboration of four European partners from Spain, Germany, and Italy. The EuTrans system supported telephone-based Spanish-English and Italian-English speech-to-speech translation in a hotel reception desk domain. In the EuTrans system, all aspects of translation were performed using stochastic finite state models ([Casacuberta *et al.*, 2001]). Standard finite state acoustic and lexical models for speech recognition were incorporated with a translation model represented as a finite state transducer. The transducer used for the translation model was automatically learned from a parallel corpus of utterances in the source and target languages ([Amengual *et al.*, 2000]). The acoustic, lexical, and translation models were combined into a single hidden Markov model, and the best recognition result and corresponding translation were found simultaneously using Viterbi search.

1.6.2 Segmentation

The main purpose of the analysis approach described in this dissertation is to convert input utterances into the Interchange Format interlingua. As explained previously, the Interchange Format interlingua represents the content of input utterances as a sequence of semantic segments called semantic dialogue units. Semantic dialogue units often correspond roughly to sentences but may also correspond to sentence fragments such as clauses or phrases. Since a single speaker turn often contains multiple semantic dialogue units, one of the critical steps of our analysis approach is the segmentation of input utterances into semantic dialogue units.

A method for identifying semantic dialogue unit boundaries in the JANUS speech-to-speech translation system for the appointment scheduling domain is described in [Lavie et al.,

1997b]. Although the Interchange Format interlingua was not used in that system, a similar speech-act-based interlingua was used. The boundary detection model used acoustic information about silences, human noises, and non-human noises. A statistical model was also used to capture the likelihood of a semantic dialogue unit boundary between each pair of words. The statistical model used three word-bigram frequencies based on a four-word window centered on the potential boundary position to estimate the likelihood. Finally, manually determined lexical cue phrases were used to boost the statistical likelihood estimate. At the time of that work, the robust GLR* parser ([Lavie, 1996a] [Lavie, 1996b]) was used in conjunction with handwritten grammars in a unification-based formalism to produce the feature structure representations of the interlingua. Since the grammars were designed to parse complete semantic dialogue units, classification of semantic dialogue unit boundaries was not strictly necessary. However, the semantic dialogue unit boundary detector served two important purposes. First, high-confidence boundaries were inserted in an utterance prior to parsing in order to reduce parse ambiguity and thereby increase parsability and efficiency. Second, the segmentation scores were used during parsing to prevent the parser from pursuing analyses that created boundaries at positions where the segmentation model was highly confident that no boundary occurred. The use of the semantic dialogue unit segmentation model improved parse efficiency and translation quality.

[Mast *et al*., 1996] describes an approach for segmenting spoken utterances into dialogue act units in the Verbmobil speech-to-speech translation system ([Wahlster, 2000]). The work was done in the setting of the Verbmobil appointment scheduling domain ([Bub and Schwinn, 1996]). The task of segmenting utterances into dialogue act units is very similar to the task of semantic dialogue unit segmentation. Each input utterance must be divided into segments for which the speaker's intention, which is represented at different levels of detail in dialogue acts and domain actions, can be identified. The segmentation approach described in [Mast *et al*., 1996] used a neural network and a language model to estimate the probability of a boundary between each pair of words in an utterance. A boundary was inserted when the probability exceeded a threshold. A neural network with two hidden layers was trained to estimate the probability of a segment boundary based on prosodic features for the six syllables preceding a potential boundary. A polygram language model was also trained to estimate the probability of a segment boundary based on the words preceding a potential boundary. The segmentation models were trained on a set of 96 German dialogues consisting of 2459 speaker turns and 6459 dialogue

act segments and tested on 31 dialogues consisting of 453 speaker turns and 1107 dialogue act segments. A top reported accuracy of 92.5% for the classification of potential boundaries was achieved using a combination of the neural network and language model.

Although a single sentence may contain multiple semantic dialogue units, the problem of detecting semantic dialogue unit boundaries is similar to the problem of sentence boundary detection, especially when the input to the segmenter is produced by an automatic speech recognizer. Stevenson and Gaizauskas ([Stevenson and Gaizauskas, 2000]) describe the application of a memory-based learning approach using TiMBL version 2.0 ([Daelemans *et al*., 2000]) to the problem of identifying sentence boundaries. They point out that text produced by a speech recognizer differs in several important ways from standard text composed by humans. Unlike standard text, speech recognizer output typically does not contain any punctuation or case information. Furthermore, spoken language may contain many phrases and sentence fragments rather than complete grammatical sentences. Finally, text produced by a speech recognizer may contain recognition errors. Stevenson and Gaizauskas simulated automatic speech recognition input (without recognition errors) using a Wall Street Journal corpus with punctuation and case information removed. The training set contained 965 sentence boundaries, and the test set contained 107 sentence boundaries. Their memory-based classifier made a binary decision regarding the presence or absence of a sentence boundary between each pair of words. The input features for the classifier included the word and associated part of speech tag immediately before and after the potential boundary, the probabilities that the words and tags occurred at a boundary, and whether or not the words were stop words. For the sake of comparison, they also included a feature to indicate whether or not the words before and after a potential boundary were capitalized. They report an F_1-measure of 0.76 when case information was included and 0.35 when case information was excluded.

[Gotoh and Renals, 2000] describes the use of a statistical approach to identify sentence boundaries in automatic speech recognition transcripts of broadcast speech from BBC television and radio news programs. The sentence boundary detection problem was formulated as a binary decision whether or not each word is the last word in a sentence. Two statistical models were developed to determine the class of each word (i.e. last word or not last word). The first model was an n-gram language model trained on pre-broadcast program scripts, which of course were similar to but not perfect transcriptions of the actual broadcast. The size of the training data for

the language model was varied between 0.9 million words and 7.2 million words. The second model was a prosody model trained to identify boundaries based on the duration of the pause following a word. The pause duration model was trained on speech recognizer output for 300 hours of broadcasts aligned with their pre-broadcast scripts. In isolation, the best pause duration model outperformed the best language model on all performance measures, and a combination of the two models outperformed either individual model. When applied to unseen automatic speech recognition output, the F_1-measures reported for the best language model, pause duration model, and combined model were 0.46, 0.65, and 0.70 respectively.

The identification of sentence boundaries in standard text composed by humans is also related to the semantic dialogue unit segmentation problem, although the identification of sentence boundaries in text generally involves decisions about whether or not punctuation marks (*.*, *!*, *?*, *:*, *;*, etc.) indicate the end of a sentence. A variety of machine learning approaches have been applied to the task of detecting of sentence boundaries in written English text. [Reynar and Ratnaparkhi, 1997] describes two maximum entropy models designed to determine whether or not 3 punctuation symbols (*.*, *!*, and *?*) mark a sentence boundary. Each word token that contained one of the punctuation symbols was considered as possible sentence boundary. The first model was intended for maximum performance and included some hand-coded knowledge about the language (English) and genre (financial newspaper text) of the task. The model included features about the portions of the token before and after the punctuation mark, the presence of certain characters in the token, whether the token was a known abbreviation, and properties of the words immediately preceding and following the token. The second model was intended to be portable across genres and languages and thus used only features that could be automatically extracted from a training corpus. The features of the second model included the identity of the pieces of the token before and after the punctuation symbol, the identities of the previous and following tokens, and whether any of the tokens were on a list of abbreviations extracted from the training data. Using a training set of 39441 sentences of Wall Street Journal data, the sentence boundary detectors were tested on 20478 sentences of Wall Street Journal data and on the Brown corpus (51672 sentences). Classification accuracies of 98.8% on the Wall Street Journal test set and 97.9% on the Brown corpus were reported for the first, performance-oriented model. Accuracies of 98.0% for the Wall Street Journal test set and 97.5% for the Brown corpus were reported for the second, portable model.

[Walker *et al.*, 2001] describes the application of the maximum entropy approach described in [Reynar and Ratnaparkhi, 1997] to a small corpus extracted from a variety of web sites. The performance of the maximum entropy model was also compared with the performance of two manually developed knowledge-based sentence boundary detectors: a hard-coded program using regular expressions and a grammar-based approach. Each of the sentence boundary detectors was used to determine whether or not 6 punctuation marks (., *!*, *?*, *:*, *;*, and *)*) indicated a sentence boundary. Using a training set containing 9504 sentences and a test set containing 861 sentences, an accuracy of 97.4% was reported for the maximum entropy model. The maximum entropy classifier was also shown to outperform both hand-coded approaches while requiring substantially less development time.

Palmer and Hearst ([Palmer and Hearst, 1997]) describe the application of neural networks and decision trees to the problem of sentence boundary detection in written text using the Satz system. Each punctuation mark (except periods used as decimal points in numbers) was considered a potential sentence boundary position. The Satz system represented each word using a 20-element vector. The vector contained two binary features to indicate whether the word was capitalized and whether the word followed a punctuation mark. The remaining 18 features were used to represent the part of speech distribution of the word. Each feature corresponded to a general part of speech category (e.g. noun, verb, article, abbreviation, etc.). The values of the features were either binary, indicating whether the word could be assigned that category, or numerical, indicating the probability that the word should be assigned that category. The input feature vectors used for each learning approach were similar and included the description vectors for each word in an n-word window centered on a potential boundary. The neural network and decision tree classifiers were tested using a variety of window sizes, training set sizes (several hundred to several thousand examples), and lexicon sizes on a set of 27494 potential boundaries extracted from an English Wall Street Journal corpus. The best reported accuracies of the neural network and decision tree classifiers were 98.1% and 99.0% respectively. Similar levels of performance were also reported for two German news test sets and a French Hansards test set.

A transformation-based learning approach to the task of identifying sentence boundaries in Modern Greek text is described in [Stamatatos *et al.*, 1999]. The sentence boundary detector learned to determine whether or not four punctuation symbols (., *!*, *;*, and …) marked sentence boundaries. The sentence boundary detector initially assumed that all punctuation symbols

indicated a sentence boundary. Automatically learned rules were then applied in two passes to modify the classification of the potential sentence boundaries. The set of rules applied during the first pass changed the classification of punctuation marks that met certain learned triggering criteria from boundary to non-boundary. The set of rules applied during the second pass changed the classification of punctuation marks from non-boundary to boundary when triggering criteria were met. The triggering criteria used in the rules included features (e.g. word length, characters, etc.) extracted from the tokens immediately preceding and immediately following a potential boundary position. An accuracy of 99.4% was reported for a classifier trained on 7274 sentences and tested on 8736 sentences.

[Mikheev, 2000] addresses the sentence boundary detection problem using a trigram part of speech tagger based on hidden Markov models and maximum entropy techniques ([Mikheev, 1997]). In this approach, sentence boundary detection was treated as part of the part of speech tagging problem. Punctuation marks were treated as separate tokens and tagged in the same way as word tokens. Each punctuation mark token was tagged as an end of sentence marker, part of an abbreviation, or both simultaneously. The part of speech tagger was also augmented with an automatically induced list of abbreviations and an automatic proper name handling mechanism. The performance of the part of speech tagger/sentence boundary detector was tested using the Brown Corpus and Wall Street Journal corpus from the Penn Treebank. The tagger was trained on the Brown Corpus and tested on the Wall Street Journal corpus and vice versa. The best reported accuracy for sentence boundary identification was 99.8% on the Brown Corpus and 99.69% on the Wall Street Journal corpus.

1.6.3 Domain Action Classification

Automatic classification of domain actions using trainable machine learning techniques is a second critical component of our analysis approach. One early approach to training classifiers to identify Interchange Format domain actions is described in [Fukada *et al.*, 1998]. Domain actions were identified by combining the output of word unigram language models for speech acts and complete concept sequences. Illegal combinations of speech acts and concept sequences were assigned a probability of 0. The input to the language model classifiers consisted of the text for a single semantic dialogue unit. Before presentation to the classifiers, the text of the semantic dialogue unit was preprocessed. Function words were removed, words were replaced with

category labels when appropriate (e.g. Monday → *day-of-week*), and common phrases were collapsed into single tokens (e.g. how many → *how_many*). The models were tested on English with 64 training dialogues and 50 test dialogues and on Japanese with 84 training dialogues and 42 test dialogues. For English input, classification accuracies of 58.8%, 57.5%, and 39.7% were reported for identification of speech acts, concept sequences, and domain actions respectively. For Japanese, classification accuracies of 79.9%, 71.6%, and 57.6% were reported. Replacing the word unigram models with word bigram models improved speech act classification performance but degraded concept sequence classification performance. Domain action classification performance improved for Japanese and decreased for English. The experiments reported in [Fukada *et al.*, 1998] demonstrated the possibility of automatically classifying Interchange Format domain actions. However, neither argument identification nor semantic dialogue unit segmentation were addressed.

Munk made a first attempt at combining handwritten grammars and automatic domain action identification for producing Interchange format representations ([Munk, 1999]). Munk developed a prototype system called SALT that produced C-STAR II Interchange Format representations from spoken input utterances in the C-STAR II speech-to-speech translation system. The SALT system used several stages to produce Interchange Format representations. In the first stage of processing, a two-layer hidden Markov model was used to label top-level arguments and speech acts in an input utterance. The input to the model consisted of an utterance tagged with parts of speech. The 400 most frequent words in the training corpus were used directly in the model, and the remaining words were replaced with part of speech tags. The first layer of the hidden Markov model segmented and labeled the utterance at the argument level. Each argument-level segment was assigned a top-level argument label. The second layer of the hidden Markov model segmented and labeled the utterance at the speech act level. Each speech act segment was equivalent to a semantic dialogue unit. In the second stage of processing, a neural network was used to identify the complete concept sequence associated with each speech act segment. The features used by the neural networks were the argument and speech act labels assigned to the segment by the hidden Markov model and the same set of frequent words used in the hidden Markov model. Finally, semantic grammars were used to parse for arguments in the argument segments identified by the hidden Markov model. Parsing within each argument

segment was restricted to the use of grammar rules associated with the argument label assigned by the hidden Markov model.

The SALT system was evaluated on English dialogues from the C-STAR II travel planning domain. The classifiers were trained on 64 transcribed dialogues and tested on 50 additional transcribed dialogues. The best reported accuracy for the speech act classifier was 59.9% using a word trigram model and a speech act bigram model. The best reported concept sequence classification accuracy was 87.2%. No results were reported for semantic dialogue unit segmentation performance or for performance on complete domain actions. The quality of the English-to-English paraphrases produced by the C-STAR II translation system using SALT was also compared with the quality of the paraphrases produced when handwritten domain-action-level semantic grammars were used for parsing. The test set consisted of 200 to 300 unseen utterances that were automatically recognized and manually transcribed. The method for judging end-to-end translation (paraphrase) quality was similar to that described in Section 5.2. SALT achieved 50.5% acceptable paraphrases on recognized input and 57.3% on transcribed input. The domain action grammar approach achieved 61.6% acceptable performance for recognized input and 73.7% acceptable translations for transcribed input.

Although the performance of the SALT system did not reach the level of the strictly grammar-based approach, it demonstrated the feasibility of combining grammar-based parsing with machine learning techniques for producing Interchange Format representations. One of the main weaknesses of the SALT approach was a tendency to oversegment at both the argument level and the semantic dialogue unit level (i.e. too many arguments and semantic dialogue units were produced). The mechanism for ensuring that legal Interchange Format representations were produced was also relatively weak. Empty or duplicate argument segments and domain actions were removed from SALT's output. However, no tests were performed to ensure that the domain action and arguments could be combined to form a valid Interchange Format representation.

[Cattoni *et al.*, 2001] describes the application of statistical language models to the domain action classification task for producing Interchange Format representations in the Italian translation server in the NESPOLE! system. Word bigram models are trained for each domain action in the training data. In order to label a semantic dialogue unit with a domain action, the likelihood of each domain action is computed using the word bigram models. The domain action with the highest likelihood is then assigned to the semantic dialogue unit. Arguments are

identified using manually developed recursive transition networks. Depending on the arguments for which they are designed, the networks may represent simple word sequences, regular expressions, or bigram language models. A hidden Markov model decoder ([Brugnara and Federico, 1997]) is used to identify the most probable argument sequence. This approach also incorporates constraints from the Interchange Format specification to find the most likely domain action and argument sequence that form a valid Interchange Format representation. Among the domain actions that license the most arguments, the one with the highest probability is selected. Any unlicensed arguments are then removed from the Interchange Format representation.

1.6.4 Speech Act and Dialogue Act Classification

The task of identifying domain actions defined in the Interchange Format interlingua is similar to the task of dialogue act (or speech act) identification. Dialogue acts have been used as one level of discourse modeling in human-computer dialogue systems, automatic tutoring systems, and machine translation systems. Like domain actions, dialogue acts are used to represent a speaker's intention for an utterance. Because domain actions are designed for use in automatic translation in task-oriented domains, they include both a domain-independent speech act and a set of domain-specific concepts. Although the specific sets of dialogue acts have varied in different lines of research, the dialogue acts have typically been similar to the speech acts used in the Interchange Format representation. Thus, domain actions provide a more detailed representation of speaker intention and a correspondingly larger set of classes for the classification problem. A variety of automatic learning techniques have been applied to the task of dialogue act classification. Direct comparisons of the dialogue act classification results for the various approaches with each other generally cannot be made because different corpora and dialogue acts were used in most of the work. Direct comparisons of dialogue act classification performance with classification performance results for Interchange Format domain actions also are not possible for similar reasons. The set of domain actions is much larger than any of the sets of dialogue acts that have been studied previously. The set of speech acts from the Interchange Format is much closer to the sets of dialogue acts that have been studied.

One learning approach that has been used successfully for the identification of dialogue acts is *n*-gram language modeling. [Reithinger and Klesen, 1997] describes the application of simple *n*-gram language models to the task of dialogue act identification. The task of the

classifiers was to identify dialogue acts in English and German appointment scheduling dialogues using the Verbmobil-I tag set ([Jekat *et al*., 1995]). The Verbmobil-I dialogue act set includes 43 unique dialogue acts that are organized hierarchically and can be collapsed into a reduced set of 18 abstract dialogue acts. The issue of segmentation was not addressed, and the models assumed that each input segment consisted of a single dialogue act. The best results were reported for a linear interpolation of unigram and bigram language models combined with a simple *n*-gram model of dialogue act history. The German data, which consisted of manually transcribed dialogues, was divided into a training set consisting of 350 dialogues and a test set containing 87 dialogues with 2912 dialogue act segments. Using the German data, language models were trained for the full 43 dialogue act tag set as well as the reduced 18 dialogue act tag set. The dialogue act with the highest likelihood according to the language models was assigned to each utterance. An accuracy of 65.18% was reported for the full tag set with 43 dialogue acts, and an accuracy of 67.18% was reported for the reduced tag set with 18 dialogue acts. The language model approach for the reduced tag set was also tested on an English data set consisting of 143 training dialogues and 20 test dialogues containing 328 dialogue act segments. The reported accuracy for the English test data with the reduced tag set was 74.7%.

[Warnke *et al*., 1997] describes an approach that integrates the tasks of dialogue act classification and dialogue act segmentation for the Verbmobil appointment scheduling domain. The probabilities of segment boundaries were estimated using neural networks with prosodic features and polygram language models as described in [Mast *et al*., 1996]. For dialogue act classification, polygram language models for each of the 18 abstract dialogue acts in the reduced tag set from Verbmobil-I and a bigram model of dialogue act sequences were used to estimate the probabilities of the dialogue acts for each segment. All of the models used only information from the current speaker turn. The combination of dialogue act segmentation and classification was accomplished using an A*-search. Each node in the A*-search tree represented a unique word sequence, a segmentation of the word sequence into dialogue act units, and classification of the dialogue units. At each search step, the best node in the search tree was expanded by adding a new word and segmentation hypothesis to the end of the word sequence along with possible dialogue act classifications if the node included a segment boundary after the previous word. The score for each node was computed using a weighted combination of the probabilities computed by the segmentation classifiers and the domain action language models. The models were trained

and tested on a set of German appointment scheduling dialogues. The training set included 96 dialogues with 2459 speaker turns and 6496 dialogue act segments, and the test set included 31 dialogues with 39 speaker turns and 992 dialogue act segments. The performance of the integrated dialogue act segmentation and classification approach was compared with the approach described in [Mast *et al*., 1996] in which segmentation was performed prior to dialogue act classification. The best reported dialogue act classification using the two-step approach was 61.9%, and the best classification accuracy using the combined approach was 61.5%. When segmentation errors (i.e. inserted and deleted segments) are also taken into account, the two-step approach gave a top dialogue act recognition accuracy of 53.0%, and the combined approach achieved 53.4% recognition accuracy.

Several other learning techniques have also been applied to the dialogue act classification task. The use of semantic classification trees ([Kuhn and De Mori, 1995]) for dialogue act classification in the Verbmobil appointment scheduling domain was described in [Mast et al., 1995]. Semantic classification trees are binary decision trees with a yes/no question at each node. Each yes/no question is a regular expression built using keywords and non-empty gaps between keywords that can be matched to an input. A semantic classification tree was trained to classify 16 dialogue acts in transcribed English and German appointment scheduling dialogues. The German and English data sets contained 214 dialogues with approximately 6000 dialogue acts and 56 dialogues with approximately 1600 dialogue act segments, respectively. Each data set was divided into a training set containing 80% of the data and a test set containing the remaining 20% of the data. Classification accuracies of 46% for German and 59% for English were reported.

The application of neural networks to the task of dialogue act classification has been explored by several researchers. [Wermter and Löchel, 1996] describes the use of simple recurrent networks ([Elman, 1990]) for dialogue act classification within the SCREEN architecture ([Wermter and Weber, 1997]). The goal of the SCREEN system was to provide robust analysis for spoken language by learning flat syntactic, semantic, and dialogue level representations. After each word in a spoken turn was assigned a basic and abstract syntactic and semantic category using simple recurrent networks, manually developed segmentation heuristics were used to break the turn into dialogue act level units (called utterances in SCREEN). Using a set of 8 dialogue acts for a meeting scheduling domain, each word in a dialogue act unit was

assigned a dialogue act using another simple recursive network. The network had one input unit for each dialogue act, and the input to the network for each word was a smoothed vector of the plausibility (estimated using the training data) of each dialogue act for the given word. The network contained one hidden layer with 7 units and a context layer that stored the activation level of the hidden units following the previous word. The network had one output unit for each dialogue act, and the dialogue act with the highest activation was assigned to the input word. The dialogue act for each segment was assigned as the majority class among the words in the segment. The segmentation heuristics and dialogue act classifier were tested using a set a 184 speaker turns containing 314 dialogue act segments. [Wermter and Löchel, 1996] reports that 84% of the 184 turns were correctly segmented using the segmentation heuristics. 100 dialogue act segments were used for training the dialogue act network, and the remaining 214 dialogue act segments were used as test data. An overall classification accuracy of 82.0% for the training data and 79.0% for the test data was reported.

The use of neural networks for dialogue act classification in the Verbmobil appointment scheduling domain is described in [Kipp, 1998]. The 18 abstract dialogue acts in the reduced Verbmobil-I tag set mentioned above were used for the classification task. One neural network was trained to identify each of the 18 dialogue acts. Each word was identified by one of 15 part of speech categories represented using an input vector of 216 units. The representation of a word within a particular part of speech category was determined by the relative importance of the category for the classification task. Categories of high importance used one input unit to represent each possible word in the category. Categories of medium importance used the binary representation of each word's position in the list of possible words for the category. Categories of low importance used a single input unit to indicate that a word was from the category. The network for each dialogue act contained a hidden layer and a recurrent context layer. The output layer of each network contained one unit for a "yes" classification and one unit for a "no" classification. An additional network with a hidden layer and a context layer was trained to combine the outputs of the individual dialogue act networks for a complete utterance and assign a dialogue act for the whole utterance. The input and output layers consisted of one unit per dialogue act. The activation of each input unit was the average difference between the "yes" and "no" units over all the words in an utterance, and the dialogue act with the highest output activation was assigned for the utterance. The networks were trained and tested on a corpus of

467 German dialogues from the Verbmobil appointment scheduling domain. The corpus was divided into a training set containing 350 dialogues and 10766 utterances, a validation set containing 87 dialogues and 2903 utterances, and a test set containing 30 dialogues and 852 utterances. The best network achieved a reported recall of 60.45% on the test set, while the statistical *n*-gram model from [Reithinger and Klesen, 1997] achieved a recall of 67.53% on the same test set. When the input representation for each word was changed from the part of speech category vector to a vector containing the probability of each dialogue act computed using a unigram model, a recall of 66.31% was reported for the neural network approach.

[Stolcke *et al.*, 2000] describes the integration of several learning approaches for identifying dialogue acts in spoken dialogues as well as an integration of dialogue act modeling with speech recognition to improve the performance of both tasks. The work was conducted using a portion of the Switchboard corpus ([Godfrey *et al.*, 1992]) consisting of 1155 telephone conversations between two English-speaking strangers. Each speaker turn in the corpus was manually divided into dialogue act segments, and a set of 42 dialogue acts based on the Switchboard-DAMSL dialogue act tags ([Jurafsky *et al.*, 1997]) was used to tag the corpus. The dialogue act models were designed under the assumption that the segmentation of speaker turns into dialogue act segments was given. The dialogue act models were trained on 1115 dialogues containing 198,000 dialogue act segments and 1.4 million words and tested on 19 dialogues containing 4000 dialogue act segments and 29,000 words. The Switchboard corpus used in this work differed from the majority of corpora used in other work on dialogue act identification in several ways. The dialogues in the Switchboard corpus were not task-oriented. The dialogue participants spoke about general interest topics rather that attempting to perform a particular task such as scheduling appointments or making travel arrangements. The size of the training corpus was also much larger than the corpora used in other work, and the size of the dialogue act tag set was larger than that used in most other work (with the exception of full Verbmobil tag set).

A hidden Markov model framework was used to incorporate several types of statistical models for dialogue act classification ([Stolcke *et al.*, 2000]). Low order *n*-gram models were used to model the probability of dialogue act sequences. When manual transcriptions of dialogues were used as input, trigram language models were trained for each dialogue act to estimate dialogue act likelihoods based on word information. When the output of an automatic speech recognizer was used as input to the dialogue act classification model, the acoustic scores

from the top 2500 hypotheses produced by the recognizer were incorporated into the model. Without the dialogue act sequence model, classification accuracies of 54.3% and 42.8% were reported for transcribed and automatically recognized input, respectively. The accuracies improved to 70.6% and 64.3% when a bigram dialogue act sequence model was included. When the automatic recognition model used only the single best recognizer hypothesis, accuracy with a bigram dialogue act sequence model fell to 61.5%. [Stolcke *et al*., 2000] also describes the incorporation of prosodic information into the hidden Markov model framework to supplement the word-based models. Using features such as duration, pause information, pitch, energy, speaking rate, and speaker gender ([Shriberg *et al*., 1998]), decision trees and neural networks were trained to identify 6 dialogue act classes including the 5 most frequent dialogue acts and a single class for all remaining dialogue acts. The training data contained an equal number of examples of each of the 6 classes. An accuracy of 45.4% was reported for the decision tree classifier, and a top accuracy of 46.0% was reported for the best neural network classifier. Since the performance difference between the two approaches was small, the decision tree classifier was incorporated in the hidden Markov model framework. Dialogue act classification accuracies of 38.9% without a dialogue act sequence model and 49.7% with a bigram dialogue act sequence model were reported. The decision tree prosody model was also combined with the automatic recognition model. Without a dialogue act sequence model, the combination of the word-based and prosody-based models resulted in a large improvement in accuracy (to 56.5%) over either individual model. However when a dialogue act sequence model was also included, the combination of the word-based and prosody-based models produced only a small improvement in accuracy (to 65.0%).

It should be noted that the models used in all of the experiments reported in [Stolcke *et al*., 2000] used an entire dialogue as evidence for assigning dialogue act tags and could thus only be used during offline processing after the completion of a dialogue. All of the approaches to dialogue act classification mentioned previously in this section were suitable for online classification. Although, the hidden Markov model framework used for the models could be adapted for use in an online dialogue classification system by using only evidence from preceding turns in a dialogue, no performance results were reported for such models.

Transformation-based learning ([Brill, 1995]) is another machine learning technique that has been successfully applied to the dialogue act identification task ([Samuel *et al*., 1998a],

[Samuel *et al.*, 1998b]). In transformation-based learning, the system learns a set of rules that are applied in sequence to assign a class to each input to the classifier. The first rule learned typically assigns a default class (e.g. the most frequent class in a training corpus) to each training example. At each learning step, the learner examines all possible class-modification rules from a set of provided rule templates and selects the rule that results in the largest improvement in classification accuracy for the training examples. The learned rules are then applied in sequence to classify unseen inputs. [Samuel *et al.*, 1998a] describes the application of transformation-based learning to dialogue act classification using the 18 abstract dialogue acts in the reduced Verbmobil-I tag set described above. A Monte Carlo version of transformation-based learning that samples from the set of possible class-modification rules rather than testing all possible rules was employed to improve training efficiency without degrading performance. The features used to define the rule templates for the transformation-based learners included manually and automatically identified dialogue act cue phrases, word *n*-grams from the utterance, speaker information, punctuation marks, utterance length, and dialogue acts from preceding and following utterances. Dialogue act classifiers were trained and tested using the same English corpus from the Verbmobil appointment scheduling domain that was used in [Reithinger and Klesen, 1997]. The training set consisted of 2701 utterances from 143 dialogues, and the test set consisted of 328 utterances from 20 dialogues. As mentioned above, [Reithinger and Klesen, 1997] reported a top accuracy of 74.7% using *n*-gram language models. The average accuracy over 5 training runs using the Monte Carlo version of transformation-based learning was reported to be 75.12%. As was the case in the work reported in [Stolcke *et al.*, 2000], the rule templates for the transformation-based classifiers described in [Samuel *et al.*, 1998a] had access to features that could only be found in an offline setting, namely the dialogue acts for following utterances.

Recently, memory-based learning has also been applied to dialogue act classification as reported in [Lendvai *et al.*, 2003]. Dialogue act identification was performed as part of a shallow interpretation of user turns in a telephone-based Dutch spoken dialogue system for accessing train timetable information. Thus, unlike the other work mentioned in this section, the dialogues used in this study were between a human and an automated information system rather than between two humans. A corpus consisting of 3738 pairs of system prompts and user responses from 441 dialogues was manually with a semantic tag set. Each complete user utterance (no

segmentation) was assigned a tag containing three elements. The tag consisted of a dialogue act, a set of semantic slots filled by the response, and a problem flag. A set of 4 dialogue acts was used for tagging user utterances: giving information/slot-filling, explicit "yes", explicit "no", and acceptance of incorrect information. A slot-filling dialogue act could occur simultaneously with each of the other three dialogue acts for a total of 7 possible dialogue acts. A set of 5 semantic slots could be filled: departure station, arrival station, date, time of day, and hour. The problem flag was used to indicate the user's awareness of a communication problem. The system prompts were also tagged using a different small set of dialogue acts (e.g. question, explicit verification, repetition, etc.) and the same set of slots. A total of 94 unique tags were used for annotating the system prompts and user responses. The number of tags used only for user responses (i.e. the number of classes learned by the classifier) was not reported. A memory-based learner using the IB1 (traditional k-nearest neighbor) algorithm in TiMBL ([Daelemans *et al*., 2002]) was trained to identify the complete tag for user utterances. The input features for the classifier included the tags for the 10 previous system prompts and the words from the 2 previous system prompts. Additional input features for representing an utterance were extracted from the output of an automatic speech recognizer and included the words and branching factors from the word lattices for the current and previous user responses as well and the confidence score of the best hypothesis for the current turn. Finally, prosodic properties of the current utterance including pitch, energy, duration, and tempo were included as input features. A 10-fold cross-validation setup was used for testing the classifier, and the best reported accuracy for matching the complete tag was 73.5% (averaged over 10 folds). For the dialogue act portion of the tag, the reported precision, recall, and F_1-measure were 93.3%, 89.2%, and 91.2%, respectively. Although the results of this study verify the usefulness of memory-based learning for the task of identifying semantic tags such as dialogue acts, they are not directly comparable with the results for human-human dialogues. Of course the tag sets, corpora, and dialogue domains differ. Additionally, in online versions of human-machine dialogues, the tags for the system prompts would not be prone to error, since the system would know what it said, whereas the tags for both sides of human-human dialogues would be uncertain.

Chapter 2 Hybrid Analysis Approach

2.1 Overview

The hybrid analysis approach described in this dissertation uses a combination of phrase-level parsing with semantic grammars and machine learning techniques to transform input utterances into the domain-action-based Interchange Format interlingua representation described in Section 1.5.4. As mentioned previously, the Interchange Format representation is comprised of four main components: a mandatory speaker tag, a mandatory speech act, an optional sequence of concepts, and a (possibly empty) set of arguments. For the purpose of analysis, the speaker tag is assumed to be given since it is determined by the role of the speaker in a dialogue. Thus, the goal of the analyzer is to identify the domain action (speech act plus concept sequence) and arguments for each semantic dialogue unit in an input utterance.

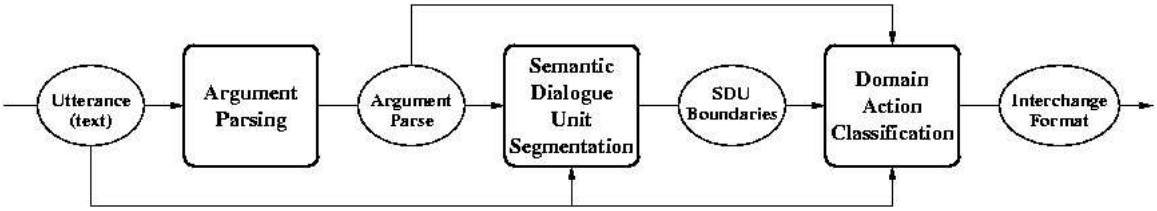

Figure 12: Architecture of the Hybrid Analyzer

The general architecture of the hybrid analyzer is shown in Figure 12. The input to the analyzer is a text string containing an utterance to be translated. The text string will generally originate from an automatic speech recognizer run on a user's spoken utterance, from an input utterance typed by a user, or from manual transcription of a user's spoken input. The hybrid analyzer operates in three stages to construct an Interchange Format representation for each input utterance. In the first stage, the hybrid analyzer uses phrase-level semantic grammars and a robust parser to parse an input utterance. The output of the first stage of processing is an *argument parse* which may include unparsed words as well as trees that cover Interchange Format arguments, useful phrases, and complete domain-independent semantic dialogue units. In

the second stage of operation, the hybrid analyzer segments the input utterance and argument parse into a sequence of semantic dialogue units. This step is required because the Interchange Format represents the meaning of utterances at the level of semantic dialogue units. However, since human utterances often contain multiple semantic dialogue units, the semantic dialogue unit boundaries in an utterance must be identified before domain actions can be assigned. Finally, in the third stage of processing, the hybrid analyzer applies automatic classification techniques along with constraints from the Interchange Format specification to identify the domain action for each semantic dialogue unit in the utterance.

2.2 Motivation

Section 1.5.1 described several previous machine translation systems that used the Interchange Format or a similar interlingua. In each of these systems, analysis from source language into interlingua was performed using manually developed grammars written to parse full domain actions. Purely grammar-based approaches to analysis may be able to provide highly accurate analyses because the grammar writers can often incorporate domain knowledge into the grammars. However, many months or years of effort by expert human grammar writers are generally required to develop and maintain an effective grammar, even for a limited domain. Furthermore, no matter how comprehensive a grammar is designed to be, it is typically infeasible for the grammar to completely cover a domain. For spoken language, where inputs are noisy and where speakers tend to be creative and adhere less strictly to grammatical conventions, the problem is further exacerbated. Grammar-based parsers are typically unable to produce a parse for inputs that deviate from the grammars, although robust parsing techniques can help to smooth the degradation in performance ([Lavie, 1996a], [Lavie, 1996b]). On the other hand, machine learning approaches generally degrade gracefully in the face of noisy input. Unlike grammar-based parsers, which must adhere strictly to the rules contained in a grammar, machine learning techniques are able to generalize beyond the data on which they have been trained. However, incorporating domain knowledge into machine learning approaches may be more difficult than in grammar-based approaches, and machine learning systems may be less accurate on data for which grammars were specifically developed. Furthermore, machine learning approaches may require a large amount of training data in order to achieve reasonable performance. Our hybrid

analysis approach attempts to take advantage of the benefits of hand-written grammars and machine learning while minimizing the disadvantages.

The decision to develop grammars for parsing arguments and to use machine learning for identifying the domain action in the hybrid analyzer was motivated by several factors. First, this division follows naturally from the way arguments and domain actions are used in the Interchange Format representation. As described in Section 1.5.4, the Interchange Format uses a feature-value representation to encode detailed semantic information from an utterance. Since argument values may be atomic or contain nested subarguments, the argument representation is basically a tree structure. Parsing is a natural approach to use for building such structures. On the other hand, domain actions in the Interchange Format are essentially flat classes whose purpose is to categorize each semantic dialogue unit based on the intention of the speaker. Thus, the domain action is a natural level at which to apply automatic classification techniques.

In addition to this natural split based on the Interchange Format representation, the set of arguments is relatively small and fixed compared to the set of possible domain actions, making it easier to develop and maintain a grammar for arguments. The Interchange Format specification for the combined Travel & Tourism and Medical Assistance domains defines only 227 top-level arguments. On the other hand, the specification defines many thousands of legal domain actions. A comparison of the growth in the number of unique domain actions and arguments as the amount of available data increases provides further evidence of this difference.

The graph in Figure 13 shows the growth in the number of unique domain actions and arguments (types) required to cover increasing amounts of data. The data used in the comparison consisted of complete dialogues in English, Italian, and German from the NESPOLE! Travel & Tourism domain and the NESPOLE! Medical Assistance domain. The Travel and Medical databases contained 8478 semantic dialogue units and 4088 semantic dialogue units respectively. We randomly extracted samples of semantic dialogue units from tagged data for each domain and counted the number of unique domain actions and arguments found in the sample. Each sample contained an equal number of semantic dialogue units from each domain, and we extracted 10 samples for each sample size. The y-axis shows the mean over the 10 samples of the number of unique domain actions or top-level arguments required to cover a sample. The x-axis shows the sample size in semantic dialogue units. As the graph illustrates, the growth in the number of arguments remains relatively flat whereas the number of domain actions continues to

grow at a steady rate. We observed similar trends in the growth curves for each individual domain. We also note that even at the highest sample size, the number of arguments observed was still well below the total number of arguments defined in the Interchange Format specification.

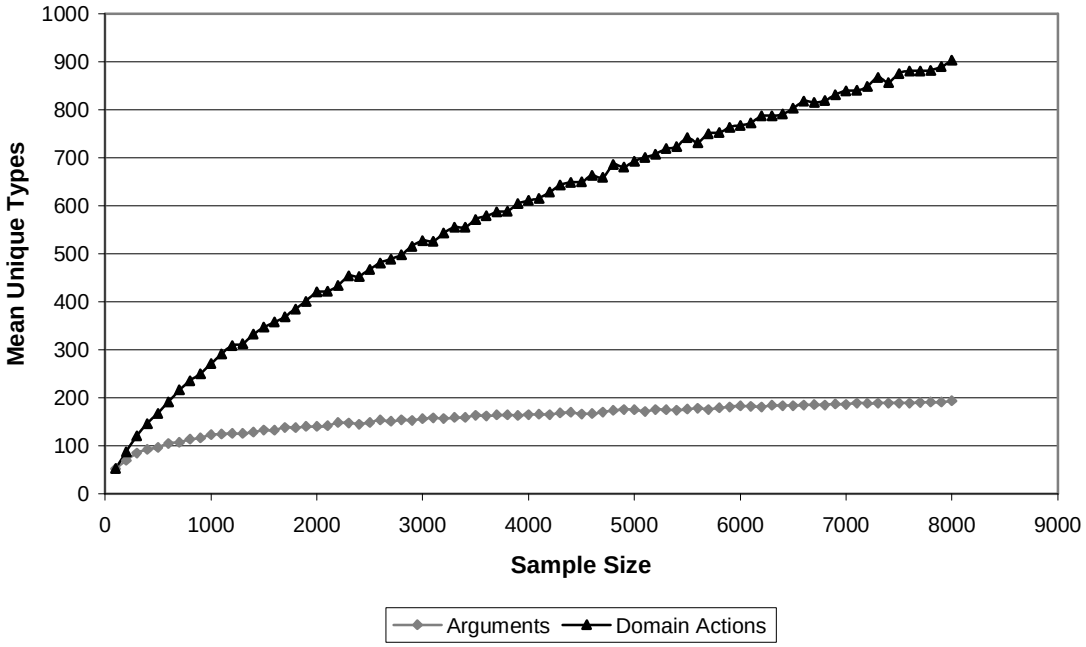

Figure 13: Growth of domain actions and top-level arguments in the combined NESPOLE! Travel and Medical domains

We also estimated the rate at which new unique domain actions and top-level arguments (types) are observed in unseen data using the same data set. Starting with an empty seen data set, we randomly selected 100 unseen semantic dialogue units (50 from each domain) and counted the number of unique domain actions and arguments in the sample not covered by the seen data. We then added the sample to the seen data set and selected a new sample of 100 unseen semantic dialogue units from the remaining data. We repeated this process until there was no data remaining, and we repeated the entire sampling procedure 10 times. The graph in Figure 14 illustrates the rate at which new domain action and argument types were observed as the amount of seen data was increased. The y-axis shows the mean number of new domain actions or

arguments in 10 unseen samples, and the x-axis shows the size of the seen data set. The number of new arguments per 100 unseen semantic dialogue units dropped below 5 after only 500 semantic dialogue units were seen and below 1 after 3000 semantic dialogue units were seen. In contrast, the number of new domain actions leveled off around 6 per 100 unseen semantic dialogue units even after 8000 semantic dialogue units had been seen. This demonstrates that the set of arguments remains mostly fixed as unseen data is encountered, whereas the set of domain actions continues to grow (albeit at a relatively low rate).

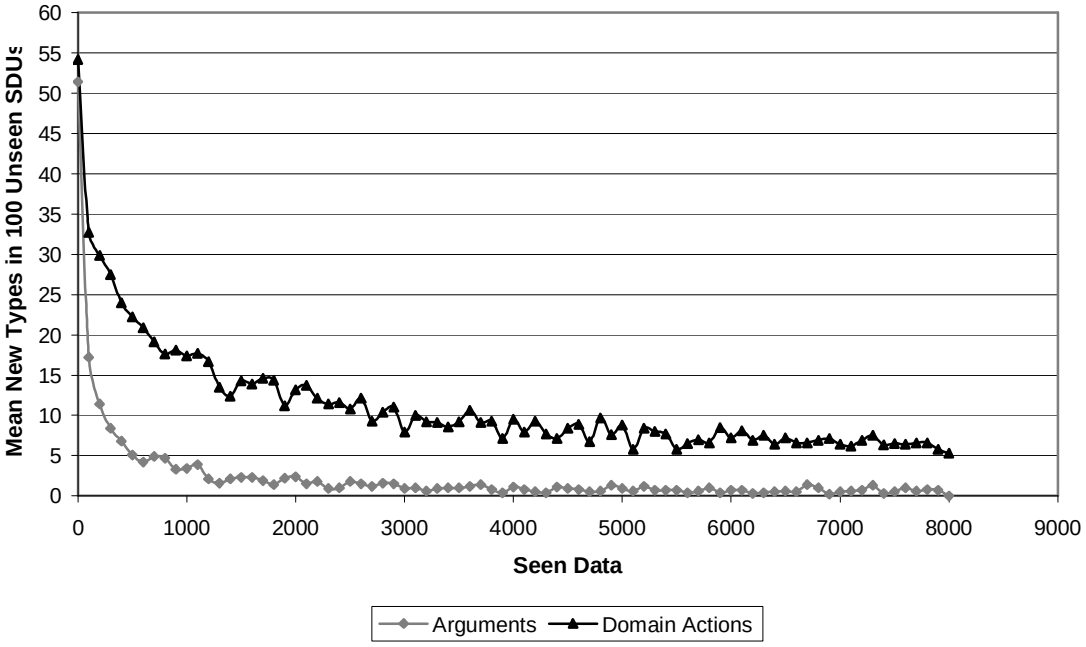

Figure 14: Rate of new domain actions and top-level arguments in the combined NESPOLE! Travel and Medical domains

Another motivating factor in the decision to develop argument grammars and classify domain actions was the fact that arguments tend to be less domain-dependent than domain actions. Of the 227 top-level arguments defined in the Interchange Format specification, only 22 were added specifically for the Medical Assistance domain. The remaining 205 arguments were either domain-independent (possibly added in the process of defining the Medical domain) or specific to the Travel & Tourism domain. We further examined the extent to which arguments

were domain-independent by looking at the overlap between the arguments used in complete dialogues from the NESPOLE! Travel & Tourism data and those used in the NESPOLE! Medical Assistance data.

	Argument Types	Type Overlap	Argument Tokens	Token Overlap
NESPOLE! Travel & Tourism	157	88 (56.1%)	11478	10053 (87.6%)
NESPOLE! Medical Assistance	141	88 (62.4%)	7938	6294 (79.3%)
Combined	210	88 (41.9%)	19416	16347 (84.2%)

Table 1: Argument overlap in the NESPOLE! Travel and Medical domains

	Domain Action Types	Type Overlap	Domain Action Tokens	Token Overlap
NESPOLE! Travel & Tourism	879	171 (19.5%)	8478	6004 (70.8%)
NESPOLE! Medical Assistance	459	171 (37.3%)	4088	2743 (67.1%)
Combined	1167	171 (14.7%)	12566	8747 (69.6%)

Table 2: Domain action overlap in the NESPOLE! Travel and Medical domains

Table 1 shows argument overlap, and Table 2 shows domain action overlap. The Overlap columns show the number of types and tokens shared between the Travel and Medical domains as well as the percentage of the data covered by those types and tokens. The types in the Overlap set can be considered to be domain-independent, at least with respect to the Travel and Medical domains. Some of the non-Overlap types may be domain-independent but occur rarely and thus only appear in the data for one domain. The tables show that the coverage of arguments and domain actions shared by both domains is high, indicating the general domain portability of Interchange Format representation. For each data set individually and for the combined data set, the overlap for both types and tokens was higher for arguments than for domain actions. The

domain-independent arguments covered 84% of the combined data. This means that argument grammars developed for one domain (with coverage for the domain-independent arguments as well) should be largely applicable to a new domain. For example, a grammar for the Travel & Tourism domain could in principle cover all of the argument tokens in the Travel & Tourism data as well as the Overlap argument tokens in the Medical Assistance data shown in Table 1. These argument tokens account for about 91.5% of the argument tokens in the combined data. Of course, the grammar is not likely to have perfect coverage in practice, and a few previously unforeseen domain-independent arguments may have been discovered while analyzing the Medical data. Nevertheless, the grammar should still provide reasonable coverage before explicit expansion to the Medical domain. As mentioned in the discussion of future work in Section 6.2.1.2, evaluating the coverage of the Travel & Tourism argument grammars on data from the Medical Assistance domain would allow us to test this experimentally.

Based on discussions with grammar writers with years of experience in developing full domain action grammars, our intuition was that argument grammars would require less effort to develop and port to new domains than full domain action grammars. Our analysis of the NESPOLE! databases provides support for this intuition. The set of arguments remains relatively fixed while the set of domain actions continues to grow as new data is encountered. In addition, domain-independent arguments cover a large percentage of the arguments found in the data.

2.3 Argument Parsing

In the first stage of processing in the hybrid analysis approach, input utterances are parsed with a robust parser using phrase-level semantic grammars developed by human grammar writers. We refer to this stage of analysis as *argument parsing*. The primary purpose of argument parsing is to identify the Interchange Format arguments in an input utterance. Phrases which may not appear directly in the Interchange Format representation but which may provide useful information for identifying the domain action are also parsed during this stage. Finally, some simple domain actions that are domain-independent are also parsed during the argument parsing stage. Examples of each of these kinds of phrases appear in the following subsections.

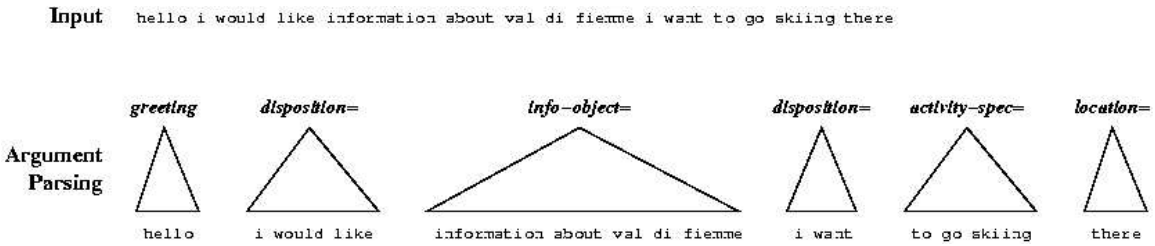

Figure 15: Argument parsing example

Figure 15 illustrates the result of argument parsing on the utterance *"hello i would like information about val di fiemme i want to go skiing there"*. The argument parse contains one tree covering a full domain action (for the greeting "hello") and five trees that cover Interchange Format arguments. The domain action (*greeting*) is shown at the root of the tree that covers "hello". For the remaining 5 trees, the figure shows only the top-level argument label at the root of the tree. The complete argument parse would also contain any values or subarguments found under the top-level elements.

2.3.1 SOUP

The SOUP robust parser [Gavaldà, 2000] is used for argument parsing in our implementation of the hybrid analyzer. SOUP is a stochastic, chart-based, top-down robust parser that was specifically designed to provide real-time analysis of spoken language using large semantic grammars. The grammars supported by SOUP are strictly context-free and are encoded internally using probabilistic recursive transition networks. Because the format of the output from the SOUP parser does not match the format of the Interchange Format representation, a deterministic mapper based on regular expressions is applied to convert argument parses to the Interchange Format representation. This mapping process simply reformats the parse output without introducing any new information into the representation. In addition, SOUP allows for the definition of a set of very simple string mappings that are applied prior to parsing an input utterance. The string mappings are typically used for preprocessing tasks such as eliminating contractions and standardizing the form of compound words.

The SOUP parser supports a number of features that are useful for analysis of spoken language and for argument parsing. One important feature supported by SOUP is word skipping. This feature allows SOUP to parse meaningful portions of input utterances in the face of

spontaneous speech effects, ungrammatical inputs, and speech recognition errors. SOUP can skip words at any point in a parse, both between and within parse trees. The type and amount of skipping that SOUP is allowed to perform can be configured using several parameters. It is also possible to specify a list of words, such as *not*, that the parser is not allowed to skip.

Another feature of SOUP that is critical for argument parsing is the ability to produce analyses consisting of multiple parse trees. The goal of argument parsing is not to produce a single parse tree covering an entire utterance. Rather, the goal is to produce a sequence of phrase-level parses, possibly separated by sequences of unparsed words. Since SOUP segments input utterances into parse trees as part of the parsing process and can skip words between trees, it is ideally suited for this purpose.

The SOUP parser also provides support for modular grammar development ([Woszczyna *et al*., 1998]). Grammars designed for different domains or purposes can be developed and stored independently. Furthermore, a library of common grammar rules to be shared by all of the individual grammar modules can be defined. At runtime, SOUP constructs a single complete grammar from all of the grammar modules. All of the subgrammars are applied in parallel during parsing, and nodes in the resulting parse trees are marked with the grammar from which they originated. When an utterance is covered by multiple parse trees, the trees can be produced by a combination of rules from any of the subgrammars.

Finally, when an input utterance can be parsed in multiple ways, SOUP can provide a ranked list of interpretations. This situation may be more prevalent in the case of argument parsing as compared to full domain action parsing because the phrase-level grammar rules cannot use context from surrounding words and arguments for disambiguation. SOUP ranks interpretations based on a weighted combination of the heuristics shown in Figure 16.

1. Maximize the number of words covered by the parse.
2. Minimize the number of parse trees used to cover the utterance.
3. Minimize the number of wildcard matches used in the parse.
4. Minimize the number of nodes used in the parse trees.
5. Maximize the sum of the arc probabilities along parse tree paths in the parse.

Figure 16: Heuristics used by SOUP to rank alternative interpretations

Figure 16 lists the heuristics used by SOUP to rank interpretations in order of importance. The weights for each heuristic are set such that a more important heuristic completely overwhelms the influence of less important heuristics. Thus, for example, the number of parse trees would only affect the ranking of interpretations that covered the same number of words. Minimizing the number of wildcard matches (heuristic 3) is a heuristic that was supported by SOUP for use in previous applications, but wildcards were not used in the grammars developed for our hybrid analysis approach. Thus the heuristic has no effect on argument parsing. Additionally, the use of arc probabilities to select among alternative paths in the parse lattice requires the availability of training data annotated with parses. Since such data was not available, heuristic 5 also had no effect on argument parsing in our hybrid analyzer. Although the ordering of these heuristics was originally determined based on work with full domain action grammars, the remaining heuristics are applicable for argument parsing as well. Taken together, these heuristics essentially attempt to provide the simplest interpretation that most completely covers the input utterance. Maximizing the coverage of the argument parse is certainly a reasonable target, and we believe that the heuristics designed to prefer simpler parses are also applicable to the task of argument parsing.

2.3.2 Grammars

Section 1.5.1 described several speech-to-speech translation systems that were predecessors to the NESPOLE! system in which our analyzer was developed. In each of those systems, the principal approach for analyzing spoken utterances was grammar-based. The C-STAR II ([Levin *et al.*, 2000a]) and LingWear ([Fügen *et al.*, 2001]) translation systems, the most recent predecessors of NESPOLE!, used handwritten semantic grammars designed for parsing complete domain actions in conjunction with the robust SOUP parser in order to produce Interchange Format representations for input utterances. Semantic grammars rules focus on identifying semantic concepts in an utterance rather than syntactic structure. Although grammar development requires an intensive effort by expert grammar writers, semantic grammars can be developed in a relatively straightforward manner for restricted domains in which a well-defined set of semantic concepts can be identified. Furthermore, semantic grammars can be especially useful for parsing spoken language because they are less susceptible to problems with fragmented utterances and syntactic deviations ([Mayfield *et al.*, 1995a], [Mayfield *et al.*,

1995b]). We continue to use semantic grammars in our hybrid analysis approach. However, our analyzer uses phrase-level grammars for parsing arguments and other meaningful phrases rather than full domain-action-level grammars.

As mentioned previously, SOUP supports the definition of multiple subgrammars. For the purpose of argument parsing, four grammars are defined: an argument grammar, a pseudo-argument grammar, a cross-domain grammar, and a shared grammar. The grammars are separated based on their purpose rather than the domain they are intended to cover. Each of the grammars has a specific function that is useful for our hybrid analysis approach.

2.3.2.1 Argument Grammar

Utterance	*about one day later than me*

Argument Parse	`[arg:super_time=]::ARG (`
	`  [time=] (`
	`  [exactness=approximate] ( about )`
	`  [time-distance=] (`
	`  [det:quantity=] (`
	`  [det:main-quantity=] (`
	`  [no-oh:n-1-99=] ( one )`
	`  )`
	`  )`
	`  [nq:time-unit=] (`
	`  [day] ( day )`
	`  )`
	`  )`
	`  [time-relation=] (`
	`  [time-rel:after] ( later )`
	`  )`
	`  [reference-time=] (`
	`  [comp:unspecified] ( than )`
	`  [nested_compared-to=i] ( me )`
	`  )`
	`  )`
	`)`

Interchange Format	time=(exactness=approximate, reference-time=unspecified, time-relation=after, time-distance=(time-unit=day, quantity=1), compared-to=i)

Figure 17: Example of a *time*= argument parsed by the argument grammar

The argument grammar is designed to parse arguments defined in the Interchange Format. Top-level argument grammar rules correspond to top-level arguments defined in the Interchange format. The grammar contains rules for domain-independent arguments (i.e. *time*=, *location*=, *who*=) as well as domain-specific arguments (i.e. *attraction-spec*=, *trip-spec*=, *health-status*=, *medical-procedure-spec*=). Figure 17 shows an example of a time expression, an argument parse

tree for the expression produced by SOUP using the argument grammar, and the result of mapping the SOUP output to the Interchange Format representation. Each of the tokens in square brackets in the argument parse represents the left-hand side of a grammar rule.

2.3.2.2 Pseudo-Argument Grammar

The pseudo-argument grammar contains rules for parsing common phrases that can be grouped into meaningful classes. We call these phrases pseudo-arguments because they are not represented in the Interchange Format representation as real arguments. Rather, they are typically useful for identifying the speaker's intention and thus important in determining the domain action. For example, many possible phrases may indicate that a speaker is making a suggestion to the listener. In the Interchange Format, this intention might be represented by the domain action *suggest* or *suggest-action*. Figure 18 shows some of the phrases that may indicate that this is the speaker's intention. None of the phrases contribute any arguments to the Interchange Format representation, but each is a strong indication that the intention of the speaker is to make a suggestion and that the domain action for the semantic dialogue unit should thus be *suggest* or *suggest-action*. Each of the phrases would be parsed by the pseudo-argument `[=suggest=]`.

Utterance	*i suggest* *i would like to recommend* *have you considered* *how about* *it is usually a good idea to* *you would have to* *what you should maybe do is* *i would consider taking* *i should think you will enjoy*

Parse	`[=suggest=]::NTA (i suggest)`
	`[=suggest=]::NTA (i would like to recommend)`
	`[=suggest=]::NTA (have you considered)`
	`[=suggest=]::NTA (how about)`
	`[=suggest=]::NTA (it is usually a good idea to)`
	`[=suggest=]::NTA (you would have to)`
	`[=suggest=]::NTA (what you should maybe do is)`
	`[=suggest=]::NTA (i would consider taking)`
	`[=suggest=]::NTA (i should think you will enjoy)`

Figure 18: Examples of suggestion phrases parsed by the pseudo-argument grammar

In addition to simple phrases such as those in Figure 18, top-level pseudo-argument grammar rules also sometimes include real arguments on the right-hand side. For example, the grammar defines rules for a `[=send=]` pseudo-argument that parse phrases associated with the Interchange Format concept *+send*, which is used to represent the sending of information between the speaker and listener. The rules for `[=send=]` may include arguments for the recipient or for the method of transmission (i.e. e-mail, fax, etc.) in addition to simple phrases (i.e. *am sending*).

2.3.2.3 Cross-Domain Grammar

Utterance	*hello* *good bye* *nice to meet you* *thank you very much* *you've been a big help* *how may i help you*

Parse	`[greeting]::XDM ( [greeting=hello] ( hello ) )`
	`[greeting]::XDM ( [greeting=goodbye] ( good bye ) )`
	`[greeting]::XDM ( [greeting=first_meeting] ( nice to meet you ) )`
	
	`[thank]::XDM ( thank you very much )`
	`[thank]::XDM ( [statement:thank-reason=help] ( you have been a big help ) )`
	
	`[offer+help]::XDM ( how may i help you )`

Interchange Format	greeting (greeting=hello) greeting (greeting=goodbye) greeting (greeting=first_meeting) thank thank (thank-reason=help) offer+help

Figure 19: Examples of parses produced by the cross-domain grammar

The cross-domain grammar contains rules for parsing complete domain actions. The grammar is designed to cover utterances that occur across many domains. The utterances are often formulaic and typically contain few or no arguments. For example, the cross-domain grammar contains rules for the *greeting* domain action (*Hello*, *Good bye*, *Nice to meet you*, etc.) and the *thank* domain action (*Thanks*, *Thank you very much*, *You've been a big help*, etc.). Top-level rules in the cross-domain grammar correspond to complete domain actions rather than arguments or

classes of useful phrases. Thus, the Interchange Format representation for semantic dialogue units parsed by the cross-domain grammar can be produced directly from the parse without the need for domain action classification. Figure 19 shows examples of parses produced using the cross-domain grammar.

2.3.2.4 Shared Grammar

Finally, the shared grammar contains common grammar rules that can be used by any of the other grammars. None of the rules are top-level rules that can occur at the root of a parse tree. Rather, the shared grammar contains low-level rules that can be used in the right-hand sides of rules in other grammars. For example, many of the rules in the shared grammar are lexical-level rules that group together synonyms, pronouns, numbers, names, etc. The shared grammar also contains the rules for parsing simple and complex values of arguments.

In order to simplify the organization of the grammars developed for the hybrid analyzer, the argument, pseudo-argument, and cross-domain grammars contain only top-level rules (i.e. rules which can appear as the root of a parse tree). The shared grammar contains all of the lower-level rules. Thus, many of the rules that conceptually belong to the argument grammar are stored in the shared grammar in practice. This organization of the grammars offers advantages that simplify grammar maintenance. First, both the pseudo-argument and cross-domain grammars may contain rules that use arguments on their right-hand sides. By storing the lower-level argument rules in the shared grammar, the rules only have to be maintained in a single place rather than in multiple grammars. Additionally, although our implementation of the hybrid analyzer does not use multiple argument grammars for different domains, placing rules for parsing argument values and subarguments in the shared grammar would allow for easier migration to domain-specific argument grammars. For example, if separate argument grammars were defined for the Travel and Medical domains in NESPOLE!, subarguments and values defined in the shared grammar would be accessible by both the grammars for both domains.

2.4 Semantic Dialogue Unit Segmentation

The second stage of processing in our hybrid analysis approach is segmentation of input utterances into semantic dialogue units. As described in Section 1.5.4, domain actions in the Interchange Format interlingua are assigned at the level of semantic dialogue units. However,

humans do not necessarily speak at this level, and input utterances must therefore be split into semantic dialogue units before the domain actions can be determined.

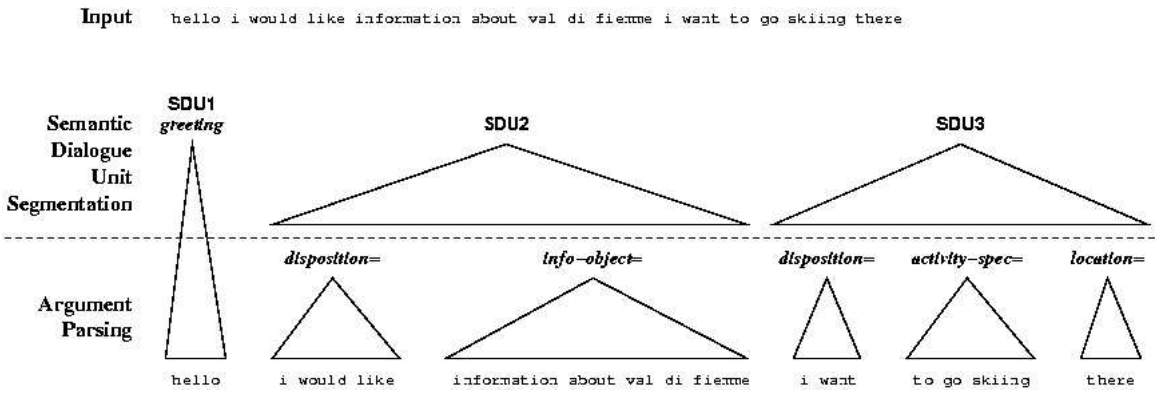

Figure 20: Semantic dialogue unit segmentation example

Figure 20 illustrates the results of applying semantic dialogue unit segmentation to the example utterance "*hello i would like information about val di fiemme i want to go skiing there*" shown in Figure 15. Three semantic dialogue units are identified in the utterance. The first contains "*hello*" and is covered by the cross-domain *greeting* tree. The second contains "*i would like information about val di fiemme*" and is covered by the first *disposition=* and *info-object=* argument trees. The third contains "*i want to go skiing there*" and is covered by the second *disposition=*, *activity-spec=*, and *location=* argument trees.

2.4.1 Using Argument Parse Information for Segmentation

When the hybrid analyzer is used to perform analysis in a speech-to-speech machine translation system, the input utterances will generally be the output of an automatic speech recognizer. Thus, neither punctuation nor case information will be available during semantic dialogue unit segmentation, and some input utterances may contain speech recognition errors. These factors may reduce the effectiveness of segmentation models that rely solely on properties of the words surrounding a potential boundary. However, because we perform segmentation after argument parsing in our hybrid analysis approach, information derived from the argument parse

may be included in the semantic dialogue unit segmentation model in addition to word-based information.

The argument parse provides several pieces of information that are useful for detecting semantic dialogue unit boundaries. First, the argument parse may contain trees produced by rules from the cross-domain grammar. Since these trees are rooted with a domain action and by definition cover a complete semantic dialogue unit, it is clear that there must be a semantic dialogue unit boundary immediately preceding and following a cross-domain tree. Therefore, the problem of segmenting an input utterance can be divided into subproblems of identifying boundaries in the portions of the utterance that are not covered by a cross-domain parse tree.

In addition to this "pre-segmentation" provided by the cross-domain trees, the argument parse further reduces the number of potential boundaries to be considered. Without the argument parse, the semantic dialogue unit segmenter would be required to test for a boundary between each pair of words in an utterance. Since argument parse trees represent units of semantic content, we make the assumption that no semantic dialogue unit boundaries occur in the middle of a parse tree. Thus, rather than considering the possibility of a semantic dialogue unit boundary between every pair of words, the segmentation model only needs to consider semantic dialogue unit boundaries at positions between argument parse trees and/or unparsed words.

Segmentation after argument parsing also allows for the root labels from the argument parse trees, which correspond to top-level arguments in the Interchange Format and pseudo-arguments useful for identifying parts of the domain action, to be included in the boundary detection model. Arguments and pseudo-arguments cluster words into semantic groups that are closely tied with the representation used in the Interchange Format and can thus be helpful for identifying semantic dialogue unit boundaries. Furthermore, the grammar labels define a fixed vocabulary that is much smaller (only a few hundred labels) than the set of words.

Finally, as mentioned previously, the Interchange Format specification includes rules that define how arguments are licensed by speech acts and concepts. In principle, it might be possible to use this knowledge during the segmentation process to identify sets of arguments that cannot occur together in the same semantic dialogue unit. However, using such knowledge directly is not necessarily feasible in practice. The Interchange Format specification defines tens of thousands of legal domain actions. Furthermore, the specification defines a sort of "catch-all" concept (called *+concept*) that is used for covering utterance fragments for which a more specific

concept or concept sequence cannot be determined. These factors mean that it is possible to find some legal domain action that licenses most sequences of arguments. Thus, there may be little to be gain from adding the extra processing that would be required to check the legality of all possible subsequences of arguments during segmentation. Of course, the value of such an approach could be determined experimentally, but we did not evaluate such an approach.

2.4.2 Detecting Semantic Dialogue Unit Boundaries

The semantic dialogue unit segmenter operates by making a binary decision about the presence or absence of a semantic dialogue unit boundary at each potential boundary position in the argument parse output. Since semantic dialogue unit boundaries are not allowed to occur within an argument parse tree, potential boundary positions can occur only between argument parse trees and/or unparsed words. In the example illustrated in Figure 20, there are no unparsed words, and potential boundary positions occur between each pair of trees in the argument parse.

At each potential boundary position, the segmenter first examines the parse elements, which may be either a parse tree or an unparsed word, surrounding the position. If either of the parse elements is a parse tree that was produced using a rule from the cross-domain grammar, a semantic dialogue unit boundary is inserted at the position. In Figure 20, the first potential boundary position (between the *greeting* and first *disposition*= trees) is an example of such a case. If neither parse element is a tree with a root label that came from the cross-domain grammar, an automatic classifier is used to determine whether or not a semantic dialogue unit boundary occurs at the position. All of the potential boundary positions in Figure 20 except the first are examples of this situation.

The data used to train the automatic segmentation classifier consists of manually transcribed utterances that have that have been annotated with semantic dialogue unit boundaries and parsed using the argument and pseudo-argument grammars. Each semantic dialogue unit is parsed separately so that argument parse trees are not allowed to span a true semantic dialogue unit boundary in the training data.

2.4.2.1 Unigram Threshold Model

Our first prototype classifier for semantic dialogue unit boundary detection used a simple statistical model similar to the one described in [Lavie *et al.*, 1997b] to estimate the likelihood of a semantic dialogue unit boundary at each position. The statistical model was based only on the

root labels of the parse trees immediately preceding and immediately following a potential boundary position. Unparsed words were dropped before segmentation, and only positions between argument parse trees were considered as possible semantic dialogue unit boundaries.

Suppose that a potential boundary position under consideration could be represented as $[A_1 \bullet A_2]$, where there was a potential boundary between argument parse trees A_1 and A_2. The likelihood that a semantic dialogue unit boundary occurred between A_1 and A_2 was estimated using the following formula:

$$F([A_1 \bullet A_2]) \approx \frac{C([A_1\bullet]) + C([\bullet A_2])}{C([A_1]) + C([A_2])}$$

The unigram counts $C([A_1\bullet])$, $C([\bullet A_2])$, $C([A_1])$, $C([A_2])$ were computed from the training data. $C([A_1\bullet])$ was the count of the number of times a that a semantic dialogue unit boundary followed an argument parse tree labeled A_1 in the data. $C([\bullet A_2])$ was the count of the number of times that a semantic dialogue unit boundary preceded an argument parse tree labeled A_2 in the data. $C([A_1])$ and $C([A_2])$ were the counts of the number of times argument parse trees labeled A_1 and A_2 occurred in the data training, regardless of boundary positions. A semantic dialogue unit boundary was inserted at a potential boundary position if the likelihood estimate exceeded a fixed threshold.

The primary weakness of this segmentation classifier was that it eliminated all word information. Unparsed words were dropped before segmentation and were no longer available for domain action classification in the next stage of processing. Of course, the unparsed words could have been ignored for segmentation purposes and included in either the preceding or following semantic dialogue unit when a boundary was detected. However, the decision of where to place the unparsed words by default would have been arbitrary. In addition to this poor treatment of unparsed words, determining a threshold that provided accurate segmentation results proved to be difficult, as described in Section 3.4.

2.4.2.2 Memory-Based Classifier

Because of the weaknesses in the prototype segmentation classifier, we created a memory-based (k-Nearest-Neighbor) segmentation classifier using TiMBL ([Daelemans *et al.*, 2002]).

The input to the TiMBL segmentation classifier consists of a set of 10 features based on the word and argument parse information surrounding a potential boundary position. The output of the classifier is a binary decision about the presence (+) or absence (-) of a semantic dialogue unit boundary at the position. Unlike the first segmentation classifier, this classifier can be applied at any potential boundary position, not just positions between argument parse trees. Table 3 lists the input features used by the TiMBL classifier. Features 1-4 and 7-10 are similar to the word and part of speech features used in [Stevenson and Gaizauskas, 2000] for sentence boundary detection.

Feature Position	Feature Name	Feature Description
1	A_{-1}	Root label of the argument parse tree immediately preceding the potential boundary position
2	$P(A_{-1}\bullet)$	Probability that a semantic dialogue unit boundary follows an argument parse tree with the preceding root label
3	w_{-1}	Word immediately preceding the potential boundary position
4	$P(w_{-1}\bullet)$	Probability that a semantic dialogue unit boundary follows the preceding word
5	*words_so_far*	Number of words since the previous boundary
6	*trees_so_far*	Number of argument parse trees since the previous boundary
7	A_1	Root label of the argument parse tree immediately following the potential boundary position
8	$P(\bullet A_1)$	Probability that a semantic dialogue unit boundary precedes an argument parse tree with the following root label
9	w_1	Word immediately following the potential boundary position
10	$P(\bullet w_1)$	Probability that a semantic dialogue unit boundary precedes the following word

Table 3: Input features for the TiMBL memory-based segmentation classifier

The values of input features 1 and 7 listed in Table 3 are simply the root labels of the argument parse trees immediately preceding (A_{-1}) and immediately following (A_1) the potential boundary position. If an unparsed word precedes or follows the position, then the root label feature is filled with a value indicating an unparsed word rather than a true root label. Similarly, the value of feature 3 is the word immediately preceding the potential boundary position (w_{-1}), and the value of feature 9 is the word that immediately follows the position (w_1).

In addition to the words and root labels surrounding a potential boundary position, the set of input features includes the probabilities that a boundary follows the preceding root label (feature 2, $P(A_{-1}\bullet)$) and word (feature 4, $P(w_{-1}\bullet)$) and the probabilities that a boundary precedes the following root label (feature 8, $P(\bullet A_1)$) and word (feature 10, $P(\bullet w_1)$). Figure 21 shows the formulas used to estimate these probabilities based on counts from the training data.

$$P(A_{-1}\bullet) = C(A_{-1}\bullet)/C(A_{-1})$$
$$P(w_{-1}\bullet) = C(w_{-1}\bullet)/C(w_{-1})$$
$$P(\bullet A_1) = C(\bullet A_1)/C(A_1)$$
$$P(\bullet w_1) = C(\bullet w_1)/C(w_1)$$

Figure 21: Formulas for estimating probabilities used in the TiMBL segmentation classifier

The two remaining features in the input feature set are based on the length of the current semantic dialogue unit. Feature 5 is a count of the number of words seen since the previous boundary, and Feature 6 is a count of the number of parse trees seen since the previous boundary.

A training example for the TiMBL memory-based segmentation classifier is created for each potential boundary position in the parsed training data. Positive segmentation examples occur between semantic dialogue units, as marked in the data. For example, the utterance shown in Figure 20 contains two positive examples: between the words "*hello*" and "*i*" and between the words "*fiemme*" and "*i*". Negative segmentation examples are created for each potential boundary position within a semantic dialogue unit. Three negative examples are created for the utterance in Figure 20: between the words "*like*" and "*information*", between the words "*want*" and "*to*", and between the words "*skiing*" and "*there*". Two additional positive training examples can also be created by using the partial information available at the beginning and end of the utterance. The information is partial in the sense that, at the beginning of the utterance, only the features containing information about the following word and root label (features 7-10) can be filled in with true values. Likewise, at the end of the utterance, only the features containing information preceding the boundary position (features 1-6) can be filled with true values. The

values for the remaining features are filled with default values. Creating such examples allows the instance base used in the segmentation classifier to include information from the turn boundaries that would otherwise be lost.

2.5 Domain Action Classification

Following argument parsing and semantic dialogue unit segmentation, the third stage of processing in the hybrid analysis approach is domain action classification. The purpose of this stage is to identify the domain action for each semantic dialogue unit in an input utterance. Some of the semantic dialogue units in an utterance may be parsed by the cross-domain grammar during argument parsing. In that case, the root label of the parse tree that covers the semantic dialogue unit identifies the domain action. Otherwise, the task of identifying the domain action is accomplished using automatic classification techniques. The Interchange Format specification is then used to ensure that a legal Interchange Format representation is produced for the semantic dialogue unit.

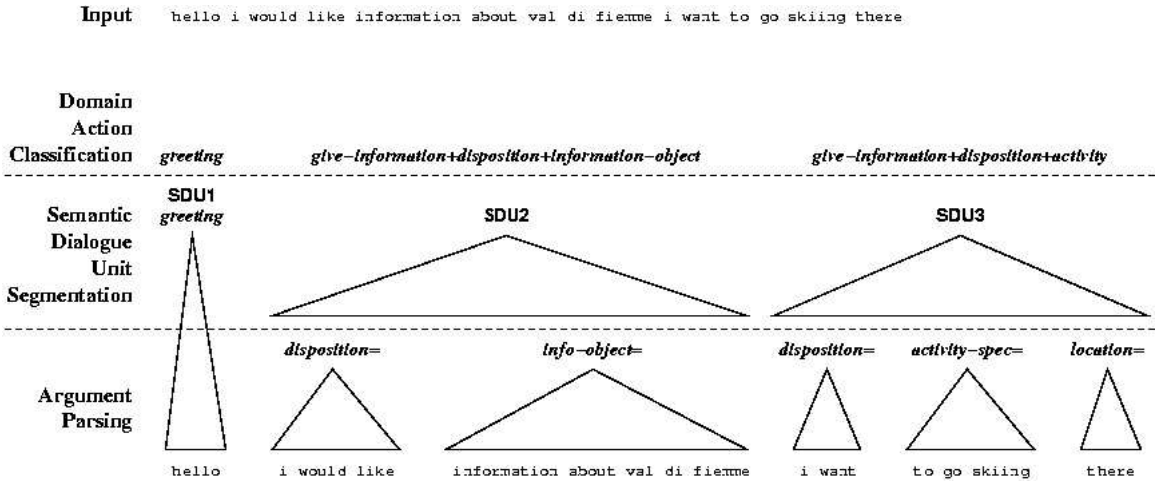

Figure 22: Domain action classification example

Figure 22 shows the results of domain action classification applied to the example from Figure 20. The domain action *greeting* for the first semantic dialogue unit is extracted from the cross-domain tree produced during the argument parsing stage. The domain actions for the

second and third semantic dialogue units are identified using automatic domain action classification. The domain action assigned to the second semantic dialogue unit is *give-information+disposition+information-object*, and the domain action for the third semantic dialogue unit is *give-information+disposition+activity*.

Several important design alternatives must be considered in constructing the domain action classification module for our hybrid analysis approach. The design alternatives examined in this dissertation include the definition of the domain action classification task, the set of input features used by the domain action classifier(s), the machine learning approach used to implement the domain action classifier(s), and the use of the Interchange Format specification to supplement domain action classification. The decisions made regarding these alternatives affect the usability of the domain action classification module in the context of a human-to-human speech-to-speech machine translation system where the goal is (near) real-time translation. Domain action classification is only one of many steps required to produce the translation for an input utterance. Thus, the domain action classification step must perform well enough to produce useful translations, but it must not be so complex that it slows down the translation of the utterance and thus the dialogue. The effects of these design considerations on usability and performance are discussed in the following sections. Further discussion and evaluation is included in Chapter 4.

2.5.1 Defining the Classification Task

There are several possible definitions for the general task of domain action classification that differ in how they separate the domain action into components. A single classifier may be used to produce a complete domain action, or multiple classifiers may be used to identify components from which a complete domain action can be assembled. The main tradeoff among the possible definitions is the difficulty of the decision that the automatic classifier(s) must make versus the complexity of extracting the domain action from the automatic classification output. The extent to which domain actions that were not observed in the training data can be produced during domain action classification also varies depending on the task definition. Each classifier is only able to output the classes that were seen in the training data. If the domain action components are classified separately, domain actions that were not seen in the training data could be produced when the components are combined.

The most straightforward approach to the domain action classification task is to train a single classifier to identify a complete domain action. Given a semantic dialogue unit as input, the task of the classifier is to output the best domain action. Assuming that the domain actions found in the training data are legal, it is impossible for the classifier to produce an ill-formed domain action. Thus, no further processing is required to ensure that the domain action is well formed according to the Interchange Format specification. This was the approach taken in [Cattoni *et al.*, 2001]. Classifying the complete domain action in one step clearly provides the least complex scenario for extracting the domain action from the classification results. After a single classification decision is made, no additional processing is required to verify the legality of the domain action. However, this approach also presents the automatic classifier with the most difficult classification decision. The number of classes from which the classifier must choose is equal to the number of domain actions in the training data, which can be on the order of 1000 classes. This approach also provides the least flexibility with respect to generalization beyond the training data. The complete domain action classifier is not capable of producing a domain action that was not seen in the training data.

It is also possible to separate the task of identifying the domain action into two subtasks of classifying the speech act and the concept sequence separately. One classifier is trained to identify the speech act, and a second classifier is trained to identify the complete concept sequence. This second approach creates only a slight increase in the complexity of extracting the domain action from the classification results. The outputs of the two classifiers are simply concatenated to produce the complete domain action. Since the concept sequence is guaranteed to be well-formed, only a simple test is required to check that the concept sequence is allowed to follow the speech act. This approach also reduces the difficulty of the decisions the classifiers must make. The speech act classifier must select from approximately 75 classes, and the concept sequence classifier must select from several hundred classes. Although these are still very difficult classification tasks, they are easier than classifying the complete domain action. This approach also makes it possible to produce domain actions that were not seen in the training data. Although no new concept sequences can be produced, speech acts and concept sequences can be combined in new ways. An additional benefit of breaking up the task of domain action classification into separate subtasks is that different classification approaches and/or feature sets can be applied to each subtask.

A third approach to the domain action classification task would be to further break down the task of identifying the concept sequence into subtasks of classifying individual concepts. As in the previous approach, a single classifier would be trained to classify the speech act. Then, a set of classifiers would be trained to identify the individual concepts. Each concept classifier would output a binary decision about whether or not the concept should be included in the domain action for the semantic dialogue unit. This approach would have the benefit that the individual concept classifiers would only have to make a simple binary decision. Furthermore, any concept sequence allowed by the Interchange Format specification could in principle be produced regardless of whether or not it appeared in the training data. However, this approach would create the need for a very complex process for extracting the domain action from the classifier outputs. Some mechanism for composing legal concept sequences from the set of concepts identified would be required. The resulting concept sequences would then have to be ranked and the best one selected. A complete domain action would be formed by combining the output of the speech act classifier with the best concept sequence assembled from the set of concepts identified for the semantic dialogue unit. The Interchange Format would be used to verify that the assembled concept sequence and the complete domain action were legal.

Based on the relative complexity of extracting the domain action from the classifier outputs, only the first two approaches, classifying the complete domain action or classifying the speech act and complete concept sequence, appear to be practical for use in an online domain action classification module. The third approach requires the application of as many as 144 concept classifiers (based on the final Interchange Format specification for the NESPOLE! project) to each semantic dialogue unit in an input utterance versus the 1 or 2 classifiers required for the first two approaches. Depending on the learning approach used to implement the classifier, the time required for running that many classifiers could be prohibitive for an online system. The process required for assembling the domain action in the third approach is also much more complex than that required for the first two approaches and may also be a prohibitive factor. The complete domain action classifier requires no additional assembly, and a simple table look-up can be used to verify that a speech act and complete concept sequence can be legally combined. On the other hand, a potentially very large number of possible concept sequences would have to be examined, tested for legality, and ranked using the third approach. Even if each of the individual concept classifiers were relatively accurate, compounding of errors from the

classifiers and errors in ranking the possible concept sequences would likely eliminate any performance advantages that might be gained from reducing the difficulty of the classification task. Furthermore, although there may be a canonical domain action for a given semantic dialogue unit, it is often the case that alternative domain actions exist which can adequately convey the speaker's intention. Assuming that the training data provides adequate coverage of the domain, it is likely that the first two approaches would be able to find an acceptable, if not perfect, domain action to convey the intended meaning. Thus, the greatly increased complexity of the individual concept classification approach outweighs its ability to produce any possible legal domain action.

2.5.2 Input Features for Classification

The set of input features to be used by the domain action classifier(s) is another important design consideration. Input features for the classifiers used in domain action classification may come from a variety of sources. The input features used should be easy to extract from the semantic dialogue unit to be classified so that feature extraction does not delay the dialogue in the online translation system. Furthermore, the features should be easy for humans to tag in the training data or easy to automatically create from the data since part of the motivation for developing the hybrid analysis approach was to reduce human effort required for system development. Possible sources for features that meet these requirements include the argument parse, the words in the semantic dialogue unit, the Interchange Format representation of the arguments in the argument parse, and other properties of the dialogue (i.e. speaker side).

1. Argument grammar root labels
2. Pseudo-argument grammar root labels
3. Words
4. Top-level Interchange Format arguments
5. Speech act
6. Speaker side
7. Speech act probabilities
8. Individual concept probabilities

Figure 23: Types of input features used for domain action classification

Figure 23 lists the types of possible input features for domain action classification that were explored. These features can be extracted directly or produced automatically from a tagged semantic dialogue unit without the need for any extra annotation effort beyond what is already done. For the work described in this dissertation, each set of features of a particular type was either used in its entirety or not used at all.

One of the primary sources of features for domain action classification is the argument parse. The argument and pseudo-argument trees produced during argument parsing group words or phrases into useful semantic categories that are closely tied to the Interchange Format representation and the domain action. The first two types of features are comprised of the root labels of parse trees produced by the argument grammar and pseudo-argument grammar respectively. The specific features included in each type are defined by the set of top-level rules in each grammar. For each feature type, a feature vector containing one binary feature for each top-level rule is defined. The binary feature for a particular rule indicates the presence (+) or absence (-) of the corresponding root label in the argument parse for a semantic dialogue unit. During training, the features can be easily produced automatically by parsing the training data with the argument and pseudo-argument grammars. During online classification, the features can be extracted directly from the output of the argument parsing stage.

The words in the original input utterance are another obvious source of input features. Although the argument parse provides one abstraction of the words in a semantic dialogue unit, the words themselves may provide additional useful information for domain action classification. The words in a semantic dialogue unit are represented using a binary feature vector similar to the one used for the argument and pseudo-argument grammar labels. Each feature indicates the presence or absence of the associated word in the semantic dialogue unit. In our hybrid analyzer, the set of words to be included in the feature vector is determined by sorting all of the words in the training data according to their average mutual information with the class (i.e. speech act, concept sequence, etc.). The top *N* words with the highest mutual information are included. Determination of the words included in the feature vector is described further in Chapter 4.

As mentioned previously, the raw output of the argument parser does not match the format of the Interchange Format representation. Thus, a mapper is applied to convert the argument parse representation into a legal Interchange Format representation. The top-level arguments in the Interchange Format representation define another type of possible input features for domain

action classification. Although the information from these features might be partially redundant with the information from the grammar label features, it may still provide useful information. In some cases, the argument grammar may contain multiple top-level rules that correspond to the same Interchange Format argument. In such cases, the Interchange Format representation of the argument would condense all of the rules into a single feature. Furthermore, top-level rules in the pseudo-argument grammar may occasionally contain top-level arguments as children. Such arguments are lost in the grammar label feature set but would appear in the Interchange Format representation. The top-level Interchange Format arguments are represented using a binary feature vector similar to the grammar label feature vectors. The Interchange Format representation of the arguments is extracted during training and online analysis by applying the mapper to the argument parse output.

The speaker side information for the input utterance can also be used as an input feature for domain action classification. The speaker side is represented as a single feature whose value is the speaker side label. This feature may be useful if the domain actions typically used by the customer and the agent differ. Semantic dialogue units in the training data are annotated with the speaker side, and the speaker side is determined by the role of the speaker in the dialogue.

When the domain action classification task is split and the speech act is classified separately from the concept sequence, the speech act may also be included as an input feature for the concept sequence classifier. The speech act is represented as a single feature whose value is the speech act assigned to the semantic dialogue unit. During training, the speech act feature is extracted from the domain action tag for a semantic dialogue unit. In online mode, the best speech act determined by the speech act classifier supplies the feature value for the concept sequence classifier. Training on the tagged speech acts and testing on the classifier output creates a mismatch between training and testing conditions for concept sequence classification. It could be determined experimentally if training on the "noisy" speech acts produced by the classifier would provide better concept sequence classification performance, but we did not conduct such an experiment.

The probabilities of each speech act and individual concept given the content of a semantic dialogue unit may also be used as input features for domain action classification. As described in Chapter 4, we evaluated the use of naïve Bayes n-gram models to estimate speech act and concept probabilities. A model is trained for each speech act and concept found in the training

data based on n-grams of the words or arguments in a semantic dialogue unit. During domain action classification, the n-gram models are used to compute the probability of each possible speech act and concept based on the semantic dialogue unit being processed. These probabilities are then represented as feature vectors with one feature for each possible speech act or concept. The numerical value of each feature is the probability of the associated speech act or concept computed by the naïve Bayes models.

2.5.3 Classification Approaches

Another important design decision to be made in developing the hybrid analyzer is what type of classifier(s) to use for performing domain action classification. Certainly one of the most important considerations is how well the machine learning approach performs the required classification task. If there is one learning technique that is clearly more accurate than any other, then the choice may be relatively simple. However, a number of additional factors may influence the suitability of a particular technique for use in domain action classification.

One important attribute of the learning approach is the speed of both classification and training. Since domain action classification is one part of a translation system designed for use between two humans to facilitate (near) real-time communication, the classifiers used must be able to classify individual utterances online very quickly. In addition to online classification using a trained model, the ease and speed of training a classifier and batch-mode classification are important factors. Human grammar writers must develop and test the argument grammars. In order to see the effects of grammar modifications on analysis quality, the grammar writers must use the full hybrid analyzer. Thus, classifier training and batch classification should be fast so that the grammar writers can update the grammars, retrain the classifiers, and test efficiently.

The machine learning approach should also be able to easily accommodate both continuous and discrete features from a variety of sources. As described in the previous section, input features for domain action classification may be derived from words and/or phrases in an utterance, the argument parse, the Interchange Format representation of the arguments, and properties of the dialogue (e.g. speaker tag). The feature values may be discrete (binary or multi-valued) or continuous. The learning technique used for domain action classification should be able to easily combine any or all of these feature types and sources.

Another desirable attribute for the learning approach used in domain action classification is the ability to produce a ranked list of possible classes. The Interchange Format specification defines how speech acts and concepts are allowed to combine as well as how arguments are licensed by the domain action. It may sometimes be the case that the combination of the best classified domain action with the arguments from argument parsing results in an illegal Interchange Format representation. In such cases, constraints from the Interchange Format specification can be used to help select an alternative domain action.

The ability of the learning approach to cope with many classes and sparse data is another important factor for domain action classification. The classifier(s) may have to deal with hundreds, or potentially thousands, of output classes. Furthermore, the training data for the classes may be sparse and unevenly distributed. Although a few of the most common classes may have hundreds or thousands of examples in the training data, many of the classes occur only 1 or 2 times. For example, about two thirds of the domain actions and concept sequences and about one third of the speech acts found in the NESPOLE! English Travel & Tourism training database occur only once or twice. To illustrate, Figure 24 and Figure 25 show the distributions of the domain actions, speech acts, and concept sequences in the NESPOLE! English Travel & Tourism training database. Figure 24 displays the distributions for only the 15 most frequent domain actions, speech acts, and concept sequences. The counts for the top 5 speech acts as well as the 15^{th} most frequent speech act are also shown in the graph. Figure 25 picks up where Figure 24 leaves off, showing the distributions for all of the remaining domain actions, speech acts, and concept sequences below the top 15. As the graphs show, the distributions have a few classes with many examples and long tails of classes with few examples.

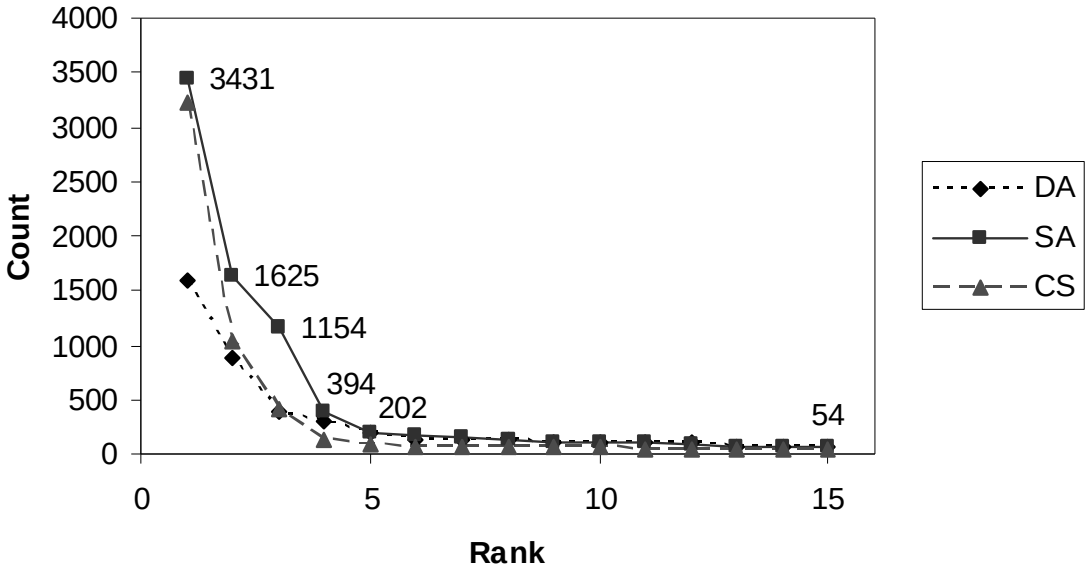

Figure 24: Distribution of the top 15 domain actions, speech acts, and concept sequences in the NESPOLE! English Travel & Tourism training data

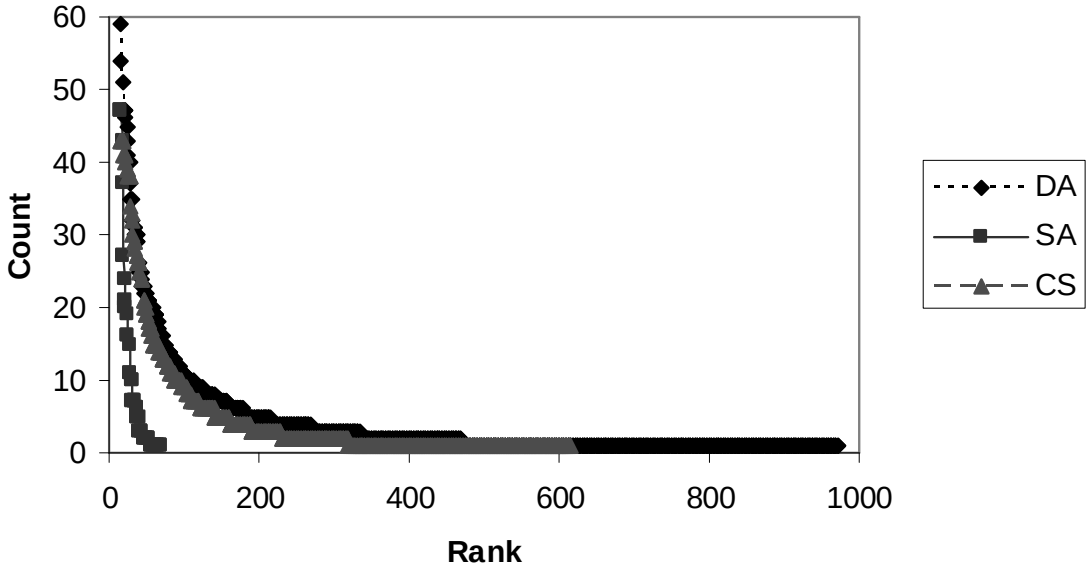

Figure 25: Distribution of domain actions, speech acts, and concept sequences in the NESPOLE! English Travel & Tourism training data (excluding top 15)

Chapter 4 includes an evaluation of the performance of four different machine-learning techniques on domain action classification: memory-based learning (k-Nearest-Neighbor), decision trees, neural networks, and naïve Bayes n-gram classifiers. We selected these approaches because they vary substantially in way the training data is processed and represented and in their methods for determining the best class. Since the focus of this dissertation was not to implement machine learning techniques from scratch but to test their effectiveness for domain action classification, existing software was used "off the shelf" for each learning approach. The ease of acquiring and setting up the software influenced our choices, as did the ease of incorporating the software into our online translation system. Table 4 lists the software packages used for each learning technique.

Machine Learning Technique	Software Implementation	Reference
Memory-Based Learning	TiMBL	[Daelemans *et al.*, 2002]
Decision Trees	C4.5	[Quinlan, 1993]
Neural Networks	SNNS	[Zell *et al.*, 1998]
Naïve Bayes n-Gram Models	Rainbow	[McCallum, 1996]

Table 4: Learning techniques used for domain action classification

2.5.4 Using Interchange Format Constraints

The Interchange Format specification imposes constraints on the ways in which various elements of the Interchange Format representation can be legally combined. Including knowledge from the Interchange Format specification in domain action classification provides two advantages over otherwise naïve classification. First, the hybrid analyzer must produce valid Interchange Format representations in order to be a useful component for our machine translation systems. By using the Interchange Format specification, the hybrid analyzer can guarantee that only legal Interchange Format representations are produced. Second, when an illegal Interchange Format representation is detected, knowledge from the specification can be used to help select a legal alternative representation.

Two elements of the Interchange Format specification are especially relevant to domain action classification. First, the specification defines constraints on the composition of domain actions. These constraints define how concepts are allowed to pair with speech acts and ordering constraints on how concepts are allowed to combine in a concept sequence. For example, the concept *+disposition* is allowed to follow the speech act *give-information*, but the concept *+activity* is not. Such constraints are used to verify that a domain action is legal. The second important element of the Interchange Format specification that is important for domain action classification is the definition of how arguments are licensed by the domain action. In order for an Interchange Format representation to be valid, each of the top-level arguments must be licensed by at least one speech act or concept in the domain action. For example, in the third semantic dialogue unit in Figure 22, the *disposition=* argument is licensed by the *+disposition* concept, the *activity-spec=* argument is licensed by the *+activity* concept, and the *location=* argument is licensed by both the *+activity* argument and the *give-information* speech act. Such argument licensing constraints are used to verify that the domain action licenses all of the top-level arguments. When an illegal Interchange Format representation is produced, both sets of constraints are used to help select an alternative legal representation for a semantic dialogue unit.

The hybrid analyzer uses these elements of the Interchange Format specification in a fallback strategy to supplement domain action classification and to guarantee that a legal Interchange Format representation is produced for each semantic dialogue unit. Suppose that domain action classification is performed using a speech act classifier and a concept sequence classifier. When the combination of the best speech-act and the best concept sequence results in an illegal domain action, the analyzer attempts to find an alternative legal domain action. The fallback strategy requires a ranked list of alternative speech acts and concept sequences. The list can be based on the output of the classifiers when possible or on the frequencies of the classes in the training data. Each of the alternative concept sequences (in ranked order) is combined with each of the alternative speech acts (in ranked order). The analyzer first tests if each alternative domain action is valid. For each legal domain action, the analyzer tests if all of the arguments found during argument parsing are licensed. If a legal domain action is found that licenses all of the arguments, the fallback process stops and the resulting Interchange Format representation is returned. If the analyzer is unable to find a legal domain action that licenses all of the arguments, the highest-ranked domain action (i.e. the first one found during the fallback search) that licenses

the most arguments is selected. In this case, any arguments that are not licensed by the selected domain action are removed from the Interchange Format representation since illegal arguments may cause a generation failure.

As mentioned previously, although there may be a canonical domain action for a semantic dialogue unit, there are often alternative domain actions that can adequately convey the speaker's intended meaning. On the other hand, the arguments in the Interchange Format representation represent specific detailed information from the speaker's utterance. If arguments are dropped from the Interchange Format representation, key pieces of meaning will often be lost. This fallback strategy takes advantage of the flexibility at the domain action level and selects a lower-ranking domain action in order to retain as many arguments as possible.

2.6 Online Implementation

The hybrid analysis approach described in this chapter is fully incorporated into the NESPOLE! speech-to-speech machine translation system described in Section 1.5.2 ([Lavie *et al.*, 2002], [Metze *et al.*, 2002], [Guerzoni *et al.*, 2003]). The online hybrid analyzer illustrated in Figure 26 fills the role of the *Text Analysis Module* in the NESPOLE! English and German translation servers as depicted in Figure 5. The online hybrid analyzer is comprised of six modules. The modules communicate with each other and with the rest of the translation server via sockets. Although the modules generally all run on the same machine, the architecture of the online hybrid analyzer also supports running each module on a separate machine with communication over the Internet.

The *Hybrid Analysis Manager* is a Perl script that manages all of the communication among the components and the communication with the translation server. When the translation server receives an utterance to translate, the text of the utterance (whether produced by automatic speech recognition or typed by the user) is passed to the hybrid analyzer. The *Hybrid Analysis Manager* receives the text and begins the analysis process. The text of the utterance is first passed to the *Argument Parser*. As described in Section 2.3, argument parsing is performed using the SOUP parser. The best argument parse produced by SOUP is passed back to the *Hybrid Analysis Manager*.

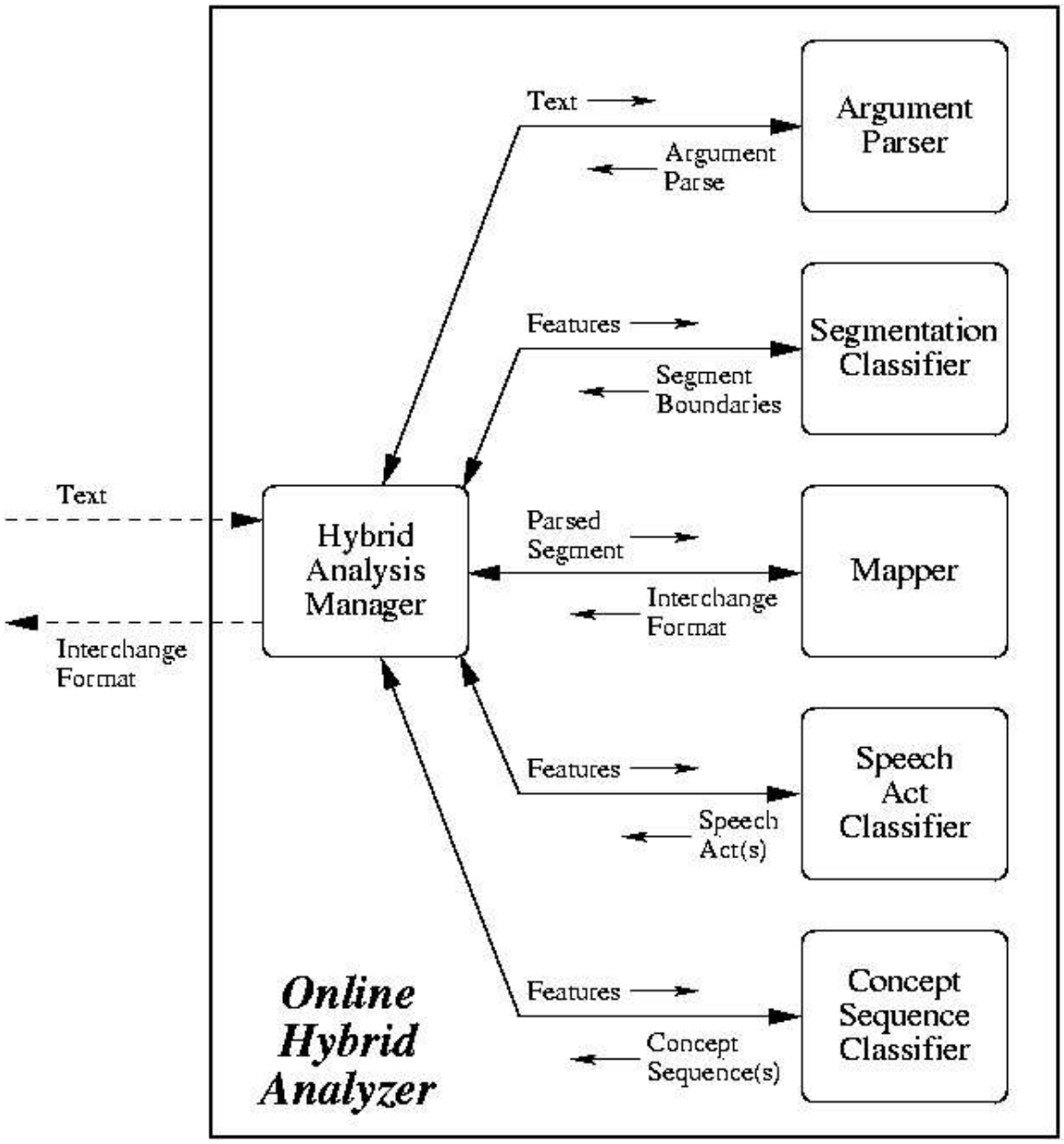

Figure 26: Online hybrid analyzer used in the NESPOLE! translation system

The *Hybrid Analysis Manager* next identifies the potential semantic dialogue unit boundary positions. For each potential boundary position that is not bordered by a cross-domain parse tree, the *Hybrid Analysis Manager* extracts the input features listed in Table 3 and sends the feature vector to the *Segmentation Classifier*. Classification of the potential boundary position is performed using the memory-based classifier described in Section 2.4.2.2 using the TiMBL software package, which is implemented in C++. The *Segmentation Classifier*

determines if there is a semantic dialogue unit boundary at the position and returns yes (+) or no (-) to the *Hybrid Analysis Manager*.

The argument parse for each semantic dialogue unit is then passed to the *Mapper*. The *Mapper* is a Perl script that converts the SOUP parse output to the Interchange Format representation. The Interchange Format representation is then passed back to the *Hybrid Analysis Manager*. If a semantic dialogue unit was parsed by the cross-domain grammar, the output of the *Mapper* consists of a complete Interchange Format representation, and no further processing is necessary for that semantic dialogue unit. Otherwise, the *Mapper* output contains only the representation of the Interchange Format arguments found in the semantic dialogue unit, and domain action classification is performed.

The online hybrid analyzer used in the NESPOLE! system performs domain action classification using the second approach described in Section 2.5.1 (classifying the speech act and complete concept sequence separately). The *Speech Act Classifier* and *Concept Sequence Classifier* are both implemented using a TiMBL memory-based classifier. The *Hybrid Analysis Manager* extracts the input features used for speech act classification and passes the feature vector to the *Speech Act Classifier*, which passes back the best speech act. The *Speech Act Classifier* may also return a small set of alternative speech acts if the nearest neighbor set contains more than one possible speech act for the semantic dialogue unit. The *Hybrid Analysis Manager* then creates the input feature vector for concept sequence classification and passes the features to the *Concept Sequence Classifier*, which passes back the best concept sequence and a ranked set of alternatives.

Following domain action classification, the *Hybrid Analysis Manager* checks the legality of the Interchange Format representation and applies the fallback strategy if necessary. If any alternative speech acts or concept sequences were returned by the classifiers, those are tested first during fallback followed by the speech acts and/or concept sequences from the training data sorted by frequency of occurrence. After each semantic dialogue unit has been processed, the *Hybrid Analysis Manager* passes the resulting Interchange Format representations for the utterance back to the translation server.

Incoming Utterance	Processing: HYPO ENG: 1056139880 { hello i would like information about val di fiemme I want to go skiing there } -src _clangley -side c
Argument Parsing	SOUP Output: ; Parsing utt 35 (line 35) ; Mapped Utt: " <s> hello i would like information about val di fiemme i want to go skiing there </s> " ; Interpretation 35.1 ; Parsed Utt: !<s> hello i would like information about val di fiemme i want to go skiing there !</s> ; Score: 14393.7 ; Coverage: 100% (15/15) in 6 trees {1 2} [greeting]::XDM ([greeting=hello] (hello)) {2 5} [arg:disposition=]::ARG ([super_who=] ([who=] ([who:i] (i))) [desire] (would like)) {5 10} [arg:super_info-object=]::ARG ([info-object=] ([information] (information) about [object-topic=] ([name-val_di_fiemme_area] (val di fiemme)))) {10 12} [arg:disposition=]::ARG ([super_who=] ([who=] ([who:i] (i))) [desire] (want)) {12 15} [arg:super_activity-spec=]::ARG ([activity-spec=] ([skiing] (to go skiing))) {15 16} [arg:super_location=]::ARG ([location=] ([there] (there)))
Segmentation	Segmentation (with Timbl): + [greeting]::XDM ([greeting=hello] (hello)) + [arg:disposition=]::ARG ([super_who=] ([who=] ([who:i] (i))) [desire] (would like)) classify [arg:disposition=]::ARG,0.12099644,like,0.10000000, 3,1,[arg:super_info-object=]::ARG,0.12195122,information,0.03409091,?. - [arg:super_info-object=]::ARG ([info-object=] ([information] (information) about [object-topic=] ([name-val_di_fiemme_area] (val di fiemme)))) classify [arg:super_info-object=]::ARG,0.54101996,fiemme,0.75675676, 8,2,[arg:disposition=]::ARG,0.56761566,i,0.59634551,?. + [arg:disposition=]::ARG ([super_who=] ([who=] ([who:i] (i))) [desire] (want)) classify [arg:disposition=]::ARG,0.12099644,want,0.13868613, 2,1,[arg:super_activity-spec=]::ARG,0.07909605,to,0.02421796,?. - [arg:super_activity-spec=]::ARG ([activity-spec=] ([skiing] (to go skiing))) classify [arg:super_activity-spec=]::ARG,0.34463277,skiing,0.42682927, 5,2,[arg:super_location=]::ARG,0.12053115,there,0.23233696,?. - [arg:super_location=]::ARG ([location=] ([there] (there)))
Mapping and Domain Action Classification (SDU2)	Arguments: disposition=(who=i, desire), info-object=(information, object-topic=name-val_di_fiemme_area) SA = CATEGORY {give-information} DISTRIBUTION { request-information 2 } DISTANCE {8.311586} CONCEPTS = CATEGORY {+disposition+information-object} DISTRIBUTION { +disposition+obtain+information-object 1 } DISTANCE {4.614288}

Mapping and Domain Action Classification (SDU3)	Arguments: disposition=(who=i, desire), activity-spec=skiing, location=there SA = CATEGORY {give-information} DISTRIBUTION { request-information 4, acknowledge 1, end-discussion 2 } DISTANCE {10.323833} CONCEPTS = CATEGORY {+disposition+action} DISTRIBUTION { +disposition+activity 6, +disposition+tour 1, +disposition+action 4 } DISTANCE {7.372863} CHECKING DA'S FROM CLASSIFICATION
Final Output	IF ENG: 1056139880 { {c:greeting (greeting=hello)} {c:give-information+disposition+information-object (disposition=(who=i, desire), info-object=(information, object-topic=name-val_di_fiemme_area))} {c:give-information+disposition+activity (disposition=(who=i, desire), activity-spec=skiing, location=there)} } -src _clangley -side c

Figure 27: Online hybrid analysis example

Figure 27 shows a log of the intermediate outputs produced by the hybrid analysis modules during analysis of the utterance shown in Figure 15. The processing of the utterance shown in Figure 27 contains an example of how the Interchange Format specification can be used to improve the initial results of domain action classification. The line "`CHECKING DA'S FROM CLASSIFICATION`" in the *Domain Action Classification (SDU3)* output indicates that the best Interchange Format representation for the semantic dialogue unit was illegal. The best concept sequence returned by concept sequence classification was *+disposition+action* which does not license the *activity-spec=* argument found during argument parsing. Thus, the fallback strategy first checks the alternative concept sequences returned by the classifier (listed in the "`DISTRIBUTION`" line). The concept sequence *+disposition+activity*, which is the correct concept sequence, is found among the alternatives. Because the resulting domain action, *give-information+disposition+activity*, licenses all of the arguments fallback processing stops there.

Chapter 3 Evaluation of Semantic Dialogue Unit Segmentation

The semantic dialogue unit segmentation classifier used in the online version of the hybrid analyzer that is included in the NESPOLE! translation servers is a memory-based classifier implemented using TiMBL Version 4.3 ([Daelemans *et al.*, 2002]). The goal of the segmentation classifier is to identify semantic dialogue unit boundaries in an input utterance. As described in Chapter 2, potential boundaries occur between argument parse trees and/or unparsed words. The segmentation classifier must decide whether or not a semantic dialogue unit boundary occurs at each potential boundary position. The ability of the classifier to accurately identify both positive instances (i.e. semantic dialogue unit boundaries) and negative instances (i.e. non-boundaries) is important for the purposes of our interlingua-based machine translation system. Oversegmentation of input utterances would lead to many semantic dialogue units containing very few words and arguments. The resulting translations might capture many of the specific details from the input utterance because each individual argument could be included in its own phrase. However, the utterance and arguments could be too broken up to identify the speaker's intention regarding the details, essentially leaving a list of details with no information about what to do with them. On the other hand, undersegmentation would produce very few semantic dialogue units with many arguments in each one. This situation would increase the likelihood of creating a set of arguments that were incompatible in the same semantic dialogue unit. Some arguments might have to be dropped and, thus, details would be lost from the translation. Even if none of the arguments were dropped, it would often be the case that only a very general domain action that lost most or all of the speaker's intention would license all of the arguments. The resulting translation would then be similar to the output in the oversegmentation case. In this chapter, we present the results of several experiments designed to test the effectiveness of the semantic dialogue unit segmentation classifier. We evaluated segmentation classification performance on English and German input in the NESPOLE! Travel & Tourism and Medical Assistance domains.

3.1 Preparation of Training Data

For each language-domain pair, the corpus used in all of the segmentation classification experiments was the final version of the NESPOLE! database. The NESPOLE! database contains dialogues that were manually transcribed and annotated with speaker turns and semantic dialogue unit boundaries. Some of the dialogues in the databases were annotated only with semantic dialogue unit boundaries and not with Interchange Format representations. However, since the Interchange Format representations are not necessary for segmentation, such dialogues may still be used for training and testing the segmentation classifiers.

The training data for the segmentation classifier was created by first extracting all semantic dialogue units for the source language from the database. All punctuation and case information was removed so that the text for each semantic dialogue unit was essentially what an automatic speech recognizer would produce if there were no recognition errors. Each semantic dialogue unit was then parsed for arguments using the argument and pseudo-argument grammars. In order to prevent the argument parser from parsing across a boundary during training, semantic dialogue units were parsed individually rather than as part of a full turn for the purpose of training the segmentation classifier. In this sense, the data is "clean" with respect to parsing errors since no true boundary could be parsed under an argument. Of course, the training data could still have included parse errors in which incorrect arguments were included in the parse. Additionally, the segmentation classifier was trained to identify all semantic dialogue unit boundaries, whether or not they might border a cross-domain grammar tree when the hybrid analyzer was run in end-to-end mode.

After the semantic dialogue units had been parsed, a training example was created for each potential boundary position in the data using the features that were listed in Table 3. The token features (features 1, 3, 7, and 9) were simply filled with the words and argument grammar labels preceding and following a potential boundary position. The probability features (features 2, 4, 8, and 10) were estimated from the training data using the counts shown in Figure 21. In addition, probabilities were estimated for unknown words and arguments. The probabilities for unknown words were estimated by counting the number of times a word that appeared only once in the training data occurred before or after a boundary and dividing by the total number of words that occurred only once in the training data. Unlike the set of words, the set of argument grammar labels is a fixed set defined by the left-hand sides of the top-level rules in the grammar. Thus,

argument grammar labels cannot be unknown in the sense that previously unseen labels will not occur during testing. However, when an unparsed word occurs before or after a potential boundary position, then the argument grammar label may be viewed as unknown. In such cases, the appropriate argument grammar label feature (feature 1 or 7) was filled with the value "?". The corresponding probabilities were estimated by counting the number of times unparsed words occurred at a boundary and dividing by the total number of unparsed words in the training data. The segment length features (features 5 and 6) were filled by counting the number of words and parse trees respectively that occurred between the beginning of the semantic dialogue unit and the potential boundary position.

A positive segmentation example was created for each boundary position between two semantic dialogue units in the same speaker turn. A negative segmentation example was created for each potential boundary position between parse trees and/or unparsed words within a semantic dialogue unit. Table 5 provides information about the data used for training and evaluating the TiMBL segmentation classifier.

	English Travel	German Travel	English Medical	German Medical
Training Examples	6992	15115	6000	3868
Positive Examples	1481 (21.2%)	4066 (26.9%)	1466 (24.4%)	1298 (33.6%)
Negative Examples	5511 (78.8%)	11049 (73.1%)	4534 (75.6%)	2570 (66.4%)

Table 5: Corpus Statistics for the segmentation classifier experiments

3.2 Testing TiMBL Parameter Settings

The TiMBL software package provides implementations of several memory-based learning algorithms as well as numerous parameters that may be used to fine-tune the performance of each algorithm. Detailed descriptions of the various algorithms and options can be found in the TiMBL reference manual ([Daelemans *et al.*, 2002]). In order to determine the best parameter settings for the segmentation classifier and to test the sensitivity of the classifier to different

settings, we first evaluated segmentation classification performance using several different parameter settings available in the TiMBL software. Our comparison by no means exhausted all of the possible parameter combinations available in the TiMBL software, but it did provide some useful insights.

In all of our experiments, we used only the IB1 algorithm. IB1 is the implementation of the k-Nearest Neighbor (k-NN) learning algorithm in TiMBL. Clearly, one parameter that may be varied in k-NN learning is *k*, the number of neighbors used to classify an instance. We tested classifiers using the following values of *k*: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 35, and 45. When the number of neighbors in the nearest neighbor set is larger than 1, the class of a new instance is determined by a vote among the neighbors. TiMBL provides several options for weighting the vote based on distance from the new instance. [Zavrel, 1997] found that the *Inverse Linear* weighting scheme generally provided better performance than the other weighting methods implemented in TiMBL. Thus, we tested classifiers using *Unweighted* (i.e. simple majority) voting and *Inverse Linear* weighting. Although the TiMBL software also offers several options for the feature weighting method and the distance metric, we did not vary those parameters in our experiments. *Gain Ratio* feature weighting was used in all of the classifiers we tested. The *Overlap* distance metric was used for the discrete features that encode the words and argument grammar labels preceding and following a potential boundary (features 1,3,7, and 9), and the *Numeric* distance metric was used for the probability and segment length features (features 2,4,5,6,8, and 10).

	English Travel	German Travel	English Medical	German Medical
k	13	13	17	15
Accuracy	0.9332	0.9227	0.9237	0.9289
P+	0.8835	0.8920	0.8695	0.9238
R+	0.7887	0.8106	0.8090	0.8590
F_1+	0.8334	0.8494	0.8382	0.8902
P-	0.9448	0.9326	0.9396	0.9312
R-	0.9721	0.9639	0.9607	0.9642
F_1-	0.9582	0.9480	0.9501	0.9474

Table 6: TiMBL segmentation classifier performance

Table 6 summarizes the performance of the best segmentation classifiers for English and German input in the Travel and Medical domains. The results shown in the table were computed over the training data using the leave-one-out testing method provided in the TiMBL software. In leave-one-out testing, each example is held out from the remainder of the training set as a test instance. The example is then classified using all of the remaining examples. This process is repeated for every example in the training data. All of the classifiers reported in the table used the *Inverse Linear* vote weighting method. The table shows the value of *k* that produced the best results along with the accuracy of the best classifier. The precision, recall, and F_1-measure (F_1 = (2*P*R)/(P+R), [van Rijsbergen, 1979]) of each classifier on positive instances (P+, R+, F_1+) and negative instances (P-, R-, F_1-) are also reported.

As Table 6 shows, the overall classification accuracy as well as the performance on negative instances was similar across languages and domains. Also, perhaps not surprisingly, the performance of the segmentation classifiers on positive instances was not as high as on negative instances. Positive instances made up a much smaller percentage of the data than negative instances, although the restriction that boundary positions could not fall inside argument parse trees increased the proportion of positive instances compared to a task where potential boundaries could fall between every pair of words. The German Medical segmentation classifier exhibited better performance on positive instances than the other three classifiers. This was probably due to the higher proportion of positive examples in the German Medical data, which resulted from the fact that there were more semantic dialogue units per speaker turn in that data set. As shown in Table 5, positive boundary examples made up 33.6% of the German Medical training data whereas the training data for the other three classifiers contained 21.2% to 26.9% positive instances.

Figure 28 illustrates the effects of changing parameter settings on segmentation classification performance for English Travel input. The graph shows the variation in overall accuracy, F_1+, and F_1- as the size of the nearest neighbor set (*k*) was increased. The solid lines in the graph represent classifiers that used the *Inverse Linear* vote weighting method, and the dashed lines represent classifiers that used *Unweighted* majority voting. We observe several interesting trends in the behavior of the English Travel segmentation classifier that are worth noting. The segmentation classifiers for the remaining language-domain pairs exhibited similar

behavior. Our findings for the task of segmenting semantic dialogue units using the k-NN methods implemented in the TiMBL software package agree with the trends reported in [Zavrel, 1997].

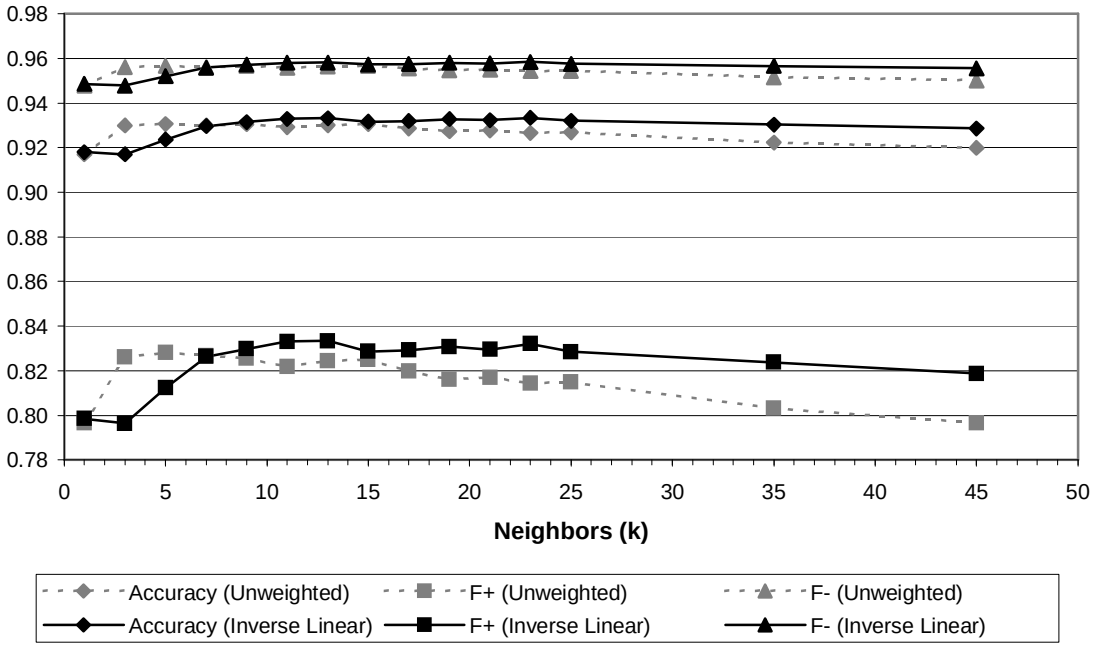

Figure 28: Effects of TiMBL parameter variation on the performance of the segmentation classifier on English Travel data

	Unweighted Voting	Inverse Linear Weighted Voting	Absolute Change
k	5	13	-
Accuracy	0.9306	0.9332	+0.0026
P+	0.8716	0.8835	+0.0119
R+	0.7887	0.7887	0.0000
F_1+	0.8281	0.8334	+0.0053
P-	0.9446	0.9448	+0.0002
R-	0.9688	0.9721	+0.0033
F_1-	0.9566	0.9582	+0.0016

Table 7: Segmentation classification performance gains for Inverse Linear weighted voting over Unweighted majority voting on English Travel data

First, except at low values of *k*, weighted voting improved the performance of the segmentation classifier over unweighted voting. Also, the optimal value of *k* for weighted voting was higher than for unweighted voting. Table 7 lists the values of the performance measures for the best unweighted and weighted classifiers and shows the change in each measure realized by using weighted voting rather than unweighted voting. Although the absolute changes were relatively small, weighted voting improved upon the performance of unweighted voting on all measures except R+ (recall on positive boundary instances), which remained the same.

	TiMBL Defaults (Unweighted Voting)	Inverse Linear Weighted Voting	Absolute Change
k	1	13	-
Accuracy	0.9170	0.9332	+0.0162
P+	0.8291	0.8835	+0.0544
R+	0.7664	0.7887	+0.0223
F_1+	0.7965	0.8334	+0.0369
P-	0.9385	0.9448	+0.0063
R-	0.9575	0.9721	+0.0146
F_1-	0.9479	0.9582	+0.0103

Table 8: Segmentation classification performance gains for Inverse Linear weighted voting over TiMBL default settings on English Travel data

Figure 28 also clearly shows that the benefit of selecting the best parameter settings for the segmentation classifiers created with the TiMBL software. The default parameters for TiMBL use the IB1 algorithm with *Unweighted* voting, *k*=1, and Gain Ratio feature weighting. These settings correspond to the *k*=1 point of the unweighted lines in the figure. Table 8 shows the improvement in performance realized by the best weighted voting classifier compared to the classifier that used the default TiMBL parameter settings.

The largest differences between the *Inverse Linear* weighted voting method and the *Unweighted* voting method were seen in the performance on positive boundary instances. Weighted voting combined with larger values of *k* provided the largest performance gains on the classification of positive boundary instances. Figure 29 provides a closer look at the variation in the F_1+ measure for the English Travel segmentation classifier shown in Figure 28. Again the

solid line represents the classifiers that used *Inverse Linear* weighted voting, and the dashed line represents the classifiers that used *Unweighted* voting (*k*=1 corresponds to the default TiMBL settings). After surpassing the performance of the unweighted classifier at *k*=9, the performance of the weighted classifier remained consistently superior to that of the unweighted classifier and did not degrade as quickly after reaching its peak at *k*=13. Furthermore, the superior performance of the weighted classifier over the best unweighted classifier was not limited to a single value of *k*. In fact, the weighted voting method was relatively robust to the specific selection of *k*, with all values of *k* between 9 and 25 for the weighted classifier providing performance superior to that of the best unweighted classifier.

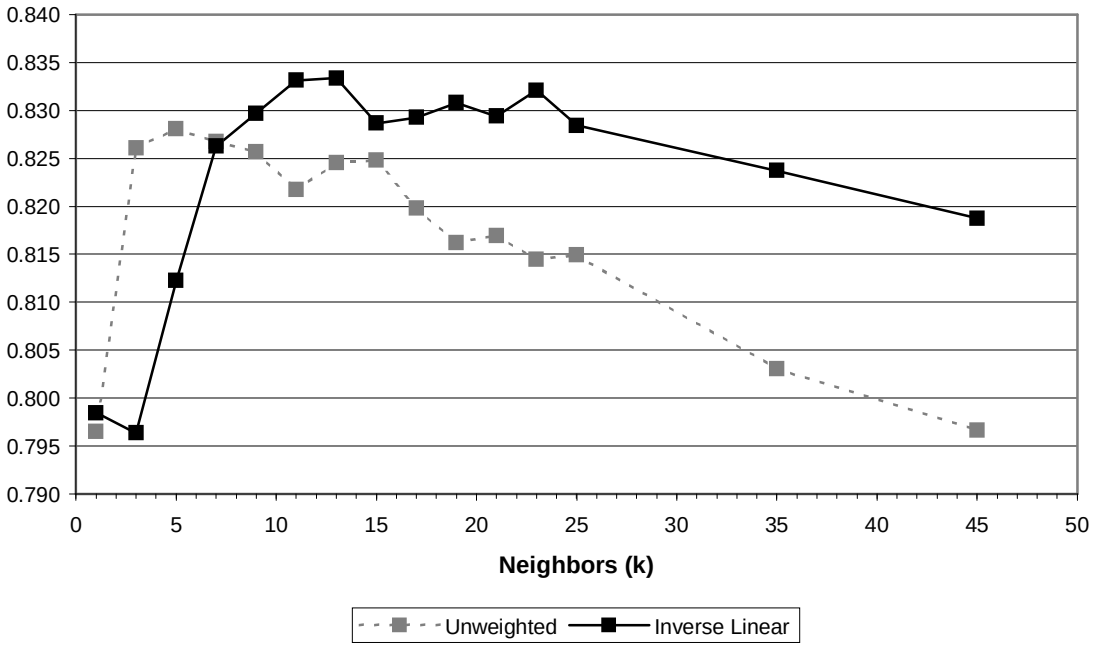

Figure 29: Effects of TiMBL parameter variation on F_1+ for the English Travel segmentation classifier

3.3 Inclusion of "Partial" Examples from Turn Boundaries

All of the results reported in the previous section were produced using only segmentation examples that occurred within each speaker turn. Segmentation classification is clearly not

necessary before the first word and after the last word of an utterance. However, as mentioned in Chapter 2, it is possible to create positive training examples for segmentation that contain partial information from the beginning and end of the utterance for each speaker turn in the training data. The positions before the first word and after the last word in a training utterance can be viewed as potential semantic dialogue unit boundaries for which training examples can be created. The information in the examples is partial in the sense that only some of the features can be filled in with true values from the utterance. The remaining features are filled in with default values.

For the beginning of the utterance, the values of the word and argument grammar label features that follow the potential boundary position (features 7, 8, 9, and 10) are filled with the first word and argument grammar label in the utterance and their corresponding probabilities. The values of both length features (features 5 and 6) are set to *0*. The features for the word and argument preceding the potential boundary (features 1 and 3) are assigned the default values "*?*". The values of the corresponding probability features are filled with the default value *1.0*. A similar positive training example is created for the end of the utterance. The values of the word and argument grammar label features that precede the potential boundary position (features 1, 2, 3, and 4) are filled with the appropriate values based on the utterance, and the length features are set based on the length of the current semantic dialogue unit. The features for the word and argument grammar label following the potential boundary position are filled with the default value "*?*", and the corresponding probability features are assigned the value *1.0*.

Although these partial examples will never perfectly match a potential boundary position found in unseen data, they may provide a close enough match in some cases to influence the segmentation decision. Without such examples, information about the first and last words and arguments in each speaker turn in the training data is retained only indirectly in the probability estimates used for segmentation. The specific words and argument grammar labels and the lengths of the last semantic dialogue unit in each turn are lost. Furthermore, all of the information that is lost borders on positive examples of semantic dialogue unit boundaries. Since all of the partial examples capture only information about true boundary positions, they may be expected to improve the performance of the classifier on positive boundary instances if any improvement occurs.

In order to test the effects of including examples with partial information on segmentation classification performance, we compared the performance of the segmentation classifiers trained only on examples extracted from within speaker turns with performance when the partial examples were included in the training data. As in the experiments with no partial examples, the data used to train the segmentation classifiers was extracted from the NESPOLE! Travel and Medical databases for English and German. All of the segmentation classifiers used the IB1 algorithm, Inverse Linear weighted voting, and Gain Ratio feature weighting. For each language-domain pair, the best value of *k* determined in the previous experiments was used. The experiments were conducted using a 20-fold cross validation setup. The segmentation training examples that were extracted strictly from within speaker turns in the data were randomly split into 20 sets, each containing 5% of the examples. For each fold, one of the sets was held out for testing, and the remaining 19 sets were used to train the classifier. In the first condition, which did not use any partial examples, only the 19 training sets were used to train the classifier. In the second condition, all of the partial examples were included in the training data for each test set. No partial examples were ever included in the test data. Within each language-domain pair, the same random split of the data was used for both conditions. Since the performance of the two training conditions on each fold was directly comparable, we used two-tailed matched pair t-tests to test for significance.

Table 9 shows the results of including partial examples in the segmentation training data for each language-domain pair. The table shows the mean values of the performance measures over the 20 sets of within-turn examples for both conditions (training with and without partial examples). The *Absolute Change* rows show the change in performance that results from adding the partial examples to the training data. The rows labeled $p<0.02$ and $p<0.005$ indicate whether the change in performance was statistically significant at each value of p using a two-tailed matched-pair t-test. The results indicate that including partial examples in the segmentation training data generally improved performance. The inclusion of partial examples increased the accuracy of the segmentation classifiers for all language-domain pairs, although only the improvement for the German Travel domain was statistically significant. Most of the remaining performance measures also showed improvement. The only exceptions were a very small decrease in R- for English Medical, and significant decreases in P+ and R- for German Medical.

English Travel							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
No Partial Examples	0.9312	0.8803	0.7832	0.8277	0.9434	0.9709	0.9569
With Partial Examples	0.9328	0.8816	0.7893	0.8315	0.9451	0.9711	0.9579
Absolute Change	+0.0016	+0.0013	+0.0061	+0.0038	+0.0017	+0.0002	+0.0010
p<0.02							
p<0.005							
German Travel							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
No Partial Examples	0.9215	0.8887	0.8098	0.8469	0.9323	0.9627	0.9472
With Partial Examples	0.9240	0.8924	0.8161	0.8521	0.9344	0.9637	0.9488
Absolute Change	+0.0025	+0.0037	+0.0063	+0.0052	+0.0021	+0.0010	+0.0016
p<0.02	x		x	x	x		x
p<0.005	x		x	x	x		
English Medical							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
No Partial Examples	0.9200	0.8678	0.7945	0.8280	0.9353	0.9607	0.9477
With Partial Examples	0.9235	0.8684	0.8094	0.8367	0.9400	0.9603	0.9499
Absolute Change	+0.0035	+0.0006	+0.0149	+0.0087	+0.0047	-0.0004	+0.0022
p<0.02			x	x	x		
p<0.005							
German Medical							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
No Partial Examples	0.9297	0.9233	0.8627	0.8913	0.9320	0.9643	0.9477
With Partial Examples	0.9325	0.9111	0.8860	0.8976	0.9423	0.9569	0.9493
Absolute Change	+0.0028	-0.0122	+0.0233	+0.0063	+0.0103	-0.0074	+0.0016
p<0.02		x	x		x	x	
p<0.005		x	x		x	x	

Table 9: Segmentation classification performance with partial examples

The largest impact of including the partial examples in training can be seen in the performance of the segmentation classifiers on positive boundary instances. In particular, the largest improvements across all language-domain pairs were seen for R+ (recall of positive instances) and F_1+. The improvements in R+ were at least marginally significant for 3 of the 4 language-domain pairs, and the improvements in F_1+ were significant for 2 of the 4 pairs. These results fit well with our expectations regarding the effects of including the partial information for the true boundary positions at the beginning and end of an utterance. The improvement in R+ indicates that the partial examples enabled the segmentation classifiers to correctly identify more of the positive boundary instances in the test data. Furthermore, it appears that for the most part, the partial examples did not adversely affect the correct classification of negative examples. The

only exception was German Medical, in which a performance drop was seen for P+ and R-. This is indicative that the inclusion of partial examples caused the segmentation classifier to incorrectly identify more negative instances as boundaries in addition to correctly identifying more true positive instances. Nevertheless, the improvement in the identification of positive instances outweighed the negative effects of the increase in false positives, resulting in improvements in accuracy, F_1+, and F_1-. Since the performance of the segmentation classifiers on positive instances was weaker than on negative instances to begin with, this seems like a worthwhile tradeoff.

The results shown in Table 9 where all produced using the best TiMBL parameters for training data without any partial examples. Because the inclusion of partial examples clearly alters the content of the training set, it was possible that the segmentation classification results with partial examples included in the training data could be further improved by selecting new parameter settings. Since it was clear that the Inverse Linear vote weighting method outperformed the Unweighted voting method for our semantic dialogue unit boundary classifiers, we tested classifiers that used Inverse Linear weighted voting with different values of *k* when partial examples were included in the training data.

k = 13							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
No Partial Examples	0.9312	0.8803	0.7832	0.8277	0.9434	0.9709	0.9569
With Partial Examples	0.9328	0.8816	0.7893	0.8315	0.9451	0.9711	0.9579
Absolute Change	+0.0016	+0.0013	+0.0061	+0.0038	+0.0017	+0.0002	+0.0010
p<0.02							
p<0.005							
k = 17							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
No Partial Examples	0.9312	0.8803	0.7832	0.8277	0.9434	0.9709	0.9569
With Partial Examples	0.9359	0.8884	0.7978	0.8395	0.9474	0.9728	0.9598
Absolute Change	+0.0047	+0.0081	+0.0146	+0.0118	+0.0040	+0.0019	+0.0029
p<0.02	x		x	x	x		x
p<0.005							

Table 10: Segmentation classification performance with partial examples and best *k* on English Travel

Table 10 shows the results for the best value of *k* for the English Travel data set when partial examples were included in the training data. When the partial examples were excluded from the training data, the best classifier for the English Travel data used a value of 13 for *k*. For the experiments with partial examples included, we only tested a small set of values of *k* (11, 13, 15, 17, 19, 21, and 23) near the original best value of *k* and the peak of the curve shown in Figure 23. With the partial examples included, all of the values of *k* that we tested resulted in improved performance over the classifier that excluded partial examples, and the best value of *k* changed from 13 to 17. As the table shows, the improvement in each of the performance measures at least doubled with the best value of *k*. Additionally, most of improvements relative to the classifier trained without partial examples became significant at the $p<0.02$ level.

3.4 Comparison with the Unigram Threshold Baseline Model

The simple statistical unigram threshold model for segmentation described in Section 2.4.2.1 can be used as a baseline against which to compare the performance of the memory-based TiMBL segmentation classifiers. In order to make a direct comparison of the two models, the unigram threshold model must be modified slightly to make decisions about potential boundary positions that may be bordered by unparsed words in addition to argument parse trees. Unparsed words can be replaced with the same default token used for the argument label features (1 and 7) in the TiMBL segmentation classifier input, and the counts for the default token used to make the probability estimates for the TiMBL classifier can be used in the unigram formula to estimate the likelihood of a semantic dialogue unit boundary.

We first determined the optimum threshold value for the unigram segmentation model. Using the argument labels and boundary class from the training examples for the TiMBL segmentation classifiers in conjunction with the argument parse label counts computed during preparation of the training data, we produced test cases for the unigram threshold model. For each example, the likelihood of a semantic dialogue unit boundary according to the unigram segmentation model was calculated. Then for each possible threshold, the classification of the unigram model was compared with the correct classification from the TiMBL training example. We tested thresholds from 0.0 to 1.0 in increments of 0.01. A threshold of 0.0 meant that every potential boundary position was classified as a boundary, and a threshold of 1.0 meant that every position was classified as a non-boundary. Based on the way the model is defined, R+ (recall of

positive boundary instances) decreases monotonically as the threshold is increased, and R- (recall of negative boundary instances) increases monotonically. Ideally, a threshold could be identified such that most of the positive instances fell above the threshold and most of the negative instances fell below the threshold.

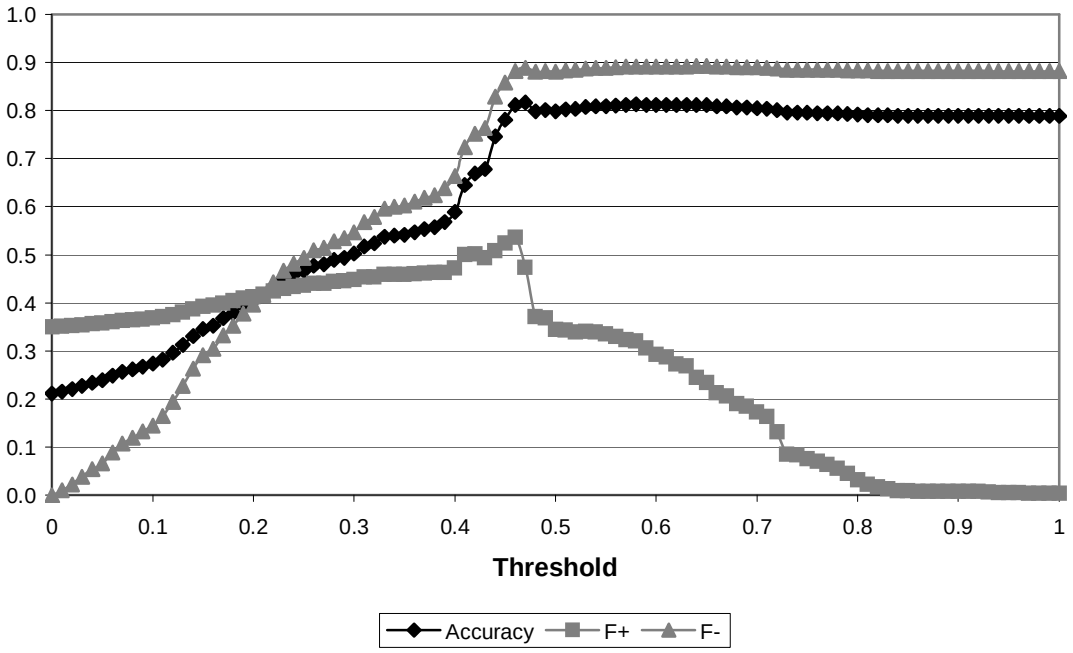

Figure 30: Effects of threshold variation on performance of the unigram segmentation model for English Travel data

Figure 30 illustrates the effects of varying the threshold for the unigram segmentation model on the NESPOLE! English Travel & Tourism data. The graph displays overall accuracy, F_1+, and F_1- as the threshold was increased from 0.0 to 1.0 in increments of 0.01. Identifying a useful threshold for the unigram segmentation model proved to be difficult. For each language-domain pair, we selected the threshold that maximized classification accuracy, F_1+, and F_1-. All three measures were maximized with the same threshold for the English and German Medical Assistance data. For the English Travel data, the F_1+ measure was maximized with a threshold one increment lower than accuracy and F_1- (0.46 versus 0.47). The same was true for the German Travel data. In these cases, we chose the threshold that maximized F_1+ in order to optimize

performance on positive boundary instances. Since the main goal of semantic dialogue unit segmentation is to identify the positive instances, a small tradeoff of overall accuracy and performance on negative instances was justified.

English Travel							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
Unigram Threshold Model (Threshold = 0.46)	0.8111	0.5580	0.5161	0.5343	0.8725	0.8907	0.8813
TiMBL Classifier (k=17)	0.9359	0.8884	0.7978	0.8395	0.9474	0.9728	0.9598
Absolute Improvement	+0.1248	+0.3304	+0.2817	+0.3052	+0.0749	+0.0821	+0.0785
Percent Improvement	15.4%	59.2%	54.6%	57.1%	8.6%	9.2%	8.9%
German Travel							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
Unigram Threshold Model (Threshold = 0.47)	0.7779	0.6002	0.5267	0.5601	0.8331	0.8707	0.8513
TiMBL Classifier (k=17)	0.9251	0.8977	0.8145	0.8536	0.9340	0.9658	0.9496
Absolute Improvement	+0.1472	+0.2975	+0.2878	+0.2935	+0.1009	+0.0951	+0.0983
Percent Improvement	18.9%	49.6%	54.6%	52.4%	12.1%	10.9%	11.54%
English Medical							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
Unigram Threshold Model (Threshold = 0.39)	0.8250	0.6662	0.5724	0.6138	0.8679	0.9063	0.8865
TiMBL Classifier (k=17)	0.9304	0.8990	0.8094	0.8505	0.9400	0.9697	0.9545
Absolute Improvement	+0.1054	+0.2328	+0.2370	+0.2367	+0.0721	+0.0634	+0.0680
Percent Improvement	12.8%	34.9%	41.4%	38.6%	8.3%	7.0%	7.7%
German Medical							
	Accuracy	**P+**	**R+**	**F_1+**	**P-**	**R-**	**F_1-**
Unigram Threshold Model (Threshold = 0.48)	0.8157	0.7318	0.7123	0.7202	0.8562	0.8683	0.8617
TiMBL classifier (k=11)	0.9341	0.9204	0.8803	0.8989	0.9402	0.9619	0.9507
Absolute Improvement	+0.1184	+0.1886	+0.1680	+0.1787	+0.0840	+0.0936	+0.0890
Percent Improvement	14.5%	25.8%	23.6%	24.8%	9.8%	10.8%	10.3%

Table 11: Comparison of performance of the TiMBL segmentation classifiers and the unigram threshold segmentation models

In order to compare the unigram segmentation model with the TiMBL segmentation classifiers for each language-domain pair, we applied the unigram model to the 20-fold data split used for the experiments with partial examples described in the previous section. Table 11 shows

the average value of each performance measure using the TiMBL classifiers with the best value of *k* trained with partial examples and the unigram threshold models with the best threshold value. The absolute change in each performance measure and the percent improvement of the TiMBL classifier over the unigram threshold model are also shown in the table. The TiMBL segmentation classifiers substantially improved segmentation classification performance over the baseline unigram models on all measures.

The largest performances gains for the TiMBL classifiers came in the identification of positive boundary instances. Although the overall accuracy of the unigram threshold models does not appear to be too bad, their performance on positive boundary instances is poor even at the best threshold values. The higher accuracy was mostly due to the fact that negative boundary instances made up the majority of the data, and the unigram models did reasonably well at identifying negative instances. The TiMBL segmentation classifiers did a much better job of finding the boundary positions and of balancing performance on boundary and non-boundary positions.

3.5 Performance on Full Speaker Turns

All of the experiments reported so far have dealt with the performance of the segmentation classifiers on input that was "clean" in the sense that the argument parser was not allowed to parse across semantic dialogue unit boundaries. When semantic dialogue unit segmentation is performed as part of online analysis, the task is more difficult because there is no such restriction. The argument parser receives the utterance for a full speaker turn, which may contain multiple semantic dialogue units, as input. Thus, argument parse trees may occasionally span across a boundary. Since positions within argument parse trees are not candidates for segmentation, such boundaries will clearly be missed be the segmentation classifiers. On the other hand, when semantic dialogue unit segmentation is performed online, the task may be made somewhat easier by the fact that the cross-domain grammar is used during argument parsing and boundaries are inserted by definition on both sides of cross-domain parse trees.

In order to evaluate the performance of semantic dialogue unit detection in an online setting, we examined the output of the hybrid analyzer for English used in the final end-to-end evaluation of the Travel & Tourism domain for the NESPOLE! project (also described in Section 5.2). The test set consisted of all of the utterances from the client side of 2 complete dialogues in

the NESPOLE! Travel domain. The dialogues in the test set contained 110 utterances composed of 232 semantic dialogue units. The end-to-end evaluation for NESPOLE! was conducted by running each utterance in the test set through the full online NESPOLE! translation system, which included the online version of the hybrid analyzer. The grammars, training data, and classifiers used were the best available at the time of the evaluation. These differ from those reported in previous sections in this chapter because additional grammar development and annotation were performed after the evaluation of the Travel domain during the porting of the NESPOLE! system to the Medical domain. At the time of the Travel domain evaluation, the dialogues in the test set were unseen both from the perspective of grammar development and from the perspective of training the classifiers in the hybrid analyzer. After the evaluation was run, the dialogues were annotated with semantic dialogue unit boundaries and Interchange Format representations in the same way as the remainder of the NESPOLE! databases.

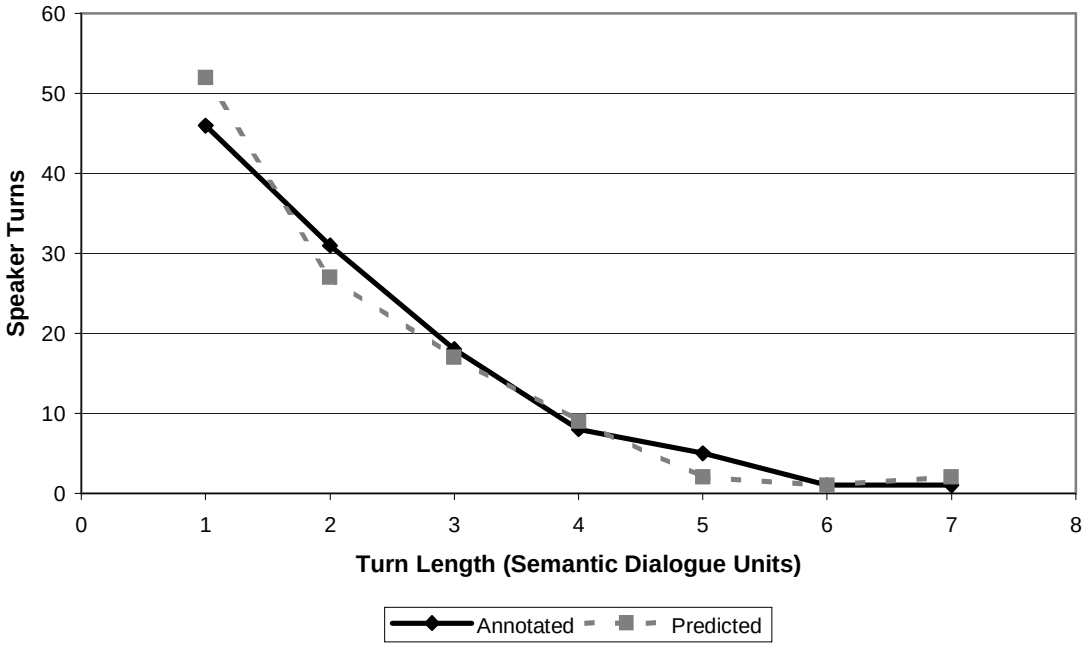

Figure 31: Frequency of speaker turn lengths for English Travel test data

We first examine how well the segmentation performed by the online hybrid analyzer identified the number of semantic dialogue units in a speaker turn regardless of the specific positions of the boundaries. The annotated test set data contained a mean of 2.11 semantic dialogue units per turn (232 semantic dialogue units in 110 speaker turns). Figure 31 illustrates the frequency of speaker turn lengths in the annotated test set as well as the frequency of turn lengths predicted by semantic dialogue unit boundary detection in the online analyzer. The automatic boundary detection in the hybrid analyzer undersegmented slightly compared to the annotated data, predicting a mean of 2.03 semantic dialogue units per turn (223 semantic dialogue units in 110 turns). Nevertheless, the frequency of turn lengths produced by automatic segmentation corresponded fairly well with the frequency of turn lengths in the annotated data.

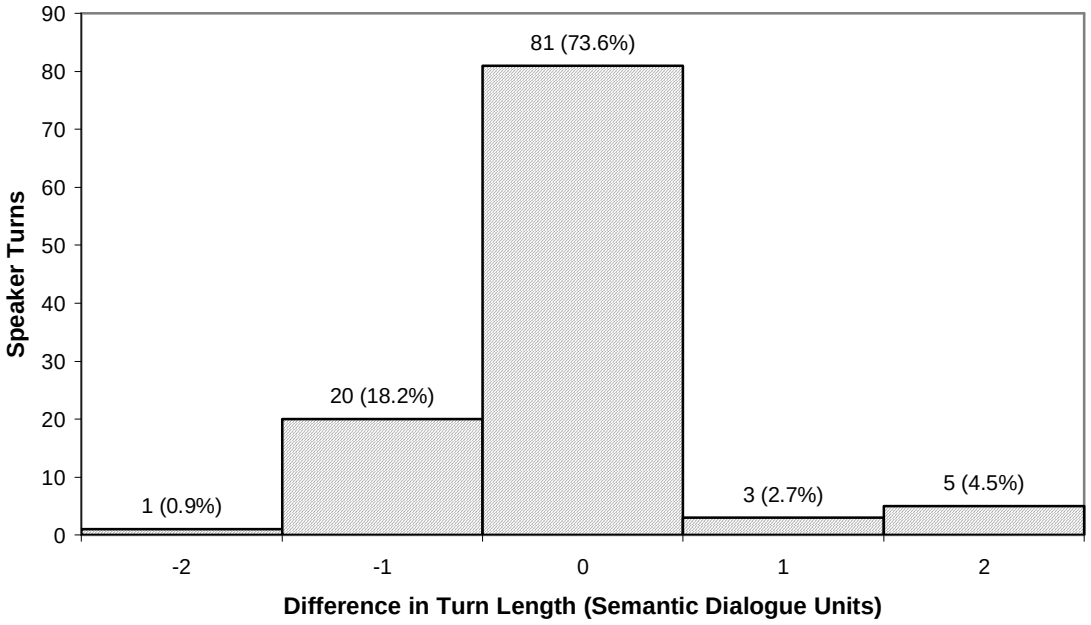

Figure 32: Distribution of differences between predicted and annotated speaker turn lengths for English Travel test data

In addition to the mean number of semantic dialogue units and the frequency of turn lengths, it is also important to know how well the predicted turn length matches the annotated turn length for individual turns. Figure 32 illustrates the distribution of the differences between

the predicted and annotated turn lengths for the test set. The x-axis shows the difference produced by subtracting the number of annotated semantic dialogue units for a turn from the number of semantic dialogue units predicted by automatic boundary detection. A negative value indicates that automatic segmentation predicted too few semantic dialogue units for the turn, and a positive value indicates that too many semantic dialogue units were predicted. The y-axis shows the number of turns for which each difference occurred. As Figure 32 shows, automatic segmentation correctly predicted the number of semantic dialogue units for 73.6% of the turns. Furthermore, automatic segmentation was never off by more than 2 semantic dialogue units for any turn, and the predicted length was within 1 semantic dialogue unit of the correct length for 94.5% of the turns. Thus, it appears that the automatic segmentation in the online hybrid analyzer did a good job of correctly identifying the number of semantic dialogue units in full speaker turns.

We now turn to an examination of how well the positions of the semantic dialogue unit boundaries were identified. Two types of segmentation errors are possible in the detection of boundaries for complete speaker turns in the online version of the hybrid analyzer. The first type of error occurs when potential boundary positions between argument parse trees and/or unparsed words are incorrectly classified. These errors may arise because the cross-domain grammar incorrectly parsed a phrase as a full domain action, resulting in the automatic insertion of an incorrect boundary on at least one side of the cross-domain tree. Such errors may also occur because the segmentation classifier misclassifies a potential boundary position. These errors reflect the performance of the semantic dialogue unit boundary detector (i.e. the combination of automatically inserted boundaries around cross-domain trees and the TiMBL segmentation classifier). The second type of segmentation error occurs when a tree in the argument parse spans a true semantic dialogue unit boundary. In such cases, the semantic dialogue unit boundary detector does not have a chance to identify the boundary because it falls within an argument parse tree. Thus, these errors contribute to the overall segmentation performance of the hybrid analysis approach, but they do not reflect the performance of the semantic dialogue unit boundary detector.

We look first at the performance of the semantic dialogue unit boundary detector based on the first type of error. After argument parsing, there were 325 potential boundary positions between parse trees and/or unparsed words. 101 (31.1%) of the positions were semantic dialogue

unit boundaries, and 224 (68.9%) of the positions were non-boundaries. Table 12 shows the performance of the semantic dialogue unit boundary detector on those potential boundary positions where a segmentation choice must be made. The values of the performance measures reported are not directly comparable with the previous experiments because the grammars, training data, and classifiers were not identical. Nevertheless, as expected, with unseen data and argument parsing of complete speaker turns rather than individual semantic dialogue units, overall segmentation performance dropped somewhat relative to the performance on "clean" data in the previous experiments. One particularly positive result was the recall of the semantic dialogue unit boundary detector on positive boundary instances, which was 0.8614 on the positions that the boundary detector had a chance to find (i.e. those not spanned by an argument parse tree).

Accuracy	0.8769
P+	0.7699
R+	0.8614
F_1+	0.8131
P-	0.9340
R-	0.8839
F_1-	0.9083

Table 12: Segmentation performance on complete speaker turns in unseen English Travel data

We also examined the overall segmentation performance of the hybrid analyzer including the second type of segmentation error in which true boundaries are spanned by argument parse trees. There were a total of 122 semantic dialogue unit boundaries annotated in the 2 test set dialogues. After argument parsing, 21 (17.2%) of the annotated boundaries were spanned by an argument parse tree. Taking those true boundaries that were covered by a parse tree into consideration, the overall recall of semantic dialogue unit boundaries was 0.7131 (87/122). However, a closer examination of the argument parsing errors associated with the covered boundaries and the resulting segmentations reveals that the errors were not as severe as they may appear at first glance. In 18 of the 21 cases where a parse tree spanned a semantic dialogue unit boundary, a boundary was identified immediately after (16) or before (2) the spanning tree. This

means that the semantic boundary detector identified a boundary as close as it could to the true boundary. In many such cases, the shifted boundary position did not cause any meaningful change in the translation output. Such cases also accounted for several of the false positives (non-boundary positions that were classified as boundaries) created by the boundary detector.

4 of the 21 boundary spanning errors were caused by inconsistencies between the annotation of semantic dialogue units and the cross-domain grammar rules for parsing those semantic dialogue units. For example, the phrase "*okay all right*" is often used to indicate acknowledgement. According to the conventions used for annotating the NESPOLE! data, the phrase contains 2 semantic dialogue units ("*okay*" and "*all right*"), and both semantic dialogue units are annotated with the domain action *acknowledge*. However, the cross-domain grammar contains a rule for the *acknowledge* domain action that parses "*okay all right*" as a single domain action. Segmentation errors such as these do not alter the meaning conveyed in the resulting translation.

An additional 7 of the 21 boundary spanning errors did not alter the acceptability of the resulting translation. For example, one of the test set utterances was "*i think july like around july first*". The utterance was annotated as two domain actions ("*i think july*" and "*like around july first*"). However, during argument parsing, the two sequential time expressions "*july*" and "*like around july first*" were parsed under a single parse tree. The English-to-English paraphrase with the annotated segmentation would have been "*It is possible that July. July the 1st approximately.*" With the incorrect segmentation due to the spanning argument parse tree, the paraphrase was "*It is possible that July, July the 1st approximately.*" Considering these errors along with the errors that resulted from inconsistencies between the grammars and annotations, our error analysis indicates that only 10 of the 21 boundary spanning errors changed the meaning conveyed by the translation. Thus only 8.2% (10/122) of the semantic dialogue unit boundaries in the test set were spanned by a parse tree with a meaningful negative effect on translation.

Chapter 4 Evaluation of Domain Action Classification

In this chapter, we describe experiments designed to empirically examine several aspects of the domain action classification task. We evaluate domain action classification on English and German input in the NESPOLE! Travel & Tourism and Medical Assistance domains. The purpose of the first round of experiments is to test the feasibility of automatically classifying domain actions in the NESPOLE! domain and to compare the performance of several different machine learning techniques on the task of domain action classification. We also assess domain action classification performance when the classification task is broken down in different ways. We then explore the effects of using a variety of feature sets as input to the domain action classifier. We also examine the effects of including various data sets in the training data for the classifiers. Finally we assess the effects of using a fallback strategy that guarantees that valid Interchange Format representations are produced when the output of domain action classification is combined with the output of the argument parsing.

4.1 Preparation of Training Data

We conducted experiments to evaluate domain action classification performance using English and German input from the NESPOLE! Travel & Tourism and Medical Assistance domains. For each language-domain pair, the primary corpus used in the domain action classification experiments was the NESPOLE! database. The databases used in the domain action classification experiments were similar to those used in the segmentation experiments reported in Chapter 3. However, unlike the segmentation classification experiments, which were conducted using the final version of the NESPOLE! databases, the domain action classification experiments were conducted using an earlier version of the databases that was available at the time the experiments were run. The earlier version of the databases was annotated in the same format as the final versions, containing manually transcribed dialogues annotated with speaker turns, semantic dialogue unit boundaries, and Interchange Format representations. The main difference between the earlier version of the databases and the final version was the extent to which all of the semantic dialogue units were annotated with Interchange Format representations. The earlier databases were annotated less completely than the final versions. Although the annotated

Interchange Format representations are not necessary for training the segmentation classifier, they are clearly required for training the domain action classifiers. The databases used were the most up-to-date versions of the databases that were available at the time the experiments were run, and all of the experiments for a language-domain pair were conducted using the same version of the database for that pair. Likewise, the grammars used for each language-domain pair were the most up-to-date versions available at the time of the experiments. The grammars used for the Travel & Tourism domain were the final versions of the grammars for the NESPOLE! system. The grammars used for the Medical Assistance domain were nearly final versions that were available before the development of full domain action grammars for the portability experiment (Section 5.3).

The NESPOLE! databases used in the domain action classification experiments were also supplemented when possible with additional databases annotated using the same format. The English and German databases for the Travel & Tourism domain were supplemented with data from the C-STAR II project, which had been annotated using the updated NESPOLE! Interchange Format definition. This was reasonable since the domain of coverage of the NESPOLE! Travel & Tourism domain was partly an expansion of the Travel Planning domain from the C-STAR II system, and the initial annotation of the NESPOLE! data often focused on semantic dialogue units that were new to the NESPOLE! domain. The English database for the Medical Assistance domain was supplemented with data from the Babylon project in a similar way. The NESPOLE! database for each language-domain pair was also supplemented with translated data from the same domain. For example, dialogues in the German Travel database that had originally been collected in German were manually translated into English and used to supplement the English database and vice versa.

The preparation of the training data for the domain action classification experiments was similar to the preparation of the training data for segmentation classification. First, semantic dialogue units from both speaker sides (client and agent) were extracted from the database. Only those semantic dialogue units for which there was source language text and a corresponding Interchange Format representation were extracted for training the domain action classifiers. As in preparing the segmentation training data, all punctuation and case information was removed from the source language text so that the text basically corresponded to perfect output from an automatic speech recognizer. The text for each semantic dialogue unit was then parsed with the

argument and pseudo-argument (and shared) grammars. In any experiments in which word-based features were used, the classifiers were trained on the text of the semantic dialogue unit after the application of any pre-parsing string mappings applied by the argument parser. Because the task of the domain action classifiers was to identify the domain action for each semantic dialogue unit and because we wanted to abstract away from the problem of segmentation, the argument parser was not allowed to produce parses that crossed semantic dialogue boundaries. However, since parse-based features in the input to the online domain action classifiers could include incorrectly parsed arguments or pseudo-arguments, such parse errors were allowed in the training data.

After argument parsing, a training example was created for each semantic dialogue unit by extracting the appropriate input features and class information from the database and/or argument parse. The possible input feature sets for the domain action classifiers were listed in Figure 23. The specific set of input features and the classes used to create the training examples varied depending on the aspect of domain action classification being tested in a particular experiment. Thus, more detail will be provided where appropriate in the descriptions of the experiments.

	English Travel	**German Travel**	**English Medical**	**German Medical**
Semantic Dialogue Units	8289	8719	3664	2294
Domain Actions	972	1001	462	286
Speech Acts	70	70	50	43
Concept Sequences	615	638	305	179
Individual Concepts	109	110	75	62
Vocabulary	1946	2815	1694	1112

Table 13: Corpus information for the databases from which the training data used in the domain action classification experiments was extracted

Table 13 contains information regarding the contents of the corpus from which the training data for the domain action classification experiments was extracted. The first row shows the number of semantic dialogue units extracted from the database for each language-domain pair. The remaining rows show the number of unique *types* of each kind of information found in the extracted semantic dialogue units. The statistics in the table illustrate that domain action

classification is a very challenging classification problem, with approximately 1000 classes found in the databases for the Travel domain. Even if the domain action classification task is divided into subproblems of identifying the speech act and concept sequence separately, the subtasks remain quite difficult. The difficulty is compounded by relatively sparse training data with unevenly distributed classes. Although there were hundreds to well over 1000 instances of the most frequent classes in our training corpus, the data contained only 1 or 2 instances for most of the classes. For example, in the English Travel & Tourism training data, 52% of the domain action types had only 1 instance, and an additional 14% had only 2 instances. Thus, 66% of the domain actions that appeared in the English Travel training data occurred only 1 or 2 times.

	English Travel	**German Travel**
Domain Action	19.2% *acknowledge*	19.7% *acknowledge*
Speech Act	41.4% *give-information*	40.7% *give-information*
Concept Sequence	38.9% No Concepts	40.3% No Concepts

Table 14: Most frequent domain actions, speech acts, and concept sequences in the training data used for the domain action classification experiments in the Travel domain

	English Medical	**German Medical**
Domain Action	25.1% *give-information+experience+health-status*	27.2% *acknowledge*
Speech Act	59.7% *give-information*	35.3% *give-information*
Concept Sequence	35.0% *+experience+health-status*	47.3% No Concepts

Table 15: Most frequent domain actions, speech acts, and concept sequences in the training data used for the domain action classification experiments in the Medical domain

Table 14 and Table 15 show the most common domain actions, speech acts, and concept sequences in the training data used for the domain action classification experiments described in this chapter along with their associated frequencies. The frequencies provide a baseline

performance that could be achieved using a very simple domain action classifier that always returned the most common class from the training data. With the exception of the English Medical domain, the domain action returned by a single domain action classifier would be different than the domain action returned by combining the outputs of separate speech act and concept sequence classifiers. In either case, *give-information+experience+health-status* would be the domain action produced by the classifier for the English Medical domain. A single classifier would return *acknowledge* for the remaining language-domain pairs. A combined classifier would produce *give-information*, which is an illegal domain action because the Interchange Format specification requires that the *give-information* speech act be followed by a non-empty concept sequence. Thus, these simple baseline classifiers also demonstrate the need for a fallback strategy to find alternative legal domain actions when separate speech act and concept sequence classifiers are used.

4.2 Comparison of Learning Approaches

In our first set of experiments, we compared the performance of several machine learning approaches on the task of domain action classification and the related subtasks of speech act classification and concept sequence classification. As listed in Table 4, we tested memory-based learning (k-Nearest Neighbor) using TiMBL ([Daelemans *et al.*, 2002]), decision tree learning using C4.5 ([Quinlan, 1993]), neural networks using SNNS ([Zell *et al.*, 1998]), and naïve Bayes n-gram classifiers using Rainbow ([McCallum, 1996]). We tested each of the learning approaches on English and German input in the Travel & Tourism and Medical Assistance domains. Our first goal in conducting these experiments was to establish the feasibility of classifying domain actions, speech acts, and concept sequences using information from argument parsing. Our second goal was to compare the performance of the machine learning approaches on the task of domain action classification. In particular, we wanted to select a learning approach to be used in the online version of the hybrid analyzer used in the NESPOLE! translation servers and for exploring further aspects of domain action classification in later experiments.

4.2.1 Experimental Setup

We tested domain action classifiers, speech act classifiers, and concept sequence classifiers for each learning approach applied to each language-domain pair. The training data for each

language-domain pair was processed as described in the previous section. In order to focus on the use of argument parse information for domain action classification, the input features for the classifiers were extracted almost exclusively from the argument parse. The input features for the domain action classifier consisted of binary features indicating the presence or absence of root node labels from the argument and pseudo-argument grammars in the single best parse for the semantic dialogue unit. This corresponds to the use of feature sets 1 and 2 listed in Figure 23. For the Travel & Tourism domain, the input feature set included 212 features for English and 259 features for German. For the Medical Assistance domain, there were 252 features for English and 297 features for German. The number of features was determined by the number of top-level rules in the argument and pseudo-argument grammars for each language and domain. The input feature set for the speech act classifier was identical to that used for the domain action classifier. The concept sequence classifier also included the same set of features. The input feature set for the concept sequence classifier also included one additional feature for the speech act assigned to the semantic dialogue unit (feature 5 in Figure 23).

The same set of input features was used for all of the learning approaches, and each approach returned the best class as output. The features and output class were represented in a manner appropriate for each approach. For the memory-based and decision tree classifiers implemented using TiMBL and C4.5 respectively, the grammar label input features were represented as a vector of binary features, and the speech act feature was represented as a single feature whose possible values were the set of speech acts. The output of the TiMBL and C4.5 classifiers was the single best class identified by the classifier. The neural network classifiers implemented with SNNS used a similar input feature representation. Each binary grammar label feature was represented as a single input unit in the neural network. The value of the input unit was 1 if the feature was active (i.e. the grammar label was present in the argument parse for the semantic dialogue unit) and 0 if the feature was inactive. The speech act feature for the concept sequence classifier was represented using a set of binary input units. The value of the input unit associated with the speech act for the semantic dialogue unit was 1, and the value all of other speech act input units was 0. The output layer of the SNNS classifiers consisted of one output unit for each possible class. The output unit with the highest activation identified the best class. In order to simulate the binary features used by the other classifiers as closely as possible, the naïve Bayes classifiers implemented with Rainbow used a simple unigram model whose

vocabulary was the set of grammar labels included in the binary feature set. The speech act feature was not included in the Rainbow concept sequence classifier. The Rainbow classifiers returned the class with the highest probability as output.

Each experiment was conducted using a 20-fold cross-validation setup. The training corpus for each language-domain pair was randomly divided into 20 sets of equal size. Each of the sets was held out as the test set for one fold with the remaining 19 sets used as training data. In each fold, the TiMBL, C4.5, and Rainbow classifiers were trained on 19 subsets of the data and tested on the remaining heldout set. The SNNS classifiers required a more complex setup to determine the number of epochs to train the neural network for each test set. Within each fold, a cross-validation setup was used to determine the number of training epochs. Each of the 19 training subsets for a fold was used as a validation set. The network was trained on the remaining 18 subsets until the accuracy on the validation set did not improve for 50 consecutive epochs. The network was then trained on all 19 training subsets for the average number of epochs determined using the validation sets. This process was used for all 20-folds in the speech act classification experiment. For the domain action classification and concept sequence classification experiments, this process ran for approximately 1.5 days for each fold. Thus, this process was run for only the first two validation folds, and the average number of epochs from those folds was used for training. Within each language, the same random split was used for all of the experiments. Because the same split of the data was used for different classifiers, the results of two classifiers on the same test set are directly comparable. Thus, we tested for significance using two-tailed matched pair t-tests.

4.2.2 Selection of Parameter Settings

Because our purpose was not to implement each learning approach from scratch but rather to test learning approaches for the task of domain action classification, we used the software for each learning approach “off the shelf” in all of our experiments. We conducted the experiments using a few different settings for some of the main parameters for each learning approach. The goal of the parameter testing was not to exhaustively search all possible combinations of parameters in order to tweak every last bit of performance out of each learning approach. Rather, the purpose of the testing was simply to check the sensitivity of each approach to variation in parameter settings and to be confident that the settings used in our experiments were providing reasonable

performance. The default parameter settings for each learning approach were among the settings tested, and default parameter settings were used for any parameters that are not explicitly mentioned.

4.2.2.1 TiMBL

Several different memory-based learning algorithms are implemented in the TiMBL software package along with a large variety of options for fine-tuning the performance of each algorithm. Details of the various algorithms and options may be found in the TiMBL reference manual ([Daelemans *et al.*, 2002]). In our domain action classification experiments, we used only the IB1 algorithm, which is the implementation of k-Nearest Neighbor learning in TiMBL. The Overlap distance metric was used for all of the features in all of the classifiers. TiMBL also provides several feature weighting methods that can automatically determine the importance of input features based on the training data. The feature weighting method determines how the input features are weighted in determining the distance of training examples from an input instance. The Gain Ratio feature weighting method was used in all of the TiMBL classifiers we evaluated.

As in the semantic dialogue unit segmentation experiments, we tested two parameters for the TiMBL classifiers. Clearly one of the parameters that can be varied in k-Nearest Neighbor learning is *k*, the number of neighbors used to classify input instances. We tested values of 1, 3, 5, 7, 9, 11, 13, and 15 for *k* for the classifiers in the domain action classification experiments. Additionally, the classification of an input instance is determined by a vote among the neighbors in the nearest neighbor set. The vote weighting method determines how the votes of the neighbors are weighted. With no weighting, the best class is determined by a simple majority vote. TiMBL also provides several weighting methods that weight the vote of a neighbor by a function of its distance from the input instance. As in the segmentation experiments, we tested unweighted voting and Inverse Linear weighted voting for the classifiers in the domain action classification experiments.

In testing the TiMBL parameters with this feature set, we found that using values of *k* larger than 1 generally decreased performance across all classification tasks and language-domain pairs. We found that using Inverse Linear weighted voting generally reduced, and occasionally eliminated, the performance drop for values of *k* higher than 1, but higher values of *k* never significantly improved performance. There was no significant difference in performance

between unweighted and Inverse Linear weighted voting for k=1. Based on these observations, we used k=1 with Inverse Linear weighted voting for all of the learner comparison experiments.

4.2.2.2 C4.5

In our domain action classification experiments, we used the decision tree learning algorithm implemented in C4.5 ([Quinlan, 1993]). Batch mode tree generation was used for all of the classifiers to produce a single tree from all of the training data. The default gain criterion (Gain Ratio) and pruning confidence level (25%) were used in all of the experiments. We varied two parameters in C4.5. First, the default value for the minimum number of instances required in at least two branches for any test is 2. Because many of the classes were represented by only one instance in the training data, we also tested with a minimum of 1 object per branch for a valid test. Additionally, we tested performance with and without pruning following the creation of the tree from the training data. Performance did not change substantially under any combination of conditions, but we generally obtained the best performance by allowing a minimum of 1 instance per branch and using pruning.

4.2.2.3 SNNS

Our neural network classifiers were implemented using Stuttgart Neural Network Simulator. The SNNS software provides a large variety of options for training, structuring, and tuning neural networks that are described in [Zell *et al.*, 1998]. In our domain action classification experiments, we used simple fully connected feed-forward networks with one hidden layer for all of the classifiers. The networks were trained using backpropagation. The order of presentation of the training examples was randomized in each training epoch, and the weights were updated after the presentation of each training example. The only parameter that we varied for the neural network classifiers was the number of units in the hidden layer.

Table 16 lists the number of input and output units for the networks for each classification task. The domain action, speech act, and concept sequence classifiers all had one input unit for each top-level grammar label feature. The concept sequence classifiers had additional input units to encode the speech act feature. The classifiers all had one output unit for each class in the training data. We tested hidden layers containing 15, 30, and 60 units for the speech act classifiers and hidden layers with 25, 50, and 100 units for the domain action and speech act classifiers. We found that the networks with the middle or larger hidden layer generally

outperformed networks with the smaller hidden layer. We observed no significant performance differences between the middle and larger sized layers, and the larger networks required more time to train. Thus, we used hidden layers with 30 units in the speech act classifiers and 50 units for the domain action and concept sequence classifiers for comparison with the other learning approaches.

	English Travel		German Travel		English Medical		German Medical	
	Input	**Output**	**Input**	**Output**	**Input**	**Output**	**Input**	**Output**
Domain Action	212	972	259	1001	252	462	297	286
Speech Act	212	70	259	70	252	50	297	43
Concept Sequence	282	615	329	638	302	305	340	179

Table 16: Sizes of input and output layers in neural networks for domain action classification experiments

4.2.2.4 Rainbow

The naïve Bayes n-gram classifiers tested in our domain action classification experiments were implemented using the Rainbow software package ([McCallum, 1996]). One of the primary parameters that can be varied in n-gram models is of course *n*. However, as mentioned in the previous section, in order to simulate the use of binary grammar label features used in the other learning approaches as closely as possible, we used unigram models whose vocabulary was the set of top-level grammar labels for the naïve Bayes classifiers. Furthermore, in the context of unigram models of grammar labels, options such as stop word lists and minimum word lengths were meaningless. Thus, for the first round of domain action classification experiments, varying the parameters of the Rainbow naïve Bayes classifiers was not necessary.

4.2.3 Results of Learning Approach Comparison

The results of the first round of experiments comparing the learning approaches on the domain action classification tasks are presented in this section. The mean accuracies of each learning approach on the tasks of domain action classification, speech act classification, and concept

sequence classification for each language-domain pair are reported along with any statistically significant differences among the classifiers within a particular language-domain pair. The classifiers used the parameter settings described in the previous section.

4.2.3.1 Domain Action Classifiers

	English Travel	German Travel	English Medical	German Medical
TiMBL	49.58%	46.51%	51.94%	51.66%
C4.5	48.90%	46.58%	52.08%	51.56%
SNNS	49.39%	46.21%	51.58%	50.17%
Rainbow	39.74%	38.32%	41.13%	45.73%

Table 17: Mean domain action classifier accuracies for all learning approaches over 20-fold cross-validation

Table 17 shows the average domain action classification accuracy for each learning approach on the 20-fold cross-validation setup. The differences in performance of the TiMBL, C4.5, and SNNS domain action classifiers and the Rainbow domain action classifier were highly significant across all language-domain pairs. For English Travel, the performance of the TiMBL ($t=2.57$, $p<0.02$) and SNNS ($t=2.58$, $p<0.02$) classifiers was significantly better than the C4.5 classifier. For German Medical, the performance of the TiMBL ($t=2.62$, $p<0.02$) and C4.5 ($t=2.54$, $p<0.02$) classifiers was significantly better than the SNNS classifier. None of the remaining performance differences shown in Table 17 were statistically significant.

4.2.3.2 Speech Act Classifiers

	English Travel	German Travel	English Medical	German Medical
TiMBL	69.86%	67.62%	77.81%	68.87%
C4.5	70.41%	67.90%	77.59%	70.52%
SNNS	71.91%	68.18%	78.41%	70.09%
Rainbow	51.39%	46.00%	72.33%	60.68%

Table 18: Mean speech act classifier accuracies for all learning approaches over 20-fold cross-validation

Table 18 shows the average speech act classification accuracy for each learning approach on the 20-fold cross-validation setup. The differences in performance of the TiMBL, C4.5, and SNNS speech act classifiers and the Rainbow speech act classifier were highly significant across all language-domain pairs. For English Travel, the performance of the SNNS classifier was significantly better than the TiMBL ($t=7.89$, $p<0.0001$) and C4.5 ($t=3.67$, $p<0.005$) classifiers. For German Medical, the performance of the C4.5 classifier was significantly better than the TiMBL classifier ($t=3.27$, $p<0.005$). None of the remaining performance differences shown in Table 18 were statistically significant.

4.2.3.3 Concept Sequence Classifiers

	English Travel	German Travel	English Medical	German Medical
TiMBL	69.22%	66.90%	64.57%	69.93%
C4.5	68.47%	66.45%	64.64%	69.98%
SNNS	71.35%	68.67%	61.65%	64.56%
Rainbow	51.64%	51.50%	47.32%	59.37%

Table 19: Mean concept sequence classifier accuracies for all learning approaches over 20-fold cross-validation

Table 19 shows the average concept sequence classification accuracy for each learning approach on the 20-fold cross-validation setup. The differences in performance of the TiMBL, C4.5, and SNNS concept sequence classifiers and the Rainbow concept sequence classifier were highly significant across all language-domain pairs. For English Travel, the performance of the SNNS classifier was significantly better than the performance of the TiMBL ($t=7.73$, $p<0.0001$) and C4.5 ($t=10.15$, $p<0.0001$) classifiers. Likewise for German Travel, the performance of the SNNS classifier was significantly better than the performance of the TiMBL ($t=5.49$, $p<0.0001$) and C4.5 ($t=9.07$, $p<0.0001$) classifiers. For English Medical, the performance of the TiMBL ($t=5.01$, $p<0.0001$) and C4.5 ($t=5.75$, $p<0.0001$) classifiers was significantly better than that of the SNNS classifier. For German Medical, the performance of the TiMBL ($t=9.31$, $p<0.0001$) and C4.5 ($t=8.10$, $p<0.0001$) classifiers was significantly better than the performance of the SNNS classifier. None of the remaining performance differences shown in Table 19 were statistically significant.

4.2.4 Additional Experiments with Rainbow Classifiers

The performance of the Rainbow classifiers was substantially worse than the performance of the other three learning approaches on all three classification tasks across all four language-domain pairs. However, the unigram model over top-level grammar labels in the argument parse did not really exploit the strengths of the naïve Bayes n-gram classification approach. Thus, we ran additional experiments to further test the performance of naïve Bayes n-gram models.

In the first additional experiment, we tested the use of higher order n-gram models of grammar labels rather than the unigram model used in the first set of experiments. With the exception of changing the value of *n*, the same training data, parameters, and 20-fold cross-validation setup were used as in the first experiment.

		English Travel	German Travel	English Medical	German Medical
Domain Action	**Accuracy**	40.62%	41.19%	42.36%	47.60%
	Change	+0.88%	+2.87%	+1.23%	+1.87%
Speech Act	**Accuracy**	53.30%	50.10%	73.53%	61.51%
	Change	+1.91%	+4.10%	+1.2%	+0.83%
Concept Sequence	**Accuracy**	53.48%	54.91%	49.37%	62.51%
	Change	+1.84%	+3.41%	+2.05%	+3.14%

Table 20: Mean accuracies of Rainbow grammar label bigram models for domain action, speech act, and concept sequence classification over 20-fold cross-validation

Table 20 shows the average accuracies of the grammar label bigram models for each classification task in each language and domain as well as the absolute improvement over the unigram grammar label model. Using a bigram model results in improved performance over the unigram model across all classification tasks, languages, and domains. We also tested trigram grammar label models, which produced additional small performance increases over the bigram models for most of the classifiers. However, the accuracy of the naïve Bayes classifiers using models based on grammar labels from the argument parse, even with bigram or trigram models, was still much lower than the accuracy of the other learning approaches.

Although the grammar labels from the argument parse provide an abstraction of the words present in a semantic dialogue unit, the words themselves are another source of features for the

classifiers. Cattoni *et al.* ([Cattoni *et al.*, 2001]) describe the application of word-based bigram models (based on utterances preprocessed to included semantic word classes relevant to the domain) to the task of domain action classification. Thus, we also conducted experiments in which the Rainbow naïve Bayes classifiers were trained on word bigrams. We tried using the default stop word list and stemming provided by the Rainbow software, but doing so produced much worse performance than using the raw text of the semantic dialogue unit. Thus, neither the stop word list nor stemming was used by the classifiers reported in Table 21.

	English Travel	**German Travel**	**English Medical**	**German Medical**
Domain Action	48.59%	48.09%	55.81%	56.06%
Speech Act	79.00%	77.46%	85.97%	81.34%
Concept Sequence	56.87%	57.77%	61.62%	65.48%

Table 21: Mean accuracies of Rainbow word bigram models for domain action, speech act, and concept sequence classification over 20-fold cross-validation

Table 21 shows the average accuracies of the Rainbow naïve Bayes domain action, speech act, and concept sequence classifiers based on word bigram models for each language-domain pair. The use of word bigram models resulted in highly significant performance gains over the Rainbow classifiers based on grammar label n-grams across all classification tasks and language-domain pairs. Furthermore, the use of word bigram models greatly improved the performance of the Rainbow naïve Bayes classifiers compared to the other learning approaches. The most remarkable improvement in accuracy was seen on speech act classification. The Rainbow word bigram speech act classifiers substantially outperformed the TiMBL, C4.5, and SNNS speech act classifiers across all language-domain pairs, producing absolute improvements in accuracy from 7.1% to 10.8% over the best classifiers shown in Table 18. The word bigram models also outperformed the other learning approaches on the domain action classification task for three of the four language-domain pairs, resulting in absolute improvements of 1.5% to 4.4% over the best classifiers shown in Table 17. The improvements over the other three learning approaches were significant for German Travel (at least $p<0.005$), English Medical (at least $p<0.0001$), and German Medical (at least $p<0.0005$). For English Travel, the Rainbow word bigram domain

action classifier did not quite reach the accuracies of the other three learning approaches. However, the differences between the Rainbow classifier and the other three learning approaches were not statistically significant. Although the word bigram models for concept sequence classification achieved better performance than the naïve Bayes classifiers based on grammar labels, their performance was still much weaker than that of the best concept sequence classifiers shown in Table 19.

4.2.5 Adding Word-Based Features to the TiMBL classifiers

The performance of the Rainbow naïve Bayes word bigram speech act and domain action classifiers and the improvement of the word bigram models over the grammar label models for the Rainbow concept sequence classifiers clearly indicated the usefulness of word-based features for domain action classification tasks. However, the performance of the classifiers for the other three learning approaches that used only features based on grammar labels in the argument parse was also good. Thus, we conducted additional experiments to examine the effects of combining grammar label information from the argument parse with word information on domain action, speech act, and concept sequence classification performance. The same training data and 20-fold cross-validation setup that were used in the previous experiments were used in these experiments. Given the similarity in the accuracies of the TiMBL, C4.5, and SNNS classifiers and the fact that none of the learning approaches clearly outperformed the others across classification tasks, languages, and domains, we chose to use only the TiMBL classifiers for all further experiments. Additional discussion of the reasons behind this choice is included in Section 4.2.6.

We tested two approaches to adding word-based information to the TiMBL classifiers. In both approaches, the word-based information for each cross-validation fold was computed based only on the data in the training set. In the first approach, information about the words in a semantic dialogue unit was included directly in the TiMBL classifiers. The second approach incorporated word-based information indirectly, using the output of Rainbow word bigram classifiers as input for the TiMBL classifiers.

4.2.5.1 Adding Words Directly to the TiMBL Classifiers

We first tested the addition of word information directly into the input feature vector for the TiMBL classifier. The input feature vector was expanded to include a set of binary features for

words in the semantic dialogue unit under consideration. The binary word features were similar to the features used for grammar labels from the argument parse, and each feature indicated the presence (+) or absence (-) of the corresponding word in the semantic dialogue unit. This approach corresponds to the inclusion of feature set 3 listed in Figure 23.

The first step in this approach to adding word information to the classifiers was to determine which words to include in the input vector. The previous experiments represent one extreme in which no word features are included. The opposite extreme would be to include a feature for every word in the vocabulary of the training data. In order to determine the specific words to include in the input feature vector, we sorted the words in the training data based on their mutual information with the class variable using the Rainbow software. Then input features for the TiMBL classifiers were included for only the top *n* words in the sorted list for various values of *n*. In order to test the effects of adding word information, we tested classifiers that included features for the top 10, 25, 50, 100, 250, 500, and 1000 words as well as the entire vocabulary. Because the word features that were added to the input feature vector were similar to the binary features used to represent the grammar labels in the argument parse, the same TiMBL parameter settings that were used in the previous experiments were also used in this experiment. Thus, all of the classifiers used k=1 with Inverse Linear vote weighting and Gain Ratio feature weighting.

Word Features	English Travel	German Travel	English Medical	German Medical
0	49.58%	46.51%	51.94%	51.66%
10	50.22%	48.56%	56.33%	54.01%
25	50.61%	49.49%	59.58%	55.14%
50	53.02%	51.92%	59.71%	55.88%
100	54.48%	53.23%	61.52%	58.33%
250	56.62%	54.44%	62.14%	59.59%
500	56.28%	54.85%	62.36%	59.85%
1000	56.86%	55.53%	62.80%	60.25%
All Words	56.82%	54.96%	62.34%	60.08%

Table 22: Domain action classification accuracy for TiMBL classifiers with binary word features included

Table 22 summarizes the mean classification accuracies of the TiMBL domain action classifiers with binary word features included for all language-domain pairs. The *Word Features* column shows the number of word features that were used in each classifier. The row for 0 word features shows the accuracy of the classifiers from the previous experiments that used only grammar label features. As expected based on the results of the experiments with the Rainbow word bigram naïve Bayes classifiers, the addition of word features to the TiMBL domain action classifiers resulted in improved accuracy compared to the classifiers without word features. Even the addition of a small number of words improved the performance of the classifiers. The addition of as few as the top 10 words (sorted by mutual information with the class variable) resulted in a significant improvement in accuracy for German Travel ($t=8.51$, $p<0.0001$), English Medical ($t=9.12$, $p<0.0001$), and German Medical ($t=3.47$, $p<0.005$). For English Travel, the improvement in accuracy achieved by adding word features became significant with 50 words ($t=8.18$, $p<0.0001$). The performance of the domain action classifiers generally continued to improve as more word features were added, leveling off around 250 or 500 words with absolute improvements in performance in the range of 7-10% over the classifiers with no word features. Although improvements in accuracy were achieved with the addition of more words, the performance differences beyond 250 words were generally not significant. The only exception was for the German Travel classifier with 1000 words. Additionally, including all of the words in the vocabulary never produced the maximum accuracy.

Word Features	English Travel	German Travel	English Medical	German Medical
0	14	16	9	6
10	16	18	10	7
25	18	20	11	8
50	21	23	12	9
100	26	29	14	10
250	41	45	22	15
500	67	72	35	22
1000	114	120	59	39
All Words	213	311	106	43

Table 23: Memory required (MB) to run the domain action classifiers with binary word features

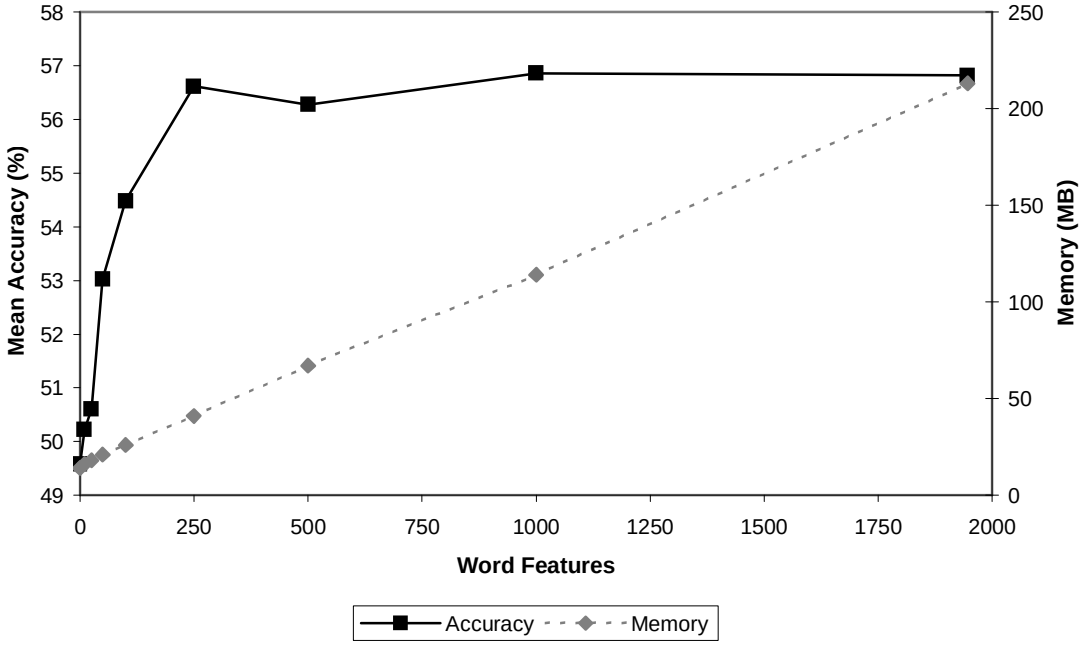

Figure 33: Domain action classification accuracy and memory requirements using grammar label and word features for English Travel data

In addition to the accuracy of the classifiers, the memory required to run the classifiers is a factor that can help determine an appropriate number of words to include in the classifiers. Table 23 shows the memory required to run the classifier for the first fold in each cross-validation experiment, and Figure 33 illustrates the tradeoff between accuracy and memory for the English Travel classifiers. The solid line and left y-axis in Figure 33 illustrate the mean accuracy over the 20 folds of the cross-validation for the English Travel domain action classifier as the number of words included was increased. The memory required for running the domain action classifier used in the first fold of each experiment as the number of word features was increased is also shown in the figure by the dashed line and the right y-axis. As the figure illustrates, the memory requirements increased roughly linearly with the number of words features included. We observed the same general trend for other language-domain pairs, although the specific memory requirements varied depending on the size of the example base. Since adding more than 250 to 500 words did not generally lead to significant improvements in accuracy but did lead to large increases in the amount of memory required, it would make sense to limit the number of word

features included in any classifiers used in an online system, especially if memory resources were limited.

Word Features	English Travel	German Travel	English Medical	German Medical
0	69.86%	67.62%	77.81%	68.87%
10	72.90%	70.32%	82.40%	72.10%
25	73.99%	73.36%	83.90%	73.98%
50	76.70%	74.81%	85.15%	76.37%
100	77.53%	74.94%	85.48%	77.12%
250	79.03%	76.85%	85.97%	77.64%
500	79.09%	77.25%	86.27%	78.42%
1000	79.80%	77.29%	86.27%	79.03%
All Words	79.02%	77.16%	86.35%	78.95%

Table 24: Speech act classification accuracy for TiMBL classifiers with binary word features included

Table 24 summarizes the mean accuracies of the speech act classifiers with binary word features included over the 20 cross-validation folds for all language-domain pairs. The addition of word features to the speech act classifiers produced performance improvements similar to those seen for the domain action classifiers. The addition of as few as 10 word features led to significant improvements in performance across all language-domain pairs (English Travel: $t=8.46$, $p<0.0001$; German Travel: $t=9.80$, $p<0.0001$; English Medical: $t=10.93$, $p<0.0001$; German Travel: $t=4.82$, $p<0.0005$). As in the case of the domain action classifiers, performance continued to increase as more words were added and leveled off around 250 or 500 words resulting in absolute increases in accuracy of 8-9%. Again, the performance improvements with more than 250 words were not significant (with the exception of the English Travel classifier with 1000 words). For the speech act classifiers, we also observed trends for the tradeoff between performance improvements and increases in memory requirements similar to those seen for the domain action classifiers.

Word Features	English Travel	German Travel	English Medical	German Medical
0	9	8	5	4
10	11	9	6	4
25	13	12	7	5
50	15	14	8	6
100	19	18	10	7
250	31	29	15	10
500	53	49	24	16
1000	85	84	49	32
All Words	175	251	95	35

Table 25: Memory required (MB) to run the speech act classifiers with binary word features

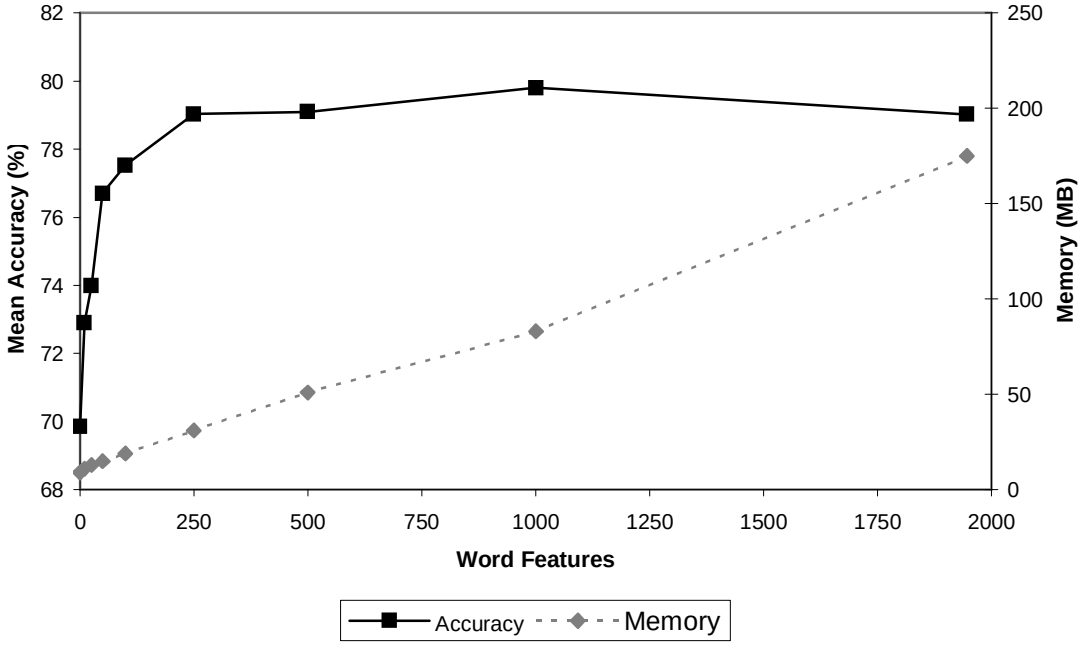

Figure 34: Speech act classification accuracy and memory requirements using grammar label and word features for English Travel data

Table 25 shows the memory required to run the classifier for the first fold in each cross-validation experiment, and Figure 34 illustrates the changes in accuracy and memory requirements as the number of word features was increased for the English Travel speech act

classifier. Although the specific values of accuracy and memory size differed, the trends were similar to those seen for the domain action classifiers.

Word Features	English Travel	German Travel	English Medical	German Medical
0	69.22%	66.90%	64.57%	69.93%
10	68.04%	67.44%	67.44%	69.45%
25	68.49%	67.24%	67.11%	69.01%
50	68.60%	67.40%	67.14%	69.70%
100	68.32%	67.26%	67.17%	70.32%
250	68.51%	67.32%	67.96%	70.62%
500	68.52%	67.65%	68.26%	71.19%
1000	68.02%	67.43%	68.64%	71.32%
All Words	67.97%	67.08%	67.58%	71.19%

Table 26: Concept sequence classification accuracy for TiMBL classifiers with binary word features included

Table 26 summarizes the mean accuracies of the concept sequence classifiers with binary word features included over the 20-fold cross-validation for all language-domain pairs. Unlike domain action classification and speech act classification, the addition of word features to the concept sequence classifier gave mixed results. For English Travel, the addition of any word features resulted in small decreases in accuracy compared to the classifier with no word features. The decreases in accuracy were not significant for 25, 50, 100, 250, and 500 words. For German Travel, adding word features resulted in small increases in accuracy that were not significant compared the use of no word features. For the German Medical classifiers, adding word features first led to small decreases in accuracy and then to small increases in accuracy. Only the English Medical classifier exhibited a substantial improvement when words were added to the input feature vector. The addition of 10 words resulted in a significant improvement over the classifier without words ($t=5.94$, $p<0.0001$). All of the classifiers with more words were also significantly better than the classifier with no words, but none of them were significantly better than the classifier with 10 words. We observed trends for memory use similar to the trends for the domain action and speech act classifiers. Table 27 lists the memory requirements for the concept sequence classifiers.

Word Features	English Travel	German Travel	English Medical	German Medical
0	11	13	7	5
10	13	15	8	6
25	15	17	9	6
50	18	20	10	7
100	23	25	13	8
250	35	38	20	12
500	57	60	31	18
1000	95	99	52	34
All Words	182	266	99	38

Table 27: Memory required (MB) to run the concept sequence classifiers with binary word features

4.2.5.2 Adding Rainbow Output to the TiMBL Classifiers

In the experiments described in the preceding section, information about the words present in a semantic dialogue unit was incorporated directly into the input feature vector for the TiMBL classifiers. We now describe an alternative approach for including word information more indirectly in the input feature vectors for the TiMBL classifiers. Rather than defining features for specific words in the semantic dialogue unit, features were included for the probabilities computed by the Rainbow naïve Bayes word bigram models for each class. In addition to the binary features for grammar labels in the argument parse, one input feature was added to the input vector of the TiMBL classifiers for each possible class. The value of each feature was the probability of the corresponding class computed by the Rainbow word bigram classifier. For example, for the English Travel speech act classifier, 70 features were added, and the value of each feature was the probability of the associated speech act computed by the Rainbow word bigram speech act classifier for English Travel.

We tested the performance of this approach for all language-domain pairs. The same 20-fold cross-validation setup used in previous experiments was used again. The data in the test set for each fold was also heldout from the training data for the Rainbow word bigram classifiers. All of the TiMBL classifiers tested used Inverse Linear vote weighting and Gain Ratio feature weighting. The Numeric distance metric was used for the probability features, and the Overlap

distance metric was used for the grammar label features as in previous experiments. Because we added Numeric features to the TiMBL classifiers, we tested with several values of *k* for each task.

	English Travel	German Travel	English Medical	German Medical
Without Probability Features	49.58%	46.51%	51.94%	51.66%
With Probability Features	??.??% (> 1500 MB)	??.??% (>1500 MB)	60.07% (389 MB)	58.90% (121 MB)

Table 28: Domain action classification accuracy and memory requirements for the TiMBL classifiers with probability features

Table 28 shows the mean accuracies of the TiMBL domain action classifiers that included features for the probability of each domain action computed by the Rainbow word bigram models. The memory required to run the classifiers is also included in the table. The value of *k* was set to 1 for the domain action classifiers with probability features. The accuracies of the TiMBL classifiers that included no word information are listed in the first row of the table for reference. Probability features were included for each domain action in the training corpus for each language and domain. Performance figures are not included for English Travel and German Travel because the classifiers required too much memory (> 1.5 GB) to run on the machines available. When the amount of memory required exceeded the amount of physical memory available on the machine, the classifier for the first fold ran for more than a day and never finished loading the example base. Thus, domain action classification performance results were only obtained for the English and German Medical classifiers, which had smaller training data sets with fewer domain actions and thus smaller example bases. For both classifiers, the improvement in accuracy over the classifiers without words was highly significant.

The mean accuracies over the 20 cross-validation sets of the TiMBL speech act classifiers that included features for the probabilities computed by the Rainbow word bigram speech act models are shown in Table 29. The speech act classifiers with probability features reported in the table used a value of 11 for *k*. The accuracies of the TiMBL classifiers with no word-based features are also shown in the table for reference. The input feature vector for the speech act

classifiers included probability features for each speech act in the training corpus for each language and domain. Again, as with the domain action classifiers, the inclusion of the probability features resulted in highly significant improvements in accuracy over the classifiers without word-based features.

	English Travel	German Travel	English Medical	German Medical
Without Probability Features	69.86%	67.62%	77.81%	68.87%
With Probability Features	81.18% (148 MB)	78.78% (153 MB)	87.04% (62 MB)	81.82% (29 MB)

Table 29: Speech act classification accuracy for the TiMBL classifiers with probability features

	English Travel	German Travel	English Medical	German Medical
Without Probability Features	69.22%	66.90%	64.57%	69.93%
With Full Concept Sequence Probability Features	68.19% (947 MB)	67.17% (1020 MB)	67.55% (279 MB)	70.71% (84 MB)
With Individual Concept Probability Features	69.16% (240 MB)	68.12% (256 MB)	67.41% (98 MB)	71.10% (41 MB)

Table 30: Concept sequence classification accuracy for the TiMBL classifiers with probability features

Table 30 shows the mean concept sequence classification accuracies over the 20-fold cross-validation of the classifiers that included features for the probabilities computed by the Rainbow word bigram classifiers. The accuracies of the TiMBL classifiers without any word information are also shown. For the concept sequence classifiers, we tested two methods for including probabilities based on Rainbow word bigram models in the input feature vector. The first approach was to add one probability feature for each full concept sequence in the training data for each language and domain to the input vector. This approach made direct use of the Rainbow concept sequence classifiers described in Section 4.2.4. The accuracies of the TiMBL classifiers that used this set of probability features are shown in the row labeled *With Full Concept Sequence Probability Features* in Table 30. The classifiers reported in the table used a

value of 9 for *k*. The second approach took advantage of the fact that a concept sequence can be broken down into individual concepts. In this approach, naïve Bayes word bigram models were trained (using Rainbow) for each individual concept rather than for each observed concept sequence. Then a feature for the probability of each individual concept was included in the input feature vector of the TiMBL classifier. The performance of the classifiers that used the second approach is shown in the row labeled *With Individual Concept Probability Features* in Table 30. The classifiers reported in the table used a value of 7 for *k*.

As Table 30 shows, the concept sequence classifiers that used features for individual concept probabilities generally outperformed the classifiers that included features for full concept sequence probabilities. The difference was significant for English Travel ($t=2.98$, $p<0.01$) and German Travel ($t=3.63$, $p<0.005$). For German Medical, the classifier with individual concept probabilities also outperformed the classifier with full concept sequence probabilities, but the difference was not significant. The full concept sequence classifier only outperformed the individual concept classifier for English Medical, and the difference in accuracies was very small and not significant. Based on these results, it appears that the use of individual concept probabilities should be preferred over the use of full concept sequence probabilities. In addition to providing better performance, the individual concept classifiers also have the advantage that the set of concepts defined by the Interchange Format specification and encountered in the training data is much smaller than the set of full concept sequences. This should reduce data sparseness for the word bigram models since a single concept sequence can be used to train multiple individual concept models. Additionally, the amount of memory required to run the classifiers that incorporated individual concept probabilities was much smaller than that required for the classifiers that used full concept sequence probabilities.

The concept sequence classifiers that used individual concept probabilities also generally provided small improvements over the classifiers that used no word information. The accuracy of the classifiers that included probabilities for individual concepts was significantly better than the accuracy without word features for German Travel ($t=6.17$, $p<0.0001$) and English Medical ($t=6.35$, $p<0.0001$). For German Medical, the addition of individual concept probabilities also resulted in a small improvement in accuracy over the classifier with no word features, but the difference was not significant. For English Travel, the accuracy of the classifier with no word-

based features was slightly better than that of the classifier with individual concept probability features, but the difference was not significant.

4.2.5.3 Comparison of Classifiers that Include Word Information

In this section, we summarize and compare the performance of the three different approaches to using information about the words in a semantic dialogue unit in the domain action, speech act, and concept sequence classifiers. The first approach used simple naïve Bayes classifiers with word bigram models (created using the Rainbow software) and will be labeled as the *Rainbow Word Bigram* classifier in the tables in this section. The second approach added binary features for the words with the highest mutual information with the class variable to the TiMBL classifiers. This approach will be referred to as *TiMBL+Words*. Additionally, although the differences between the accuracies of the TiMBL+Words classifiers for more than 250 words generally were not significant, the accuracies reported in the tables in this section will use the highest accuracy achieved with any number of words. The number of words included for each classifier will be shown in the tables along with the accuracy of the classifier. The third approach added features for probabilities computed by naïve Bayes word bigram models and will be called *TiMBL+Probability*.

	English Travel	**German Travel**	**English Medical**	**German Medical**
Rainbow Word Bigram	48.59%	48.09%	55.81%	56.06%
TiMBL+Words	56.86% (1000)	55.53% (1000)	62.80% (1000)	60.25% (1000)
TiMBL+Probability	--	--	60.07%	58.90%

Table 31: Summary of domain action classification accuracies for classifiers with word information

Table 31 recaps the performance of the domain action classifiers that made used of word information. As the table shows, for the task of domain action classification, both the TiMBL+Words classifiers and the TiMBL+Probability classifiers clearly outperformed the Rainbow Word Bigram classifiers. Furthermore, the TiMBL+Words classifiers outperformed the TiMBL+Probability classifiers for the Medical Assistance domain. The difference was

significant for English (t=4.28, p<0.0005) but not for German. As mentioned previously, we were unable to run the TiMBL+Probability classifiers for the Travel & Tourism domain due to memory limitations.

	English Travel	German Travel	English Medical	German Medical
Rainbow Word Bigram	79.00%	77.46%	85.97%	81.34%
TiMBL+Words	79.80% (1000)	77.29% (1000)	86.27% (1000)	79.03% (1000)
TiMBL+Probability	81.18%	78.78%	87.04%	81.82%

Table 32: Summary of speech act classification accuracies for classifiers with word information

Table 32 contains a summary of the performance for the three different types of speech act classifiers that included word information. The TiMBL+Words classifiers reached a level of accuracy roughly equivalent to that of the Rainbow Word Bigram classifiers. For English Travel, the TiMBL+Words classifier outperformed the Rainbow Word Bigram classifier, and the difference was marginally significant (t=2.67, p<0.02). The differences between the TiMBL+Words classifier and the Rainbow Word Bigram classifier for the remaining language-domain pairs were not significant. For speech act classification, the TiMBL+Probability classifier performed better than the other two classifiers across all language-domain pairs. The difference between the TiMBL+Probability classifier and the Rainbow Word Bigram classifier was significant for English Travel (t=9.10, p<0.0001) and German Travel (t=7.04, p<0.0001). The difference between the TiMBL+Probability classifier and the TiMBL+Words classifier was significant for English Travel (t=4.98, p<0.0001), German Travel (t=4.07, p<0.001), and German Medical (t=3.46, p<0.005).

Table 33 summarizes the concept sequence classification performance for the classifiers that made use of word information. For the TiMBL+Probability classifiers, the performance of the classifiers that included probabilities for individual concepts is shown. The TiMBL+Words and TiMBL+Probability classifiers clearly outperformed the Rainbow Word Bigram classifiers across all language-domain pairs. Furthermore, there were no significant performance differences between the TiMBL+Words and TiMBL+Probability concept sequence classifiers.

	English Travel	German Travel	English Medical	German Medical
Rainbow Word Bigram	56.87%	57.77%	61.62%	65.48%
TiMBL+Words	68.52% (500)	67.65% (500)	68.64% (1000)	71.32% (1000)
TiMBL+Probability (Individual Concepts)	69.16%	68.12%	67.41%	71.10%

Table 33: Summary of concept sequence classification accuracies for classifiers with word information

4.2.6 Discussion

One of the main goals of these experiments was to test the feasibility of automatically classifying domain actions in the NESPOLE! domains using the results of argument parsing as input and to compare the performance of several different machine learning techniques on the task of domain action classification. Domain action classification is an especially challenging problem with as many as 1000 classes found in our training data. Even when the task is divided into subproblems of speech act classification and concept sequence classification, the subtasks remain quite difficult. The difficulty is compounded by relatively sparse training data with unevenly distributed classes. Although the most common classes in our training corpus had over 1000 training examples, many of the classes had only 1 or 2 examples.

Despite these difficulties, our results indicate that domain action classification is quite feasible. For speech act classification in particular we were able to achieve very strong performance. Although performance on concept sequence classification and domain action classification was not as high, it was still quite strong. This is especially true given that there was roughly an order of magnitude more classes for those tasks than for speech act classification. Based on our experiments, it appears that all of the learning approaches we tested were able to cope reasonably well with data sparseness at the level found in our data.

Another point worth noting is that our experiments with word information included in the classifier input provide evidence that domain action classification could be performed reasonably well using only word-based information without any information from the argument parse. Of course, some form of argument parsing would still be required in order to perform translation using the Interchange Format interlingua. Although our best-performing classifiers combined

word and argument parse information, the Rainbow naïve Bayes word bigram classifiers performed especially well on the speech act classification task. Furthermore, the Rainbow word bigram domain action classifiers actually outperformed all of the domain action classifiers that used only grammar labels from the argument parse as input. With additional training data, the performance of the concept sequence and domain action word bigram classifiers could be expected to improve. Also, as mentioned previously, very little preprocessing was performed on the text of the semantic dialogue units that was used to train the word bigram models in our experiments. It is likely that some performance improvements could be achieved by performing some simple preprocessing of the semantic dialogue unit text similar to that used in the classifiers described in [Cattoni *et al.*, 2001]. In the bigram models used in that work, the input text was normalized by grouping together words and phrases into semantic categories relevant to the domain.

Despite the success of the naïve Bayes word bigram classifiers, the results of our experiments demonstrate that information from the argument parse is also useful for identifying the domain action. For the task of full domain action classification, the performance of the classifiers that combined information from the argument parse with word information clearly produced superior performance over classifiers that used only one type of information. The advantage of combining information was less extreme for the speech act classification and concept sequence classification tasks. For speech act classification, the Rainbow classifier that used only a simple word bigram model generally performed nearly as well as the best classifiers that combined both types of information. On the other hand, for concept sequence classification, the TiMBL classifiers that used only information from the argument parse generally performed nearly as well as the classifiers that used both argument parse and word information.

It appears from these results that information about the words in a semantic dialogue unit is especially important for determining the speech act and information about the arguments is especially useful for determining the concept sequence. Considering the roles of the speech act and concept sequence in the Interchange Format representation, this observation makes sense. The concepts in a domain action represent the semantic focus of a semantic dialogue unit. Because the main role of the arguments is to encode detailed semantic information, it is not surprising that they would be most useful for identifying the concepts. The speech act captures the general intention of the speaker conveyed by a semantic dialogue unit, and it may be possible

to express several intentions regarding the same set of arguments. In such cases, it would often be the words and phrases surrounding the arguments that would provide the information necessary to identify speech act. These factors also help to explain why the domain action classifiers that used both types of information were particularly successful.

Another goal of our experiments was to help in the selection of a machine learning approach to be used in the hybrid analyzer. Certainly one of the most important considerations is how well the learning approach performs the classification task. However, the performance of the classifiers is not the only consideration to be made in selecting the classifier for our hybrid analyzer. Several additional factors are also important in selecting the particular machine learning approach to be used. One important attribute of the learning approach is the speed of both classification and training. Because the classifiers are part of a translation system designed for use between two humans to facilitate (near) real-time communication, the domain action classifiers must classify individual utterances online very quickly. Furthermore, since humans must write and test the argument grammars, training and batch classification must also be fast in order to allow the grammar writers to update the grammars, retrain the classifiers, and test the resulting hybrid analyzer efficiently. All four of the learning approaches that we tested met the requirement of fast classification of individual instances, classifying from several inputs per second to hundreds of inputs per second. Additionally, the TiMBL and Rainbow software packages provide server modes for online classification of single instances, whereas classification must be run in an offline batch mode when using the C4.5 and SNNS software. One of the primary disadvantages of the neural network approach was the time required for training. While the other three approaches could be trained in seconds to minutes, the neural network approach generally took hours to days. This is clearly undesirable for iterative testing by grammar writers.

Another important property of the machine learning approaches used for domain action classification is the ability to easily accommodate both continuous and discrete features from a variety of sources in the input feature set. As described in Section 2.5.2, input features may be based on words and/or phrases in a semantic dialogue unit, information from the argument parse, the Interchange Format representation of the arguments, and properties of the dialogue (e.g. speaker tag). Although the use of contextual information for domain action classification was not explored in this dissertation, it would also be possible to extract features from previous semantic

dialogue units or speaker turns. Thus, the learning approach used for any of the domain action classification tasks should be able to easily combine features from any or all of these sources. The combination of continuous and discrete features from different sources is exemplified by the combination of binary grammar features and probabilities based on word bigrams in the TiMBL+Probability classifiers described in Section 4.2.5.2. Among the approaches that were tested, the memory-based learning approach and the decision tree approach most easily accommodated the largest variety of features. For example, the TiMBL+Probability concept sequence classifiers were able to very easily represent binary discrete features (grammar labels from the argument parse), multi-valued discrete features (the speech act), and continuous numerical features (concept probabilities). Representing binary and continuous features in a neural network approach is clearly very easy. Representing multi-valued discrete features in neural networks is also possible but somewhat less convenient, requiring, for example, a set of input units for each feature with one unit per value of the feature. The lack of a mechanism for including different types of features from multiple sources is perhaps the biggest weakness of the naïve Bayes n-gram approach.

The ability to produce a ranked list of possible classes is another desirable aspect of the machine learning approach used for the domain action classification tasks. As described in Section 1.5.4, the Interchange Format specification defines how speech acts and concepts are allowed to combine as well as how arguments are licensed by the components of the domain action. These constraints can be used to select an alternative domain action if the best domain action from classification violates the specification in some way. This property is one of the clear strengths of the naïve Bayes n-gram approach, which ranks all of the possible classes by their probability given the input, and the neural network approach, for which a ranked list of classes can be created based on the activation levels of output units. The main disadvantage of the memory-based learning and decision tree approaches is their inability to produce a ranked list of all possible classes. However, both approaches may produce a small ranked subset of classes based on the distribution of the classes in the nearest neighbor set or leaf node from which the best class was determined. Furthermore, as described in Section 4.6.1, a list of alternative classes can be extracted from the training data and ranked based on frequency in the data as a backup in case the small ranked set from the classifier is not sufficient to find an alternative domain action. Although the order of the alternatives may (or may not) be worse than the order produced by a

naïve Bayes or neural network classifier, all of the alternatives would be present in the list and could thus be found during fallback. Furthermore, the list could be computed offline and would thus not require computation at run time.

Based on all of these considerations, memory-based learners implemented using TiMBL appear to be the best choice for use as the primary classifiers in the online hybrid analyzer. Classifiers implemented using the TiMBL software meet all of the requirements discussed above except the ability to produce a complete ranked list of the alternative classes for each input instance. However, such a list can be produced from the training data if necessary. Adding new features to TiMBL classifiers would also be very easy. Thus, both the TiMBL+Words classifiers, which incorporated word information directly into the input feature vector with the argument parse features, and the TiMBL+Probability classifiers, which combined argument features with probabilities based on word bigram models, seem like an excellent choices for the domain action classification tasks. The TiMBL+Words classifiers outperformed the TiMBL+Probability classifiers for domain action classification (at least for the Medical domain), but the TiMBL+Probability classifiers were better for speech act classification. For concept sequence classification, the classifiers were essentially equivalent.

The TiMBL+Probability classifiers can directly address the only requirement that the use of TiMBL classifiers fails since the Rainbow word bigram classifiers produce a ranked list of alternative classes. This is certainly true for the speech act classifiers. However, as shown in Section 4.2.5.2, the TiMBL+Probability concept sequence classifiers that used individual concept probabilities performed better than the classifiers that used full concept sequence probabilities. If individual concept sequence probabilities are used, a ranked list of alternative classes (i.e. concept sequences) is not produced by the Rainbow classifiers. Thus, another approach for producing the ranked list of alternative classes would still be required. Of course, all of the TiMBL+Words classifiers would require that the ranked list of alternative classes be produced based on the training data.

The main drawback of the TiMBL+Probability classifiers was the memory that they required. The TiMBL+Probability classifiers generally required much more memory than the TiMBL+Words classifiers. For the Travel domain, for which the training data size and number of classes were very similar for English and German, the TiMBL+Words classifiers with 1000 words required less than 20% of the memory of the TiMBL+Probability classifiers for the

domain action task, about 55% for speech act classification, and about 40% for concept sequence classification (compared to the classifiers that used individual concept probabilities). In addition, the TiMBL+Words classifiers have the advantage that the performance-memory tradeoff can be controlled directly by changing the number of words included in the classifiers. Although the maximum accuracy was generally achieved with 1000 words, there were also generally no significant differences in accuracy from at least 250 words up to 1000 words. If a smaller number of words were included, the memory advantage of the TiMBL+Words classifiers over the TiMBL+Probability classifiers would have been even larger. Finally, the TiMBL+Probability approach also has two further disadvantages. First, two classifiers (TiMBL and Rainbow) must be run for each classification task, and second, the number of probability features may increase as the size of the database increases and new classes are encountered, especially for domain actions and concept sequences.

In summary, our experiments demonstrate that it is useful to combine information about the argument parse and the words in a semantic dialogue unit for domain action classification and its subtasks of speech act classification and concept sequence classification. Based on the fact that performance across learning approaches was similar and because of the useful properties of memory-based learning as implemented in the TiMBL software, we chose to use TiMBL to implement the classifiers in our hybrid analyzer. Both the TiMBL+Words approach of adding words directly to the input feature vectors and the TiMBL+Probability approach of adding probabilities based on word bigram models offer advantages for the domain action classification tasks. The main advantages of the TiMBL+Probability approach are better accuracy (at least for speech act classification) and the availability of a ranked list of alternatives tailored to each semantic dialogue unit. On the other hand, the TiMBL+Words classifiers generally provide comparable performance with much smaller memory requirements that can be adjusted based on the task and resources available. Furthermore, the TiMBL+Words approach requires only a single classifier for each task and provides a fixed feature set size regardless of the size of the training corpus. Based on these considerations, we chose to use only TiMBL without probabilities computed by Rainbow in the online hybrid analyzers that are included in the NESPOLE! translation servers.

4.3 Effects of Non-Task-Specific Training Data

The NESPOLE! translation system supports dialogues between an Italian-speaking agent (i.e. an agent at a tourism office or a doctor) and an English-, German-, or French-speaking client (i.e. a traveler or a patient). Although the translation techniques used in the NESPOLE! system are certainly not specific to this particular scenario, the data available for developing the system was collected with these scenarios in mind. Within the context of the NESPOLE! translation system, the hybrid analysis approach described in this dissertation is used as the analysis module for the English and German translation servers. Thus, the specific task of the English and German domain action classification modules in the NESPOLE! system is to identify the domain actions of semantic dialogue units spoken by a client in one of the NESPOLE! domains. However, as mentioned in Section 4.1, the data used to train the classifiers was supplemented with non-task-specific data. In addition to client-side utterances from the NESPOLE! domain that were originally spoken in the source language, the training data included agent-side utterances, utterances that were originally spoken in another language and manually translated into the source language, and utterances from different but related domains (C-STAR II Travel Planning and Babylon Medical). Thus, we conducted experiments to determine the effects of including such non-task-specific data on the accuracy of the classifiers on task-specific semantic dialogue units.

All of the experiments were run using the same training corpora as in the experiments described in Section 4.2. Additionally, the same memory-based TiMBL classifiers described in Section 4.2.2 and used in the first round of experiments for comparing learning approaches were used in all of these experiments. Thus, input feature vectors for the domain action and speech act classifiers included binary features for the root labels of the parse trees in the argument parse. Likewise, the concept sequence classifiers used the same features plus a single feature for the speech act assigned to the semantic dialogue unit.

In each experiment described in this section, we ran 20-fold cross-validation experiments with two conditions to determine the effects of the various types of supplemental data. In the first condition, supplemental data of a particular type was removed from the training data set. The remaining data was then randomly split into 20 folds. Each fold was heldout as a test set, and the remaining 19 folds were used for training. In the second condition, the same random split of the non-supplemental data was used, but the supplemental data was added to the training set for each

fold. Because the same test sets were used in both conditions, we tested for significance using two-tailed matched pair t-tests.

4.3.1 Using Non-Domain-Specific Data

	English Travel		German Travel		English Medical	
	In	**Out**	**In**	**Out**	**In**	**Out**
Semantic Dialogue Units	5667	2622	6111	2608	1578	2078
Domain Actions	716	453	749	448	308	255
Speech Acts	62	48	63	48	40	30
Concept Sequences	474	282	498	279	204	183

Table 34: Corpus contents for in-domain data and out-of-domain data

We first examined the effects of including out-of-domain data on the classification accuracy of in-domain semantic dialogue units. For the Travel domain, the training data was first split into a set containing only NESPOLE! data and a set containing only C-STAR II data. For the English Medical domain, the supplemental Babylon data was separated from the NESPOLE! data. The German Medical domain was not included in this experiment because there was no out-of-domain data available. Table 34 contains details about the contents of the in-domain (*In*) and out-of-domain (*Out*) data sets after the corpora were separated.

For each language-domain pair, the NESPOLE! data (*In*) was randomly split into 20 folds that were used as test sets for two experimental conditions. In the first condition, each in-domain fold was used as a test set, and the classifiers were trained on only the 19 remaining in-domain training subsets. In the second condition, the out-of-domain data (*Out*) was added to the training set for each in-domain test set.

	English Travel	German Travel	English Medical
Without Out-of-Domain Data	48.33%	47.18%	38.47%
With Out-of-Domain Data	48.37%	47.24%	37.39%

Table 35: Domain action classification accuracy on in-domain data with and without out-of-domain training data

	English Travel	German Travel	English Medical
Without Out-of-Domain Data	67.85%	66.21%	62.42%
With Out-of-Domain Data	68.50%	66.86%	64.69%

Table 36: Speech act classification accuracy on in-domain data with and without out-of-domain training data

	English Travel	German Travel	English Medical
Without Out-of-Domain Data	69.68%	68.75%	60.02%
With Out-of-Domain Data	69.91%	68.76%	60.02%

Table 37: Concept sequence classification accuracy on in-domain data with and without out-of-domain training data

Table 35, Table 36, and Table 37 show the mean classification accuracies over the 20 folds of in-domain test sets for domain actions, speech acts, and concept sequences respectively. For domain action classification, we observed very small increases in accuracy for English and German Travel and a decrease for English Medical. None of the differences were statistically significant. Similar results were obtained for concept sequence classification. A small increase in accuracy was seen for English Travel, and there was virtually no change for German Travel and English Medical. Again, none of the differences were significant. The effects of out-of-domain data on speech act classification were somewhat different. For the speech act classifiers, we observed increases in accuracy across the board. The improvements were significant for English Travel ($t=2.98$, $p<0.01$), German Travel ($t=2.88$, $p<0.01$), and English Medical ($t=3.71$, $p<0.005$).

Based on these observations, it appears that the inclusion of training data from similar domains that do not exactly match the specific domain of coverage is a good idea when training classifiers for the domain action classification tasks. The out-of-domain data had very little effect on domain action and concept sequence classification of in-domain examples. Since there were no significant negative effects on these tasks and some small improvements, it seems reasonable

to increase the coverage of the classifiers by including the out-of-domain data. The significant positive effects on speech act classification provide further support for the inclusion of the out-of-domain data in the training set.

4.3.2 Using Translated Data

	English Travel		German Travel		English Medical		German Medical	
	Src	**Trn**	**Src**	**Trn**	**Src**	**Trn**	**Src**	**Trn**
Semantic Dialogue Units	4357	3932	2777	5942	3329	335	1746	548
Domain Actions	683	538	385	855	432	92	208	156
Speech Acts	59	51	42	65	48	21	39	24
Concept Sequences	427	370	265	552	283	64	134	101

Table 38: Corpus contents for original source language data and translated data

We next examined the effects of including manually translated training data on the performance of the classifiers on original source language data. For each language-domain pair, the training data was divided into two sets. The first set contained only semantic dialogue units that had originally been collected in the source language. The second set included semantic dialogue units from utterances that were originally spoken in some other language and then manually translated into the source language. Table 38 contains details about the contents of the original source language data set (*Src*) and the translated data set (*Trn*) after the corpora were separated. For each language-domain pair, the source language data was randomly split into 20 folds for two cross-validation experiments. The classifiers were first trained and tested only on the original source language data. Then, the translated data was added to the training set for each fold, and the classifiers were again tested on the heldout source language test set.

	English Travel	German Travel	English Medical	German Medical
Without Translated Data	47.63%	54.37%	52.81%	55.22%
With Translated Data	48.64%	54.95%	52.93%	55.96%

Table 39: Domain action classification accuracy on original source language data with and without translated training data

	English Travel	German Travel	English Medical	German Medical
Without Translated Data	68.79%	71.88%	78.73%	69.82%
With Translated Data	69.54%	72.64%	78.82%	70.57%

Table 40: Speech act classification accuracy on original source language data with and without translated training data

	English Travel	German Travel	English Medical	German Medical
Without Translated Data	66.61%	71.95%	64.52%	74.11%
With Translated Data	67.64%	72.81%	64.79%	74.74%

Table 41: Concept sequence classification accuracy on original source language data with and without translated training data

Table 39, Table 40, and Table 41 show the mean domain action, speech act, and concept sequence classification accuracy over the 20 folds of original source language test sets with and without translated training data. We observed at least small improvements in performance on all three tasks across both languages and domains. For domain action classification, the only significant increase in accuracy was for English Travel ($t=4.03$, $p<0.001$). For speech act classification, the only significant improvement with translated training data was again for English Travel ($t=3.06$, $p<0.01$). The performance gains on concept sequence classification were significant for English Travel ($t=4.74$, $p<0.0005$) and marginally significant for German Travel ($t=2.73$, $p<0.02$).

The results of this experiment indicate that the inclusion of translated data in the training set for the domain action, speech act, and concept sequence classifiers is justified. Although some of the performance gains were small, including translated data in the training set never reduced classification accuracy and led to significant improvements in some cases. This result makes sense given that the translated data was from dialogues collected in the intended domain of coverage. Although translated utterances may not be as natural as utterances originally spoken

in the source language, they may nevertheless provide additional alternative ways of expressing important concepts in the domain and thus broaden the coverage of the classifiers.

4.3.3 Using Agent-Side Data

	English Travel		German Travel		English Medical		German Medical	
	C	A	C	A	C	A	C	A
Semantic Dialogue Units	5616	2673	5564	3155	2168	1496	1167	1127
Domain Actions	679	491	677	532	249	288	148	198
Speech Acts	58	51	57	56	33	38	31	34
Concept Sequences	457	363	456	388	196	193	110	121

Table 42: Corpus contents for client-side data and agent-side data

We also examined the effects of including semantic dialogue units from the agent side of the dialogue on the classification accuracy on semantic dialogue units from the client side of the dialogue. The data was first divided into two sets according the role of the speaker in the dialogue, either client or agent. Table 42 contains details about the contents of the client-side data set (*C*) and the agent-side data set (*A*) after the corpora were separated. Because the task of our hybrid analyzer in the NESPOLE! translation system is to analyze utterances from the client side of a dialogue, we are interested in how the inclusion of agent-side data in the training set affects the accuracy of the classifiers on client-side semantic dialogue units. Thus, the client-side data for each language-domain pair was randomly split into 20 subsets. In each cross-validation fold, one subset was heldout as a test set for two different classifiers. The first set of classifiers was trained only on the 19 remaining subsets of client-side data for each fold. For the second set of classifiers, the agent-side data was included in the training set for each fold along with the 19 client-side training subsets.

	English Travel	German Travel	English Medical	German Medical
Without Agent Data	51.42%	49.01%	51.76%	49.29%
With Agent Data	51.12%	48.80%	49.49%	50.57%
With Agent Data and Side Feature	51.48%	48.98%	51.94%	50.49%

Table 43: Domain action classification accuracy on client-side test data with and without agent-side training data

	English Travel	German Travel	English Medical	German Medical
Without Agent Data	69.82%	68.28%	83.31%	74.30%
With Agent Data	69.34%	67.33%	80.17%	72.49%
With Agent Data and Side Feature	69.43%	68.15%	83.44%	74.12%

Table 44: Speech act classification accuracy on client-side test data with and without agent-side training data

	English Travel	German Travel	English Medical	German Medical
Without Agent Data	71.05%	68.77%	62.68%	69.00%
With Agent Data	71.23%	68.93%	62.59%	70.37%
With Agent Data and Side Feature	71.26%	68.91%	62.55%	69.69%

Table 45: Concept sequence classification accuracy on client-side test data with and without agent-side training data

The mean accuracies of the domain action, speech act, and concept sequence classifiers trained with and without agent-side and tested only on client-side data are shown in Table 43, Table 44, and Table 45 respectively. The effects of including agent-side data in the training set were somewhat mixed. For domain action classification, there was a drop in accuracy for English and German Travel as well as for English Medical. However, the addition of agent-side data increased domain action classification accuracy for German Medical. Only the English Medical difference was significant ($t=3.75$, $p<0.005$). Speech act classification performance dropped for all language-domain pairs. The decreases in accuracy were not significant for English Travel, marginally significant for German Travel ($t=2.82$, $p<0.02$), and significant for both English Medical ($t=6.62$, $p<0.0001$) and German Medical ($t=3.06$, $p<0.01$). For concept sequence classification, the English Medical classifier suffered a small drop in accuracy. The English and German Travel classifiers and the German Medical classifier showed at least small improvements in performance. The only significant change in concept sequence classification accuracy was for German Medical ($t=3.77$, $p<0.005$).

Due to the mixed results of including non-side-specific data in the training set, we tested one additional condition. We added a single feature for the speaker side to the input vector for each instance in the data. The value of the feature was *c* for client-side instances and *a* for agent-side instances. Using the same division of the data and split into test folds, we ran the classification experiments again. The mean accuracies of the classifiers with the speaker side feature over the 20 client-side test sets when the agent-side data was included in the training sets are shown in the tables in the *With Agent Data and Side Feature* rows. The results of including the side feature when only client-side data is used for training are not shown because they were identical to the results without the side feature. This was of course the expected behavior since the side feature had the same value for all of the training and test instances and thus provided no information. As the results in the tables show, adding the side feature when data from both speaker sides were used for training generally reduced the effects that were seen when the agent-side data was added without the side feature. In particular, the decreases in speech act classification accuracy that were observed when the agent-side data was included without the side feature were essentially eliminated by including the side feature. With the side feature included, the only significant difference in performance between the classifiers trained only on client-side data and the classifiers trained on data from both sides of the dialogue was the increase in accuracy for German Medical domain action classification ($t=3.20$, $p<0.005$). All of the decreases in accuracy observed when the training set included the agent-side data without the side feature were either reduced so the difference was no longer significant or even reversed to small, non-significant improvements. Thus, it appears that including the side feature allows for the training of a single classifier that can support analysis for both sides of a dialogue if necessary without harming the performance on client-side input.

The inclusion of the agent-side data without the side feature in the training set for client-side semantic dialogue units produced much more varied effects than including non-domain-specific or translated data. For the Travel domain, the inclusion of agent-side data led to only minor changes in performance, none of which were significant. However, for the Medical domain, the changes in performance were larger, especially the drop in accuracy for the speech act classifiers. The more significant drop in speech act classification for the Medical domain is probably due to the nature of the domain. The Medical domain involves diagnostic dialogues in which a doctor attempts to identify the health problems a sick traveler is having. Thus, the agent

(doctor) in the Medical dialogues often asks questions about the nature of the client's symptoms, and the client (patient) generally describes their symptoms and responds to the agent's questions. Because the questions and answers may mention similar sets of arguments and the input features to the classifiers included only argument parse information, such a dichotomy could adversely affect the speech act classifiers. On the other hand, the Travel domain involves dialogues between travelers (client) and tourism bureau agents, both of whom may be expected to provide information and ask for information in a more balanced manner.

4.3.4 Training Without the Pseudo-Argument Grammar

The training sets that were used in all of the previous experiments were created by parsing the corpora using the argument and pseudo-argument (and shared) grammars. In order to assess the effects of using the pseudo-argument grammar on classification accuracy, we conducted experiments in which the corpora were parsed using only the argument grammar. Other than parsing with only the argument grammar, all other conditions were exactly the same as those used to test the TiMBL classifiers in the learning approach comparison experiments described in Sections 4.2.2 and 4.2.3. We also used the same 20-fold split that was used in the learning approach comparison experiments. Thus, the results of each fold are directly comparable with the results of those experiments, and we used two-tailed matched-pair t-tests to determine significance.

	English Travel	**German Travel**	**English Medical**	**German Medical**
With Pseudo-Argument Grammar	49.59%	46.51%	51.94%	51.66%
Without Pseudo-Argument Grammar	37.80%	41.89%	43.20%	46.73%

Table 46: Domain action classification accuracy with and without using the pseudo-argument grammar during argument

	English Travel	**German Travel**	**English Medical**	**German Medical**
With Pseudo-Argument Grammar	69.86%	67.62%	77.81%	68.87%
Without Pseudo-Argument Grammar	61.03%	63.88%	67.71%	65.39%

Table 47: Speech act classification accuracy with and without using the pseudo-argument grammar during argument

	English Travel	German Travel	English Medical	German Medical
With Pseudo-Argument Grammar	69.22%	66.90%	64.57%	69.93%
Without Pseudo-Argument Grammar	61.80%	64.85%	63.62%	68.62%

Table 48: Concept sequence classification accuracy with and without using the pseudo-argument grammar during argument

Table 46, Table 47, and Table 48 show the mean accuracies of the domain action classifiers, speech act classifiers, and concept sequence classifiers when the pseudo-argument grammar was used during argument parsing and when only the argument grammar was used. The results clearly demonstrate the importance of the pseudo-argument grammar to the classification tasks. Performance dropped across all classification tasks, languages, and domains when the pseudo-argument grammar was excluded during argument parsing. The decreases in accuracy were quite large for domain action classification and speech act classification, and all of the changes were highly significant. For concept sequence classification, the magnitude of the drops in accuracy was generally smaller than for the other two tasks, with the exception of the English Travel classifier. The performance decreases were still highly significant for the English and German Travel classifiers. The decrease was also significant for the German Medical classifier ($t=3.21$, $p<0.005$) but not for the English Medical classifier.

The decrease in performance observed when the pseudo-argument grammar was excluded is not surprising, especially for classifiers that use only features based on the argument parse. The argument and pseudo-argument grammars were developed in parallel and designed to complement each other. The argument grammar was not written with the intention of being used without the pseudo-argument grammar. Thus, removing the pseudo-argument grammar during argument parsing would likely have reduced the quality of the argument parses. Furthermore, one of the main purposes of the pseudo-argument grammar is to parse phrases with similar meanings that may be useful for determining the Interchange Format representation of a semantic dialogue unit but that would not be covered by an Interchange Format argument. In particular, the phrases are often especially useful for identifying the speech act. This helps to explain the large decreases in speech act classification performance when the pseudo-argument grammar was not used as well as the drops in domain action accuracy since the speech act is part

of the domain action. The results of this experiment also support the observation that the arguments provide critical information for determining the concept sequence, since the drops in performance for the concept sequence classifiers were generally much smaller than those for the speech act and domain action classifiers.

4.4 Domain Action Classification versus Speech Act + Concept Sequence Classification

As discussed in Section 2.5.1, the task of automatically identifying the domain action for a semantic dialogue unit could be broken down in at least three different ways. The first and most obvious approach would be to simply train a single domain action classifier to identify the complete domain action. This task definition will be referred to as the *single classifier* approach. A second approach that is only slightly more complex than a single domain action classifier would be to split the task of domain action classification into two subtasks: speech act classification and concept sequence classification. Under this approach, one classifier is trained to identify the speech act associated with a semantic dialogue unit, and a second classifier is trained to identify the complete concept sequence. This second task definition will be referred to as the *dual classifier* approach. A third possible approach would be to further breakdown the concept sequence classification task into subtasks of identifying the individual concepts associated with a semantic dialogue unit. As in the second approach, a single classifier would be trained to identify the speech act. Then, one classifier would be trained for each individual concept to indicate whether or not that concept should be present in the domain action for the semantic dialogue unit.

One of the goals of the experiments described in this chapter was to aid in the selection of a particular task definition for identifying domain actions in the hybrid analyzer. We argued in Section 2.5.1 that the complexity of extracting a complete domain action from the output of the classifiers in the third task definition makes that approach undesirable. Thus, we only evaluated the accuracy of the classifiers that would be used in the single classifier and dual classifier approaches in the experiments reported earlier in this chapter. The domain action classifiers reported in those experiments correspond to the single classifier task definition, and the speech act and concept sequence classifiers would be used for the dual classifier task definition. Although the accuracy of the speech act and concept sequence classifiers on their individual

subtasks is useful for examining the performance of those tasks, the performance of the subtask classifiers in isolation is not directly comparable with the performance of the single domain action classifiers. Thus, in order to make a more informed decision between the two task definitions, we must examine how well the combination of the outputs from the speech act and concept sequence classifiers identifies the domain action for a semantic dialogue unit.

Because the same random splits of the training corpora were used in the cross-validation experiments for the domain action, speech act, and concept sequence classifiers, we can combine the outputs of the speech act and concept sequence classifiers from the previous experiments to find the domain action that would be predicted by the dual classifier task definition. The accuracy of the dual classifier approach on domain action identification can then be computed and compared directly with the accuracy of the single classifier approach, which is simply the accuracy of the domain action classifiers. Additionally, since the same test set was used for both conditions, we again tested for significance using two-tailed match-pair t-tests.

	English Travel	**German Travel**	**English Medical**	**German Medical**
Single Classifier Approach	49.58%	46.51%	51.94%	51.66%
Dual Classifier Approach	49.46%	46.42%	51.45%	51.05%

Table 49: Domain action classification accuracy of the single and dual classifier approaches based on the basic TiMBL domain action, speech act, and concept sequence classifiers

Table 49 shows the mean domain action classification accuracy of the single and dual classifier approaches based on the output of the TiMBL classifiers used in the first round of experiments described in Sections 4.2.2 and 4.2.3. The row labeled *Single Classifier Approach* contains the mean accuracies over the 20-fold cross-validation of the TiMBL domain action classifiers shown in Table 17. The *Dual Classifier Approach* row shows the mean domain action classification accuracy when the output of the TiMBL speech act classifiers (shown in Table 18) was combined with the output of the TiMBL concept sequence classifiers (shown in Table 19). The TiMBL classifiers reported in the table used only the binary grammar label features based on the trees in the argument parse. As the results in the table show, the single classifier approach

was slightly more accurate at identifying the domain action than the dual classifier approach across all languages and domains. However, the differences were very small and not significant.

	English Travel	German Travel	English Medical	German Medical
Single Classifier Approach	56.86% (1000)	55.53% (1000)	62.80% (1000)	60.25% (1000)
Dual Classifier Approach	57.52% (1000,500)	55.76% (1000,500)	62.99% (1000,1000)	60.60% (1000,1000)

Table 50: Domain action classification accuracy of the single and dual classifier approaches based on the TiMBL+Words domain action, speech act, and concept sequence classifiers

We also compared the performance of the single and dual classifier approaches based on the TiMBL+Words classifiers described in Section 4.2.5.1. The mean domain action classification accuracy of the single and dual classifier approaches is shown in Table 50. For the purpose of this comparison, we used the output from the classifier with the number of word features that maximized the classification accuracy of each classifier. The number of words for each classifier is shown in parentheses in the table along with the mean accuracy. For the dual classifier approach, the number of words for the speech act classifier is listed first, and the number of words for the concept sequence classifier is listed second. When the best TiMBL+Words classifiers were used for all classifiers, the dual classifier approach exhibited a small performance advantage over the single classifier approach across all language-domain pairs. Again, most of the differences were small, and the differences were not significant for German Travel, English Medical, and German Medical. However, for English Travel the accuracy of the dual classifier approach was significantly better than the accuracy of the single classifier approach ($t=3.17$, $p<0.01$).

Although we chose not to incorporate the probability features computed by Rainbow in the online version of the hybrid analyzer, the point of this comparison was simply to examine the classification accuracy achievable using the dual classifier approach and compare it with the single classifier approach. Thus, we also looked at the domain action classification accuracies of the single and dual classifier approaches using the outputs of the TiMBL+Probability classifiers.

Due to memory limitations, there were no single classifier domain action classification results for the Travel domain using the TiMBL+Probability approach described in Section 4.2.5.2.

	English Travel	German Travel	English Medical	German Medical
Single Classifier Approach	??.??%	??.??%	60.07%	58.90%
Dual Classifier Approach	58.13%	55.59%	61.38%	61.33%

Table 51: Domain action classification accuracy of the single and dual classifier approaches based on the TiMBL+Probability domain action, speech act, and concept sequence classifiers

Table 51 shows the mean domain action classification accuracies based on the outputs of the TiMBL+Probability classifiers for which results were available. The concept sequence classifiers with features for individual concept probabilities were used for the dual classifier approach. Using the TiMBL+Probability classifiers for the Medical domain, the dual classifier approach again outperformed the single classifier approach. The increase in accuracy was significant for German ($t=3.43$, $p<0.005$) and marginally significant for English ($t=2.81$, $p<0.02$). Additionally, for the English Travel and German Medical dual classifier approach, the combined TiMBL+Probability classifiers outperformed the combined TiMBL+Words classifiers. The reverse was true for German Travel and English Medical. However, none of the differences between the TiMBL+Probability and TiMBL+Words dual classifier approaches were significant.

	English Travel	German Travel	English Medical	German Medical
Single Classifier Approach	56.86%	55.53%	62.80%	60.25%
Dual Classifier Approach	58.13%	55.76%	62.99%	61.33%

Table 52: Best domain action classification accuracy of the single and dual classifier approaches

Table 52 summarizes the domain action classification results from the best performing single and dual classifier approaches. The TiMBL+Words domain action classifiers provided the

best accuracy for the single classifier approach. For the dual classifier approach, the TiMBL+Probability classifiers provided the best accuracy on domain actions for English Travel and German Medical, and the TiMBL+Words classifiers provided the best accuracy for German Travel and English Medical. As the results in the table show, the best dual classifier approach for each language-domain pair outperformed the best single classifier approach. The only significant difference was for English Travel ($t=3.00$, $p<0.01$).

The results of these experiments provide no evidence that the use of a single domain action classifier provides superior performance compared to the use of separate speech act and concept sequence classifiers in the dual classifier approach. In the worst case, there appears to be essentially no difference in domain action classification performance between the single and dual classifier approaches. In the best case, the dual classifier approach provides at least a small performance boost over the single classifier approach. Furthermore, as discussed in Section 2.5.1, the dual classifier approach provides more flexibility than the single classifier approach with respect to the domain actions that can be produced. Whereas the single classifier approach can only output domain actions that were present in the training data, the dual classifier approach is able to form new domain actions by combining speech acts and concept sequences that were never paired together in the training data. The need to combine the output of the two classifiers introduces only minimal complexity to the hybrid analyzer.

The primary disadvantage associated with the dual classifier approach is the extra processing requirements required for running two classifiers rather than one. Running separate speech act and concept sequence classifiers requires the system to verify that the outputs of the classifiers can be combined to form a legal domain action, but the test can be performed by a simple table look-up. Additionally, running two classifiers may require more memory. For example, based on the memory requirements listed in Table 23, Table 25, and Table 27, the TiMBL+Words domain action classifier for German Travel in these experiments required approximately 120MB. The dual classifier approach that used the TiMBL+Words classifiers required 144MB (84MB + 60MB). If memory were extremely scarce, perhaps the single classifier approach would be preferred, but the difference does not seem to be prohibitive. Furthermore, in the case of the TiMBL+Words approach, the memory requirements of the classifiers could be adjusted by including fewer word features without substantially changing the accuracy of the individual classifiers. If the number of word features used in the German Travel

speech act classifier were reduced from 1000 to 500, the dual approach classifier would require only 109MB, and the domain action classification accuracy would change from 55.76% to 55.84% (a non-significant change). Based on all of these considerations, we chose to use the dual classifier approach in our hybrid analyzer.

4.5 Effects of Input Feature Set Selection

1. Argument grammar root labels
2. Pseudo-argument grammar root labels
3. Words
4. Top-level Interchange Format arguments
5. Speech act
6. Speaker side
7. Speech act probabilities
8. Individual concept probabilities

Figure 35: Types of input features used for domain action classification

As described in Section 2.5.2, there are a number of sources from which sets of input features for the domain action, speech act, and concept sequence classifiers may be drawn. The experiments described thus far in this chapter have used classifiers with various combinations of the types of input features originally listed in Figure 23. The list of input feature types is repeated in Figure 35 for easy reference. In this section, we examine more thoroughly the effects of different combinations of input feature sets on classification performance.

Because we chose to use the dual classifier approach for the hybrid analyzer, we focused our experiments on the effects of feature set selection on speech act classification and concept sequence classification. For each task, we conducted several experiments to assess the effects of various combinations of input feature sets on the performance of the classifier. All of the experiments were conducted on the English Travel & Tourism data using the same corpus, grammars, and 20-fold cross-validation setup that were used in previous experiments. We tested for significance using two-tailed matched-pair t-tests. All of the classifiers tested were memory-

based classifiers implemented using TiMBL. Since the purpose of the experiments was to examine the effects of different input feature sets and not to optimize the performance of a particular combination of input features, the same TiMBL parameter settings were used for all of the classifiers. The classifiers used the IB1 (k-NN) algorithm with a value of 1 for *k*, Gain Ratio feature weighting, and Inverse Linear vote weighting. Thus, the only factor that changed across experimental conditions was the feature sets used in the input vector for the classifiers. It is quite possible that small improvements in accuracy could be obtained for a particular combination of input features by fine-tuning the parameter settings of the classifier, but we did not perform any such tuning.

The binary features for argument and pseudo-argument grammar root label features (feature sets 1 and 2) were created as before by extracting the root labels from the argument parse trees for each semantic dialogue unit produced using the argument and pseudo-argument grammars. In addition to the features based on the root labels of the argument parse, a set of binary features for top-level Interchange Format arguments (feature set 4) was also included. The top-level Interchange Format arguments for a semantic dialogue unit were extracted after the argument parse for a semantic dialogue unit was mapped into the Interchange Format representation. Only the top-level Interchange Format argument feature set was not used in any of the previous experiments. For the word features (feature set 3), binary features were created to indicate the presence or absence of the 250 words with the highest mutual information with the class. Although more than 250 word features could have been used, the experiments with TiMBL+Words classifiers showed that there were generally only small differences for larger numbers of words. Since the word features were only one of several input feature sets, we tested performance using the minimum number of word features that provided essentially maximal accuracy. In addition to the direct word information, the probabilities of speech acts (feature set 7) and individual concepts (feature set 8) were computed using Rainbow naïve Bayes word bigram models as in the TiMBL+Probability classifiers described previously. The speech act feature (feature set 5) was a single feature whose value was the speech act assigned to the semantic dialogue unit, and the side feature (feature set 6) was a single feature that indicated the role of the speaker of a semantic dialogue unit in a dialogue. Table 53 lists the codes used to refer to specific feature sets in the descriptions of the experiments that follow.

	Feature Set	Code
1	Argument grammar root labels	ArgGra
2	Pseudo-argument grammar root labels	PseudoGra
3	Words	Words
4	Top-level Interchange Format arguments	IFArgs
5	Speech act	SA
6	Speaker Side	Side
7	Speech act probabilities	SAProbs
8	Individual concept probabilities	ConcProbs

Table 53: Abbreviation codes used to refer to input feature sets

4.5.1 Speech Act Classification

Although various combinations of some of the feature sets listed in Figure 35 were used in speech act classifiers described earlier, all of the possible feature sets applicable to the speech act classification task were never used in the same classifier. Thus, we first trained such a classifier using all of the feature sets except the speech act (feature 5) and the individual concept probabilities (feature set 8) as a reference against which to compare other combinations of input feature sets. Then in order to test the sensitivity of the classifier to the presence of individual feature sets when all of the possible feature sets were included in the input vector, we systematically removed each of the feature sets from the input vector. A new classifier was trained with each feature set removed, and the performance was compared with the classifier that included all of the feature sets.

Feature Sets	Mean Accuracy
All	80.79%
All-SAProbs	79.20%
All-Words	79.11%
All-ArgGra	80.67%
All-PseudoGra	79.76%
All-IFArgs	80.72%
All-Side	80.57%

Table 54: Mean accuracy of speech act classifiers with all features sets and with single feature sets excluded

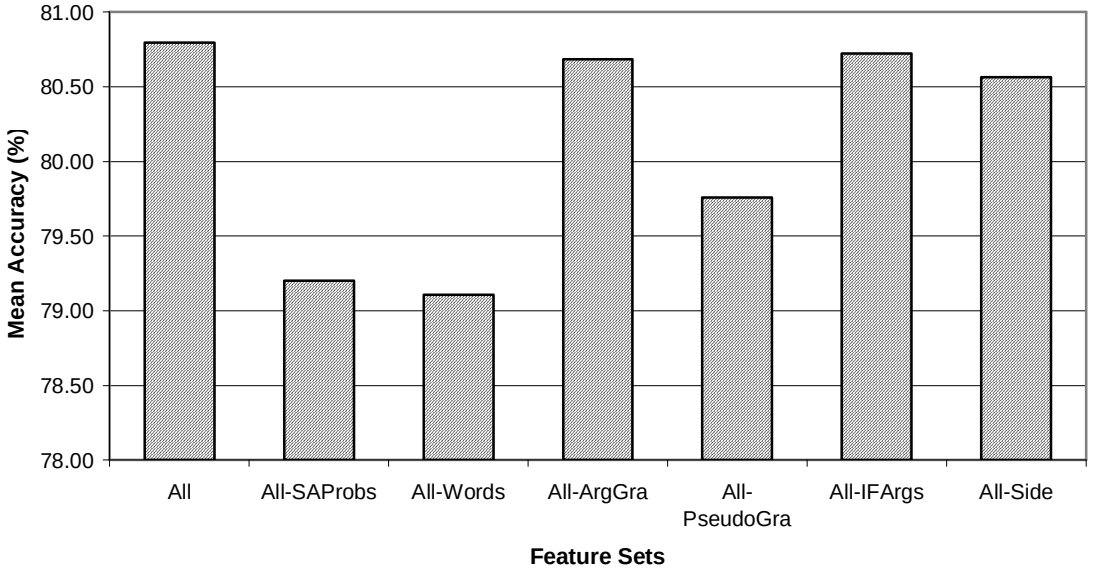

Figure 36: Mean speech act classification accuracy with all feature sets and with single feature sets excluded

The mean accuracy over a 20-fold cross-validation of the speech act classifier with all of the feature sets included in the input vector is shown in Table 54 in the row labeled *All*. The remaining rows in the table list the mean accuracies of the speech act classifiers for which one of the feature sets was excluded from the input vector. For example, the row labeled *All-SAProbs* shows the mean accuracy of the classifier for which the speech act probability features were excluded. The same results are displayed visually in Figure 36. The results show that the removal of any individual feature set from the input vector decreased classification accuracy, although none of the decreases were large in absolute terms. The decreases were significant when the speech act probabilities ($t=6.44$, $p<0.0001$), words ($t=7.82$, $p<0.0001$), or pseudo-argument grammar labels ($t=5.39$, $p<0.0001$) were removed.

Because we chose not to use the probability features in the online version of the hybrid analyzer, the performance of the classifiers without those features is of particular interest. Although the removal of the speech act probabilities resulted in a significant drop in accuracy, the absolute decrease was not too large (1.59%), and the resulting classifier still performed the task quite well. As expected, this result concurs with the results of the experiments described in Section 4.2.5.3. With the exception of the *All-SAProbs* classifier, all of the classifiers with one

feature set excluded listed in Table 54 included the speech act probability features. We observe that the performance of each of those classifiers was similar to the performance of the Rainbow naïve Bayes word bigram classifier from which the probabilities were computed and that the speech act probability features may have overwhelmed the effects of the other feature sets. Thus, in order to test the impact of the individual feature sets in the absence of the speech act probability features, we repeated the single set exclusion experiment starting with the *All-SAProbs* classifier.

Feature Sets	Mean Accuracy
All-SAProbs	79.20%
All-SAProbs-Words	69.76%
All-SAProbs-ArgGra	78.84%
All-SAProbs-PseudoGra	76.51%
All-SAProbs-IFArgs	79.18%
All-SAProbs-Side	78.98%

Table 55: Mean accuracy of speech act classifiers with all features sets except SAProbs and with additional single feature sets excluded

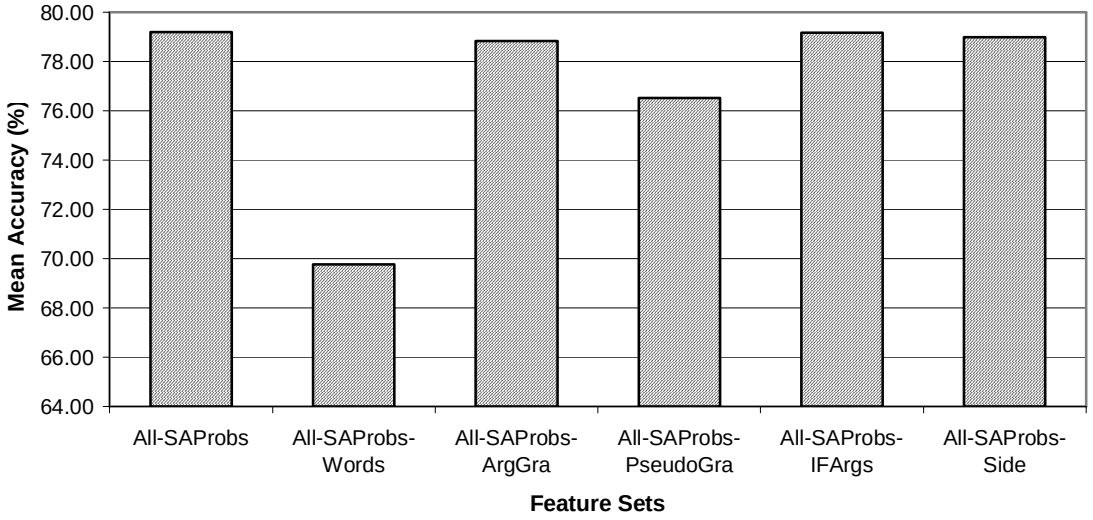

Figure 37: Mean speech act classification accuracy with all features sets except SAProbs and with additional single feature sets excluded

Table 55 lists the performance of the classifiers starting without the SAProbs feature set and removing one additional feature set. The same results are also displayed visually in Figure 37. The effects of excluding each input feature set were similar to those observed when the speech act probability features were included. As before, the removal of each feature set caused at least a small drop in accuracy compared to the classifier that used all of the feature sets, and the only significant decreases were again caused by removing the word features ($t=19.83$, $p<0.0001$) or the pseudo-argument grammar features ($t=10.91$, $p<0.0001$). The main difference without the speech act probability features included was that the effects of excluding the word features or the pseudo-argument grammar label features were more pronounced.

There are two key observations to be made based on these experiments. First, the results provide additional evidence that the features based on information about the words in a semantic dialogue unit and the labels produced by the pseudo-argument grammar are very important for determining the speech act. In both experiments, the removal of the word or pseudo-argument grammar label features produced significant decreases in accuracy. Second, excluding the argument grammar label feature set, the Interchange Format argument feature set, or the side feature did not cause a significant decrease in performance. In the experiments described in Section 4.3.3, the inclusion of the side feature also produced only a small change in performance, so a similar result was not surprising here. The fact that excluding the argument grammar label features or Interchange Format argument features had only small effects on performance also agrees with observations made previously that the arguments are not critical for determining the speech act, at least when other more useful features are available. Additionally, the two feature sets provide at least somewhat redundant information, so excluding one would not necessarily create a large effect.

The fact that some feature sets may be excluded without producing a significant change in performance also allows us to reduce the size of the classifier. When more features are included in the input feature vector, the size of the data files and the example base used by the classifier naturally increases. When a feature set does not provide information that contributes significantly to classification performance, that feature set may be removed from the input vector so that only the most relevant feature sets are included and the size of the training examples can be minimized. Thus, we conducted an additional greedy feature set exclusion experiment to find a minimal set of input feature sets.

Since the classifiers used in the online hybrid analyzer do not use the probability features, we started with the *All-SAProbs* classifier as a baseline. In each round of the experiment, classifiers were trained with one of the input feature sets removed, and the accuracies of the resulting classifiers were compared with the accuracy of the baseline classifier. The criterion for removing a feature set from the input vector was that the performance of the classifier with the feature set excluded was either better than or not significantly worse than the baseline classifier. If one of the feature sets met this requirement, that feature set was removed from the input vector to start the next round, and the remaining feature sets were then excluded one by one. The experiment was greedy in the sense that when several different feature sets met the removal criterion, the feature set removed in the following round was the one for which the associated classifier had the maximum accuracy. For example, the results shown in Table 55 are exactly the tests that would be run for the first round of the experiment. The Interchange Format argument features would be selected for exclusion in the second round because their removal produced the classifier with the highest accuracy that was not significantly worse than the *All-SAProbs* classifier. This process was repeated until the removal of any remaining feature set resulted in a significant decrease in accuracy relative to the baseline.

Round	**Feature Sets**					**Mean Accuracy**	**Significant Decrease?**
	Words	*ArgGra*	*PseudoGra*	*IFArgs*	*Side*		
0	x	x	x	x	x	79.20%	
1		x	x	x	x	69.76%	Y
	x		x	x	x	78.84%	N
	x	x		x	x	76.51%	Y
	x	***x***	***x***		***x***	***79.18%***	***N***
	x	x	x	x		78.98%	N
2		x	x		x	69.97%	Y
	x		***x***		***x***	***79.09%***	***N***
	x	x			x	76.64%	Y
	x	x	x			79.03%	N
3			x		x	54.86%	Y
	x				x	76.66%	Y
	x		***x***			***78.86%***	***N***
4			x			54.20%	Y
	x					76.03%	Y

Table 56: Results of greedy feature set exclusion for speech act classification

Table 56 summarizes the results of the greedy feature set exclusion experiment for the speech act classifier. Each row in the table represents a speech act classifier with different features sets included in the input feature vector. The *Feature Sets* columns indicate which feature sets were included in the input vector for each classifier. An 'x' in the column for a feature set indicates that the feature set was included. The baseline classifier, which was the *All-SAProbs* classifier described in the previous experiment, is shown in the table in the row for Round 0. The *Significant Decrease?* column indicates whether or not there was a significant decrease in accuracy between a particular classifier and the baseline classifier. Only classifiers for which there was not a significant decrease in accuracy had their feature sets considered for exclusion in the next round. The highlighted row in each round indicates the classifier that was selected for processing in the following round. Thus, after Round 1, the *IFArgs* feature set was excluded because the associated classifier produced the highest accuracy that was not significantly worse than the *All-SAProbs* classifier. After Round 2, the *ArgGra* feature set was excluded, and after Round 3, the *Side* feature was excluded. This greedy feature set exclusion experiment provides additional verification that words in the semantic dialogue unit and the pseudo-argument grammar labels are key for identifying the speech act. In fact, a classifier with only those two feature sets provided speech act classification accuracy that was only slightly worse than the classifier that included all of the features.

Finally, we conducted one additional experiment in which we trained speech act classifiers using only a single feature set in the input vector for the classifier. The mean accuracies of the classifiers are shown in Table 57. Figure 38 displays the same information along with the accuracies of the classifier that included all of the feature sets and the classifier that included only the Words, PseudoGra, and Side feature sets. As expected, none of the classifiers trained on only single feature sets performed at the same level as the classifiers that used combinations of feature sets. However, the two feature sets based on word information provided enough information to come within a few percentage points of the best classifiers, providing further confirmation of the importance of the words for identifying the speech act. Interestingly, using the pseudo-argument grammar label features in isolation was worse than using either the argument grammar labels or the Interchange Format arguments. Thus, it appears that the information provided by the pseudo-argument grammar labels is quite useful for supplementing other feature sets but is less useful in isolation.

Feature Set	Mean Accuracy
SAProbs	77.86%
Words	76.03%
ArgGra	59.31%
PseudoGra	54.20%
IFArgs	58.28%
Side	41.39%

Table 57: Mean speech act classification accuracy using only single input feature sets

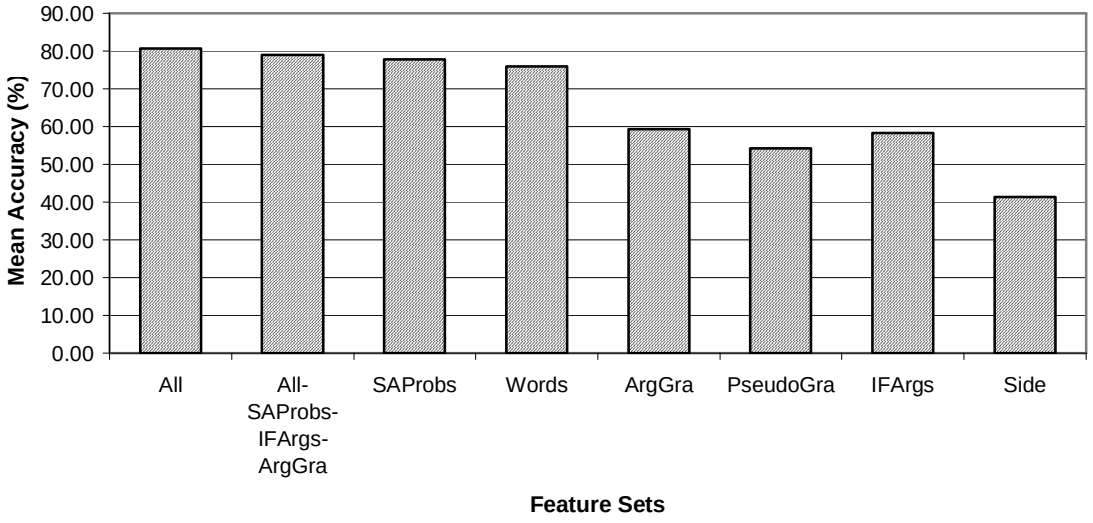

Figure 38: Mean speech act classification accuracy using only single input feature sets

4.5.2 Concept Sequence Classification

We conducted a similar set of experiments to test the effects of various input feature sets on concept sequence classification. All of the feature sets listed in Figure 35 were included in the initial classifier. As in the speech act classification experiments, the probability features corresponding to the classification task, in this case the individual concept probabilities, were included. The speech act was also included as an input feature for the concept sequence classifier. Finally, since the speech act was one of the possible features for the concept sequence

classifier, we also tested the use of the speech act word bigram probabilities as features in the concept sequence classifier.

We first tested the performance of a classifier with all of the features included as well as classifiers with each feature set removed. Additionally, we were especially interested in the effects of excluding the probability features since they were not used in the online version of the hybrid analyzer. Thus, we also tested a classifier with both sets of probability features removed.

Feature Sets	Mean Accuracy
All	68.33%
All-SAProbs	68.46%
All-ConcProbs	67.73%
All-Words	67.32%
All-ArgGra	68.39%
All-PseudoGra	67.08%
All-IFArgs	68.34%
All-Side	68.26%
All-SA	66.26%
All-Probs	68.09%

Table 58: Mean accuracy of concept sequence classifiers with all features sets and with single feature sets excluded

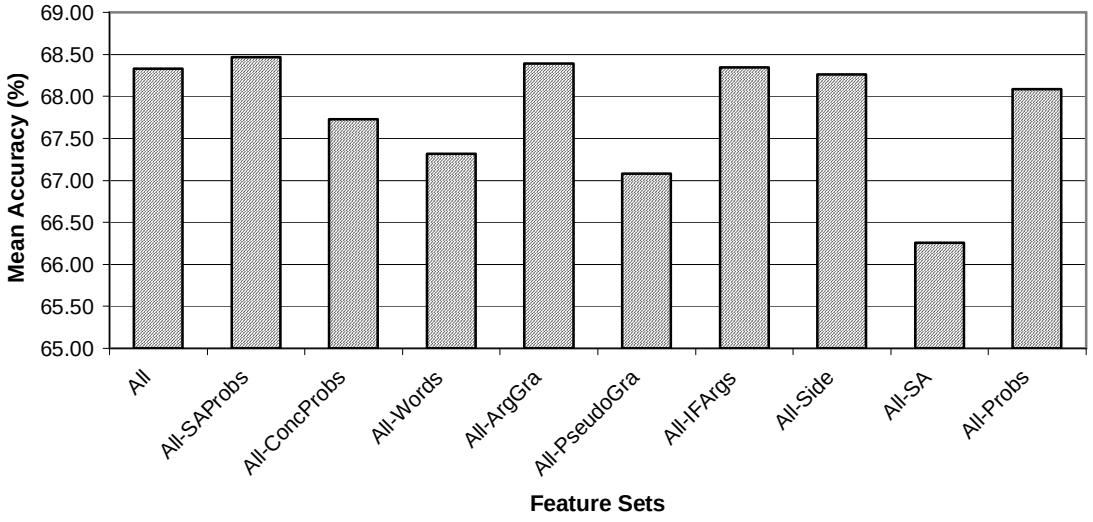

Figure 39: Mean concept sequence classification accuracy with all feature sets and with single feature sets excluded

Table 58 lists the mean concept sequence classification accuracies of the classifier with all feature sets included (*All*) and the classifiers with single feature sets removed. In addition, the accuracy of the classifier with both probability feature sets excluded is listed in the row labeled *All-Probs*. In the descriptions of the experiments in this section, the code *Probs* will be used to refer to the combination of both probability feature sets. Figure 39 shows the same information visually. None of the changes in performance observed in this experiment were large in absolute terms. The largest drop in accuracy occurred when the speech act feature was removed, and the decrease was significant ($t=13.38$, $p<0.0001$). Additionally, exclusion of the concept probability features ($t=2.88$, $p<0.01$), word features ($t=3.12$, $p<0.01$), or pseudo-argument grammar features ($t=4.27$, $p<0.0005$) led to classifiers that were significantly worse than the classifier with all feature sets included. Interestingly, excluding the speech act probability features actually led to a very small non-significant increase in performance. Also, although the removal of only the concept probability features led to a small but significant drop in accuracy, removing both probability feature sets (*All-Probs*) provided slightly superior performance compared to removing only the concept probabilities. The performance of the *All-Probs* classifier was not significantly worse than the performance of the *All* classifier. Thus, unlike in the case of the speech act classifier, it appears that the probability features based on the naïve Bayes word bigram models may be excluded from the concept sequence classifier without significantly harming performance.

We again performed the greedy feature set exclusion experiment beginning with the classifier with no probability features included (*All-Probs*) as the baseline reference classifier. The results of the experiment are summarized in Table 59. The accuracy of the *All-Probs* classifier is listed in the row for Round 0. Since Round 1 of the experiment involved the removal of each individual feature set, those results are not repeated in a separate table. Excluding the speech act feature again resulted in a highly significant ($t=15.47$, $p<0.0001$) decrease in concept sequence classification accuracy, providing evidence that the speech act is a useful indicator of the concept sequence that follows. Removing the pseudo-argument grammar label features also led to a significant decrease in accuracy ($t=3.66$, $p<0.005$). On the other hand, removing the argument grammar label features ($t=2.83$, $p<0.02$) or the Interchange Format argument features ($t=3.22$, $p<0.005$) increased the accuracy of the classifier. Using the same exclusion criterion that

was used in the speech act classification experiment, the *ArgGra* feature set was excluded after Round 1 because its removal led to the maximum accuracy that was not significantly worse than the baseline classifier. After Round 2, the *Words* feature set was excluded because its removal led to a significant increase in accuracy ($t=2.89$, $p<0.01$). In Round 3, the accuracy of the classifier with the *Side* feature excluded was significantly better than the baseline accuracy ($t=3.09$, $p<0.01$) so that feature was excluded for Round 4. In Round 4, as in Round 3, the removal of the *PseudoGra*, *IFArgs*, or *SA* feature set caused a significant decrease in accuracy, so no further sets were excluded.

Round	**Feature Sets**						**Mean Accuracy**	**Significant Decrease?**
	Words	*ArgGra*	*PseudoGra*	*IFArgs*	*Side*	*SA*		
0	x	x	x	x	x	x	68.09%	
1		x	x	x	x	x	67.28%	N
	x		***x***	***x***	***x***	***x***	***68.69%***	***N (Inc)***
	x	x		x	x	x	66.96%	Y
	x	x	x		x	x	68.62%	N (Inc)
	x	x	x	x		x	68.04%	N
	x	x	x	x	x		64.86%	Y
2			***x***	***x***	***x***	***x***	***69.37%***	***N (Inc)***
	x			x	x	x	67.61%	N
	x		x		x	x	67.51%	N
	x		x	x		x	68.71%	N (Inc)
	x		x	x	x		65.06%	Y
3				x	x	x	59.80%	Y
			x		x	x	61.59%	Y
			x	***x***		***x***	***69.36%***	***N (Inc)***
			x	x	x		62.28%	Y
4				x		x	59.31%	Y
			x			x	61.23%	Y
			x	x			62.63%	Y

Table 59: Results of greedy feature set exclusion for concept sequence classification

We can make several observations about the feature sets that are useful for concept sequence classification based on the greedy feature set exclusion experiment. First, the probability features do not appear to be necessary for obtaining the best concept sequence

classification performance. In fact, the best classifiers without the probability features actually outperformed the *All* classifier, which included all possible feature sets. The best concept sequence classifier obtained during the feature set exclusion experiment included the *PseudoGra*, *IFArgs*, *SA*, and *Side* feature sets. The accuracy of the same classifier without the *Side* feature was essentially the same as with the feature. The performance improvement over the *All* classifier was marginally significant with the *Side* feature ($t=2.56$, $p<0.02$) and significant without the *Side* feature ($t=2.89$, $p<0.01$).

Another important observation is that the information about the arguments found in a semantic dialogue unit is important for concept sequence classification, providing further evidence for observations made based on previous experiments. Based on the first round of the feature set exclusion experiment and on the results when the probability features were included, it appears initially that the *ArgGra* feature set and the *IFArgs* feature set are not important since removing either set actually led to an increase in accuracy over the baseline classifier. However, both feature sets provide information about the arguments in semantic dialogue unit, and the information provided may be highly redundant. After one of the sets is removed, the other set apparently becomes important to concept sequence classification. In Round 1 of the feature set exclusion experiment, removing the *IFArgs* feature set while the *ArgGra* feature set was present resulted in an increase in accuracy over the baseline classifier and the second best classifier in the round. In Round 2, when the *ArgGra* feature set was excluded, removing the *IFArgs* feature set led to a decrease in performance and the second worst classifier in the round. Thus, unlike the speech act classifier for which all argument information could be removed without significantly harming the accuracy of the classifier, the concept sequence classifier benefits from having argument information. Finally, the speech act assigned to a semantic dialogue unit clearly provides useful information for identifying the concept sequence. Excluding the *SA* feature consistently led to highly significant decreases in classification performance.

We also conducted an experiment in which classifiers were trained using only one feature set in the input feature vector. Table 60 contains the mean accuracies of the resulting classifiers, and the same information is displayed visually in Figure 40. The classifier with all feature sets included (*All*) and the best classifier from the feature set exclusion experiment (*All-Probs-ArgGra-Words*) are also shown in Figure 40. The performance of the classifiers that used the *ArgGra* feature set and the *IFArgs* feature set was similar, further indicating that the two feature

sets provide roughly equivalent information for concept sequence classification. Interestingly, although the word features were not critical when combined with other feature sets, the classifier that used the *Words* feature set performed the best when the feature sets were used in isolation. Apparently the word features can provide useful information for concept sequence classification, but that information is unnecessary when the right combination of alternative feature sets is available.

Feature Set	Mean Accuracy
SAProbs	47.07%
ConcProbs	58.08%
Words	61.33%
ArgGra	50.49%
PseudoGra	46.48%
IFArgs	50.88%
Side	38.91%
SA	51.73%

Table 60: Mean concept sequence classification accuracy using only single input feature sets

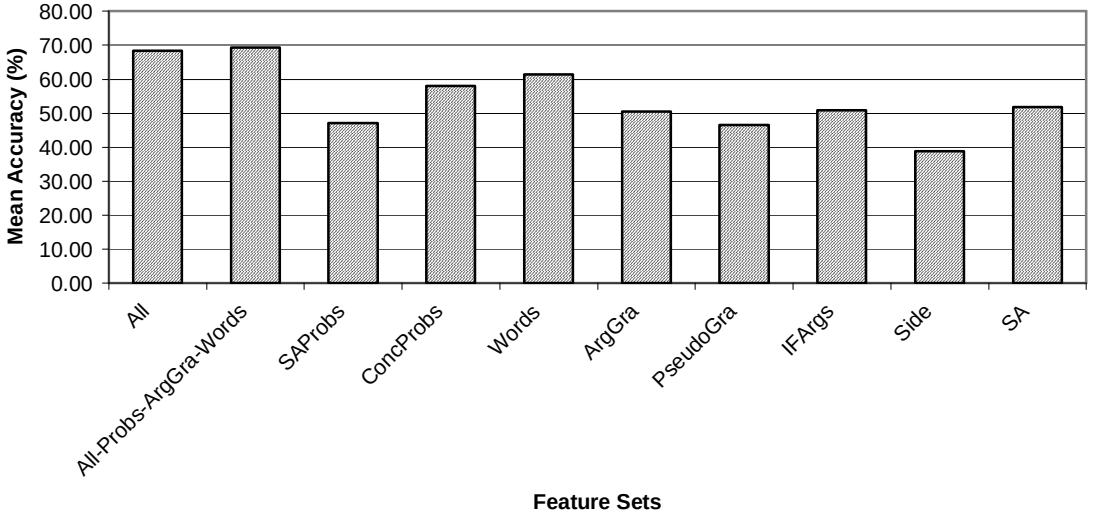

Figure 40: Mean concept sequence classification accuracy using only single input feature sets

4.6 Effects of Interchange Format Specification Fallback

The tasks of parsing arguments, identifying the speech act, and identifying the concept sequence are performed in three separate steps in our hybrid analyzer. As a result, there is no guarantee that combining the best output of the modules that perform each task will produce a valid Interchange Format representation. Although the concept sequence is guaranteed to be legal as long as the manually annotated training data contains no illegal concept sequences, combining the best classified speech act with the best classified concept sequence may result in an illegal domain action. Furthermore, even when the domain action is legal there is no guarantee that it will license all of the top-level arguments present in the argument parse.

4.6.1 Fallback Strategy

As described in Section 2.5.4, the online hybrid analyzer employs a fallback strategy using the Interchange Format specification to ensure that the Interchange Format representation produced for each semantic dialogue unit is valid. Following the completion of speech act and concept sequence classification, the hybrid analyzer first tests the domain action formed by combining the best speech act with the best concept sequence from the classifiers. If the best domain action is legal, the hybrid analyzer next checks if the domain action licenses all of the top-level arguments from the argument parse. If the best classified domain action is legal and licenses all of the arguments, then no further processing is required. However, if the best domain action is invalid or some of the top-level arguments are not licensed by the domain action, then the fallback strategy is used to find the best available alternative. The general idea behind the fallback strategy is to find the highest ranked speech act and concept sequence that form a legal domain action and license the most arguments.

The first step in the fallback strategy is to search for alternative domain actions among the outputs of the speech act and concept sequence classifiers. Recall that the classifiers in the online hybrid analyzer are implemented using the IB1 (k-Nearest Neighbor) algorithm in TiMBL. It may sometimes be the case that the nearest neighbor sets for an input instance contain multiple classes. In such cases, the classifiers return the most frequent class in the nearest neighbor set as the best class, but they can also return the distribution of all of the classes in the nearest neighbor set. If at least one of the classifiers produces a nearest neighbor set with more than one class, the hybrid analyzer sorts each nearest neighbor set in order of decreasing frequency. Then each

concept sequence, in ranked order, is combined with each speech act, in ranked order. So, for example, first the best speech act is tested with each alternative concept sequence. Then the second best speech act is tested with each concept sequence and so on. For each legal domain action found, the hybrid analyzer checks if all of the top-level arguments are licensed. If a legal domain action that licenses all of the arguments is found, fallback processing stops, and the resulting Interchange Format representation is returned.

If the hybrid analyzer fails to find a domain action that licenses all of the top-level arguments using the nearest neighbor sets, then the next step in the fallback process is to test the speech acts and concept sequences found in the training data. As with the nearest neighbor sets, the sets of speech acts and concept sequences from the training data are sorted in order of decreasing frequency. The only exception is that concept sequences that end with *+concept* are moved to the end of the sorted list. The *+concept* concept is a kind of "catch-all" concept that is intended to cover fragmentary utterances and therefore licenses many arguments. Thus, it is moved to the end of the concept sequence list in order to give the fallback process a chance to find more meaningful concept sequences that license the arguments if possible. The best speech act from the speech act classifier is first tested with the alternative concept sequences from the training data. Then the fallback process proceeds through the training data speech act and concept sequence lists in exactly the same way as for the nearest neighbor sets. As soon as a legal domain action is found that licenses all of the top-level arguments, processing ends. If the hybrid analyzer exhausts all possible alternative domain actions without finding one that licenses all of the arguments, then the highest-ranked domain action (i.e. the one found earliest in the fallback process) that licensed the most arguments is returned. In such cases, any unlicensed arguments are dropped from the Interchange Format representation.

4.6.2 Evaluation

In order to examine the effects of using the Interchange Format specification fallback strategy on the output of the hybrid analyzer, we tested the output of the system under three conditions. In the first condition, the best outputs of the speech act and concept sequence classifiers were combined with the best argument parse without any fallback processing. In the second condition, the fallback strategy just described was used. Finally, although the online hybrid analyzer uses only the top-ranked argument parse produced by the SOUP parser, the third

condition was a pilot experiment to test the feasibility and effects of using multiple parses from SOUP in the fallback process. The input to the speech act and concept sequence classifiers in the multiple parse condition was based only on the top ranked argument parse. Fallback processing for multiple argument parses proceeded basically as described above. However, whenever a legal domain action was identified, the hybrid analyzer tested each alternative argument parse in ranked order. If the domain action licensed all of the top-level arguments in any of the argument parses, processing was immediately terminated. If the fallback process ended without finding a domain action that licensed all of the arguments in some argument parse, then the highest ranking domain action that licensed the most arguments in one of the argument parses was returned along with the corresponding argument parse (with unlicensed arguments removed).

We tested each of the three conditions using the NESPOLE! English Travel & Tourism domain data. Each condition was tested using the same type of 20-fold cross-validation setup as was used in previously described experiments. The data was randomly split into 20 sets (containing about 413 examples each), and the same random split of the data was used for all three conditions. Matched pair two-tailed t-tests were used to test for significance. The experiments were run offline on the semantic dialogue unit level in order to isolate the effects of the fallback strategy from segmentation errors. The training data for the first two conditions was prepared as described in Section 4.1 for the domain action classification experiments. However, 28 of the 8289 semantic dialogue units were removed from the data because they caused SOUP to crash when producing multiple parses for the third condition. The removed semantic dialogue units were primarily very long strings of digits (i.e. credit card numbers). SOUP had no problem parsing such inputs when only the best parse was returned as in the online version of the hybrid analyzer. The experiments were conducted using the baseline TiMBL speech act and concept sequence classifiers described in Sections 4.2.2 and 4.2.3. For each test fold, the classifiers were trained on the remaining 19 folds. Then the classifiers were run on the test fold, and the best class and nearest neighbor sets for each test instance in each test fold were recorded. Since the classifier input was based only on the top-ranked argument parse, the same classifier outputs were used for all three test conditions. The fallback list for each test fold was created based on the contents of the 19 training folds.

For each of the first two conditions, the best-ranked parse from SOUP was mapped into its corresponding Interchange Format representation, and the top-level arguments were extracted for

use during fallback processing. The argument parses for the multiple parse condition were extracted from the ranked output of SOUP, but the ranked SOUP output was not used directly for several reasons. First, SOUP allows two types of nonterminal grammar nodes. In addition to the nodes that appear in the output structure of a parse tree, SOUP also supports a second type of nonterminal that is mainly intended for the convenience of the grammar writers. This second type of nonterminal allows grammar writers to group grammar nodes into classes to simplify grammar writing but does appear in the parse tree output. Thus, it is often possible to arrive at the same parse tree output structure through a variety of paths depending on the nonterminals of the second type that were used in the parse. We eliminated such redundant parses from consideration in our pilot experiment. The second reason for not using the ranked SOUP output directly was that parses with the same top-level arguments are identical from the perspective of the fallback mechanism, even if values and/or subarguments differ, since we make the assumption that the argument parses produced using manually developed grammars will be legal. Thus, for the purpose of our pilot experiment, we also eliminated all but the first occurrence of parses with the same (ordered) set of top-level arguments.

In order to produce a reasonably sized set of alternative parses for the multiple parse condition, we set the SOUP parser to return the top 1000 parses. For the vast majority of semantic dialogue units, this resulted in a comprehensive list of all parses. SOUP only returned 1000 parses for 352 (4.3%) of the 8261 semantic dialogue units in the training corpus. Many of the lower ranking parses returned by SOUP covered only a few words in a semantic dialogue unit and contained many fewer arguments than the top parse. Using the fallback strategy described above, such parses with very low coverage would often be selected during fallback because finding a domain action to license few arguments was easier than finding one to license many arguments. In order to avoid such situations, only parses with coverage within 1 word of the best parse were considered in our experiment. The removal of redundant parses and the restriction on coverage led to an average of 4.7 alternative parses per semantic dialogue unit over the whole data set.

We first evaluated the effects of the fallback strategies on the validity of the Interchange Format representations produced by the hybrid analyzer. Illegal Interchange Format representations would generally lead to translation failures since the generation modules expect legal representations. Table 61 shows the average percentage of legal and illegal domain actions

over the 20 cross-validation folds produced by simply combining the best outputs of the speech act and concept sequence classifiers. Without fallback processing, 11.1% of the semantic dialogue units would have been assigned an illegal domain action. Of course, with fallback processing no illegal domain actions were produced.

Legal Domain Actions	Illegal Domain Actions
88.9%	11.1%

Table 61: Mean percentage of legal and illegal domain actions without fallback

	Best Parse	Multiple Parses
Speech Act	8.7%	8.1%
Concept Sequence	27.1%	21.6%
Domain Action	32.0%	26.9%

Table 62: Mean percentage of speech acts, concept sequences, and domain actions changed during fallback processing

Illegal domain actions are only one possible source of invalid Interchange Format representations. Invalid representations may also be produced if a legal domain action does not license all of the top-level arguments found in the parse for a semantic dialogue unit. Table 62 shows the mean percentage of speech acts, concept sequences, and domain actions changed when each fallback strategy was applied. The results in the table reflect the combination of changes made due to illegal domain actions and changes made because of unlicensed arguments. Of course all of the illegal domain actions would have been changed during fallback processing. The remainder of the changed domain actions would have been changed because the best domain action did not license all of the top-level arguments. The percentage of changed domain actions is not simply the sum of the changes made to the speech acts and concept sequences because one or both components could have been changed in order to produce a different domain action. The column labeled *Best Parse* corresponds to the second condition and shows the changes made when only the best parse produced by SOUP was used during fallback. The column labeled

Multiple Parses corresponds to the third condition and shows the changes made when alternative argument parses were considered during fallback.

One important observation is that at least 32% of the Interchange Format representations produced without fallback would have been invalid. This corresponds to the percentage of domain actions changed in the *Best Parse* condition. Since no alternative parses were available, the primary mechanism for repairing illegal representations in the *Best Parse* condition was to change the domain action. The percentage provides a minimum since it was also possible that the best domain action from classification was used (because it licensed the most arguments) but that some arguments were dropped. We also note that speech acts were changed much less often than concept sequences. This was not surprising since many of the changes were triggered by unlicensed arguments, and most arguments are licensed by concepts. Finally, we observe that the *Multiple Parses* condition changed fewer speech acts, concepts sequences, and domain actions than the *Best Parse* condition. This was not surprising since the fallback process could retain the best classified domain action by selecting an alternative parse for which all of the top-level arguments were licensed. Although the difference was small for speech acts, it was significant ($t=4.81$, $p<0.0005$). The differences for concept sequences and domain actions were also highly significant. In the *Multiple Parses* condition, the fallback process selected an alternative parse for an average of 15.8% of semantic dialogue units. Of course, there may have been some overlap between the set of semantic dialogue units for which the domain action was changed and the set for which an alternative parse was selected.

	No Fallback	**Best Parse**	**Multiple Parses**
Speech Act	70.0%	71.6%	72.3%
Concept Sequence	69.4%	57.7%	60.3%
Domain Action	49.6%	44.3%	46.5%

Table 63: Mean speech act, concept sequence, and domain action identification accuracy with and without fallback processing

We also examined the speech act, concept sequence, and domain action identification performance under each of the three conditions. Table 63 shows the mean speech act, concept sequence, and domain action identification accuracy for each condition. The accuracies for the

No Fallback condition were computed by comparing the best output of the classifiers with the annotated domain action for each semantic dialogue unit. The accuracies for the two fallback conditions were computed in the same way using the domain actions produced after fallback processing was completed. Both the *Best Parse* ($t=6.96$, $p<0.0001$) and *Multiple Parses* ($t=9.09$, $p<0.0001$) fallback conditions significantly improved speech act identification accuracy compared to the *No Fallback* condition. However, the *No Fallback* condition exhibited better accuracy on concept sequence and domain action identification, and the differences were highly significant. Additionally, using multiple argument parses during fallback processing significantly improved speech act ($t=6.52$, $p<0.0001$), concept sequence ($t=8.46$, $p<0.0001$), and domain action ($t=7.81$, $p<0.0001$) identification accuracy compared to using only the best parse.

At first glance, the results shown in Table 63 are somewhat puzzling because it appears that using fallback processing to filter out illegal Interchange Format representations actually reduced accuracy. However, this phenomenon can be explained by the fact that the accuracy results reported in the table for the *No Fallback* condition were computed based on the raw classifier output without regard to the validity of the full Interchange Format representation produced. The classifiers were trained using real, possibly erroneous, parses produced by running the argument parser over the training data. Therefore, in some cases the classifiers learned to associate argument parses containing errors with the accompanying speech acts and concept sequences. Then during testing, when the same argument parser was used to parse the test data, the same types of parse errors occurred. Thus, the speech act and concept sequence produced were often correct according to the annotated domain. However, the full Interchange Format representation produced by combining the erroneous argument parse with the classified domain action could be illegal, as shown in Table 62 and discussed previously.

	No Fallback	**Best Parse**	**Multiple Parses**
Speech Act	51.4%	71.6%	72.3%
Concept Sequence	51.0%	57.7%	60.3%
Domain Action	39.9%	44.3%	46.5%

Table 64: Mean speech act, concept sequence, and domain action identification accuracy with and without fallback when invalid Interchange Format representations are counted as incorrect

Table 64 is similar to Table 63 except that the accuracies shown in the *No Fallback* column take into account the validity of the resulting Interchange Format representation in addition to whether or not the classifier output matched the annotated domain action. Speech acts, concept sequences, and domain actions that occur as part of an invalid Interchange Format representation were counted as incorrect for the results reported in Table 64 even if they matched the annotated domain action. Since fallback processing guarantees that the final Interchange Format representation will be valid, the accuracies in the *Best Parse* and *Multiple Parses* columns do not change. However, the accuracies for the *No Fallback* condition drop substantially when the validity of the Interchange Format representation is taken into account. The differences between the *No Fallback* condition and the both fallback conditions were highly significant. These results show that fallback processing does in fact improve the overall accuracy of speech act, concept sequence, and domain action identification when the validity of the full Interchange Format representation in taken into account.

	Parsed Arguments (Count)	**Dropped Arguments (Count)**	**Dropped Arguments (Percent)**
No Fallback	552.2	184.4	33.4%
Best Parse	552.2	16.3	3.0%
Multiple Parses	530.1	3.0	0.6%

Table 65: Mean arguments dropped with and without fallback

Finally, we also examined the effects of fallback processing on the top-level arguments produced by the analyzer. The *Parsed Arguments* column in Table 65 shows the mean total number of top-level arguments over the entire test set found in argument parses. For the *No Fallback* and *Best Parse* conditions, the count is the sum over all the semantic dialogue units in a test set of all the top-level arguments found in the best parse. For the *Multiple Parses* condition, the count is the sum of the top-level arguments present in the parse selected during fallback processing. The *Dropped Arguments* columns show the mean count and percentage of arguments dropped over a whole test set. For the *Best Parse* and *Multiple Parses* conditions, the table shows the total number of arguments that were dropped as a result of fallback to produce the final

Interchange Format representation for each semantic dialogue unit. Since the *No Fallback* condition did not actually do any fallback processing, no arguments would actually have been dropped from the Interchange Format representations. However, for the sake of comparison, we calculated the total number of arguments that would have to be dropped from illegal Interchange Format representations. The count of "dropped" arguments for the *No Fallback* condition included two components. First, all arguments that appeared in a semantic dialogue unit for which an illegal domain action was produced were counted as dropped (63.9, 11.6%). Second, when a legal domain action was produced for a semantic dialogue unit, any arguments that were not licensed by the semantic dialogue unit were counted (120.5, 21.8%). As the table shows, 1 out of every 3 arguments would have been dropped as a result of being part of an illegal Interchange Format representation without fallback processing. Using only the best parse, the percentage of arguments dropped was reduced by a factor of 10. The use of multiple parses during fallback processing further reduced the percentage of arguments dropped to below 1%. Under both the *Best Parse* and *Multiple Parses* fallback strategies, arguments were only dropped after an exhaustive search through all possible combinations of known speech acts and concept sequences failed to find a domain action that licensed all of the arguments. One additional interesting result of our experiments was that the number of arguments dropped in each test set exactly matched the number of exhaustive searches conducted. This indicates that only one argument was dropped for each semantic dialogue unit for which an exhaustive search was required.

	Before Fallback	**Best Parse**	**Multiple Parses**
Correct Arguments	284.1	283.1	281.9
Extra Parsed Arguments	268.2	252.9	245.3
Missed Tagged Arguments	306.6	307.6	308.8
Precision	0.514	0.528	0.535
Recall	0.481	0.479	0.477
F_1-Measure	0.497	0.502	0.504

Table 66: Mean top-level argument identification performance

We also evaluated how well the arguments present in the annotated data were identified during argument parsing and fallback processing. Table 66 shows mean top-level argument identification results (over complete test sets) for the best parse before fallback processing was applied and for the final Interchange Format representations after fallback processing with only the best parse and with multiple parses. The *Correct Arguments* row shows the mean number of parsed arguments that were also present in the tagged data. The *Extra Parsed Arguments* row shows the number of arguments that were only present in the argument parses, and the *Missed Tagged Arguments* row shows the number of arguments that were present in the tagged data but not in the argument parses. The final three rows show the mean precision, recall, and F_1-measure calculated based on the counts of correct, extra, and missed arguments over a whole test set. The results in the table show that the main effect of fallback processing was to remove extra arguments that appeared in the argument parses but not in the tagged data. The number of correct and missed arguments changed very little when fallback processing was applied.

Based on the results of our experiments with fallback processing, we are able to make several observations regarding the performance of the hybrid analysis approach. First, the results clearly demonstrate the necessity of having a mechanism for ensuring that valid Interchange Format representations are produced. In a purely grammar-based analysis approach that uses hand-written grammars to parse complete domain actions, the system is guaranteed to output only legal Interchange Format representations, assuming that the grammars do not contain rules that violate the specification. However, when the Interchange Format components are identified by essentially independent automatic processes as in our hybrid analysis approach, there is no guarantee of valid output representations. Without some form of fallback processing, approximately one third of the Interchange Format representations produced by the hybrid analyzer would have been illegal. However, with fallback processing, all of the output representations were guaranteed to be valid, and the results indicate that our fallback strategy did a very effective job of finding domain actions to license most of the top-level arguments present in each argument parse.

The results of our experiments also provide some insights into the argument parsing and domain action classification performance of our hybrid analyzer. At first glance, it appeared strange that the domain action classification accuracy would be higher before fallback than after. However, a closer analysis of the results revealed a likely explanation. First, as noted before,

speech act classification accuracy was actually improved as result of fallback processing. Thus, the decline in domain action classification performance was primarily a result of errors in concept sequence identification, which is not surprising since concepts are more closely tied to the arguments than speech acts. Furthermore, the results from argument parsing indicate that the main effect of fallback was to remove extraneous arguments. Of course, the fallback strategy that used only the best parse would not have been able to find arguments from the tagged data that were missed since only one parse was used. With multiple parses available during fallback processing, one might expect that the number of missed arguments would be reduced if the main reason for missed arguments were ambiguities in argument parsing in which the same phrase could be parsed under multiple top-level arguments depending on context. However, even when multiple parses were used during fallback processing, the number of arguments from the tagged data that were missed in the final parse selected remained essentially the same. We interpret this to be an indication that the argument grammars simply did not cover some of the arguments from the tagged data. Since the concept sequence is closely linked with the top-level arguments, it would not be surprising that the correct concept sequence could not be identified. Rather, the fallback strategy would identify the most likely concept sequences that licensed the arguments that were present. The fact that the concept sequences and domain actions were identified more accurately before fallback was probably due to the fact that the concept sequence classifiers were trained using the output of the argument parser rather than the canonical arguments from the tagged data. Thus, although the argument parses may have contained erroneous arguments that were not legal with the tagged domain actions, the classifiers would have been trained with such parse errors and learned to associate them with the concept sequence. Additionally, as was discussed previously, the domain action classification performance was not better without fallback processing when the validity of the Interchange Format representation was taken into account.

Finally, our pilot experiment using with multiple argument parses during fallback processing showed that using multiple parses could in fact improve system performance. The multiple parse strategy significantly improved classification accuracy compared to using only the best parse. The multiple parse strategy also reduced the percentage of top-level arguments that had to be dropped as a result of fallback processing and the number of extraneous arguments present in the final output compared to the best parse strategy. These results indicate the promise

of using multiple parses in the hybrid analysis approach. However, the approach used in our pilot experiment would be impractical for use in a real-time online translation system. In order to get a relatively small set of alternative top-level argument sets, we had to set SOUP to produce a large number of parses, and then each of those parses had to be mapped into its Interchange Format representation so that we could find unique sets of top-level arguments. If the identification of parses with unique top-level arguments were not performed, then the number of alternative parses that would have to be tested with each domain action during fallback processing would have been much larger.

Chapter 5 Evaluation of Portability

One of the primary research goals of the NESPOLE! project was to investigate methods for improving the domain portability of interlingua-based speech-to-speech machine translation systems. With this goal in mind, one of the motivations for developing the hybrid analysis approach described in this dissertation was to reduce the effort required for human experts to port the analysis module for the interlingua-based translation system to a new domain of coverage. The first domain addressed in the NESPOLE! translation system was the Travel & Tourism domain. The translation system for this domain was intended to facilitate dialogues between tourists traveling to a foreign country and agents at a local tourism bureau in the country. Tourists could use the NESPOLE! system to get information about the area to which they were traveling directly from the local agent. In order to assess the portability of the NESPOLE! translation system to a completely new domain, translation engines were also developed for the Medical Assistance domain near the end of the project. In this new domain, the translation system was intended to support dialogues between travelers in a foreign country who were not feeling well and a local doctor who did not speak the traveler's language. In this case, the traveler might access the NESPOLE! system from their hotel in order to get preliminary medical advice from the local doctor. The domain portability of the hybrid analysis approach was evaluated within the context of porting the NESPOLE! translation system from the Travel & Tourism domain to the Medical Assistance domain.

The experiments described in this chapter were designed to assess the portability of the hybrid analysis approach to a new domain. The first section describes a data ablation experiment designed to examine the effects of different quantities of training data on domain action classification performance. Annotation of training data requires human effort, so the amount of data required for training domain action classifiers is an important aspect of the portability of our hybrid analysis approach. Our data ablation experiments demonstrate that acceptable levels of classification performance can be achieved with relatively small amounts of training data that can be annotated reasonably quickly. The following section describes an end-to-end evaluation of the NESPOLE! translation system for the Travel & Tourism domain. This evaluation was designed to assess the performance of the full translation system using our hybrid approach for

analysis and serves as a reference point for performance on a well-developed domain. The final section describes an experiment designed to assess the portability of our hybrid analyzer to the Medical Assistance domain. We compare the times required for data annotation and grammar development for the hybrid analysis approach with those for a purely grammar-based approach and find that the hybrid approach results in reduced development and maintenance effort. We also find that the purely grammar-based approach performs only slightly better than our hybrid approach and that the hybrid approach suffers less performance degradation on unseen input and on input from an automatic speech recognizer. The results presented in this chapter indicate improved portability by showing that our hybrid approach requires less human development effort than a purely grammar-based approach while providing nearly equivalent performance and improved robustness.

5.1 Data Ablation Experiments

One important issue regarding the use of machine learning approaches for domain action classification is how much tagged data is required for training the classifiers. The amount of training data required lends insight into the portability of the of the hybrid analysis approach because it directly affects the time and effort required for annotation. In order to assess the training data requirements for domain action classification in our hybrid analysis approach, we conducted a set of data ablation experiments in which we varied the number of examples used for training. As described in Sections 2.6 and 4.4, we chose to perform domain action classification using a dual classifier setup in the online version of our hybrid analyzer that is used in the NESPOLE! translation system. In the dual classifier setup, one classifier was trained to identify the speech act, and a second classifier was trained to identify the complete concept sequence. Thus, in our data ablation experiments we tested the performance of the speech act and concept sequence classifiers with increasing amounts of training data. We also looked at the effects of the amount of training data on domain action classification accuracy when the best outputs of the speech act and concept sequence classifiers were combined.

The data ablation experiments were run using the same corpora and grammars that were described in Section 4.1 and used for the domain action classification experiments in Chapter 4. In order to create the training data for the classifiers, each semantic dialogue unit in the training corpora was first parsed for arguments using the argument and pseudo-argument grammars. The

speech act and concept sequence classifiers used in the data ablation experiments were the same memory-based TiMBL classifiers used in the first set of domain action classification experiments described in Sections 4.2.2 and 4.2.3. The input feature vectors for both classifiers included binary features for the argument grammar root labels and pseudo-argument grammar root labels in the argument parse. As in the previous experiments, the concept sequence classifiers also used the speech act assigned to a semantic dialogue unit as an input feature. We ran data ablation experiments for English and German for both the NESPOLE! Travel & Tourism domain and the NESPOLE! Medical Assistance domain.

In the data ablation experiments, we tested the accuracy of the speech act and concept sequence classifiers using training set sizes from 100 examples, where each example represents a semantic dialogue unit, up to all of the available data. The training set size was incremented in steps of 100 examples from 100 to 1000. Increments of 500 examples were then used from 1000 to the maximum available training set size. The test set size for each language-domain pair was selected to allow for the largest possible multiple of 500 for the maximum training set size. For example, the English Travel corpus contained 8289 semantic dialogue units (as shown in Table 13). Thus, the size of the test sets for the English Travel data ablation experiments was 289 examples, and the maximum training set size was 8000 examples. For each language-domain pair, the same test set size was used for all training set sizes. Table 67 lists the size of the training corpus as well as the test set size and maximum training set size for each language-domain pair.

	English Travel	**German Travel**	**English Medical**	**German Medical**
Semantic Dialogue Units	8289	8719	3664	2294
Test Set Size	289	219	164	294
Maximum Training Set Size	8000	8500	3500	2000

Table 67: Training and test set sizes for data ablation experiments

Examples containing the appropriate input features and class for the speech act and concept sequence classifiers were first created for each semantic dialogue unit in the corpus for each language-domain pair. Then the data ablation experiments were conducted by randomly

sampling training and test sets from the collections of examples. For each sample, a test set was first randomly extracted from the database, and then a training set was randomly selected from the remaining examples. For the maximum training set size, the training set included all of the examples not used in the test set. Following the extraction of the training and test sets, a TiMBL classifier was trained on the selected training set, and the accuracy of the resulting classifier on the selected test set was computed. In order to allow for examination of the domain action classification accuracy of the combined speech act and concept sequence classifier outputs, the training and test sets for the speech act and concept sequence classifiers were extracted in parallel so that the examples were derived from the same semantic dialogue units. Thus, the outputs of the classifiers for a particular example could be combined to get the classified domain action. This sample-train-test process was repeated 50 times for each training set size.

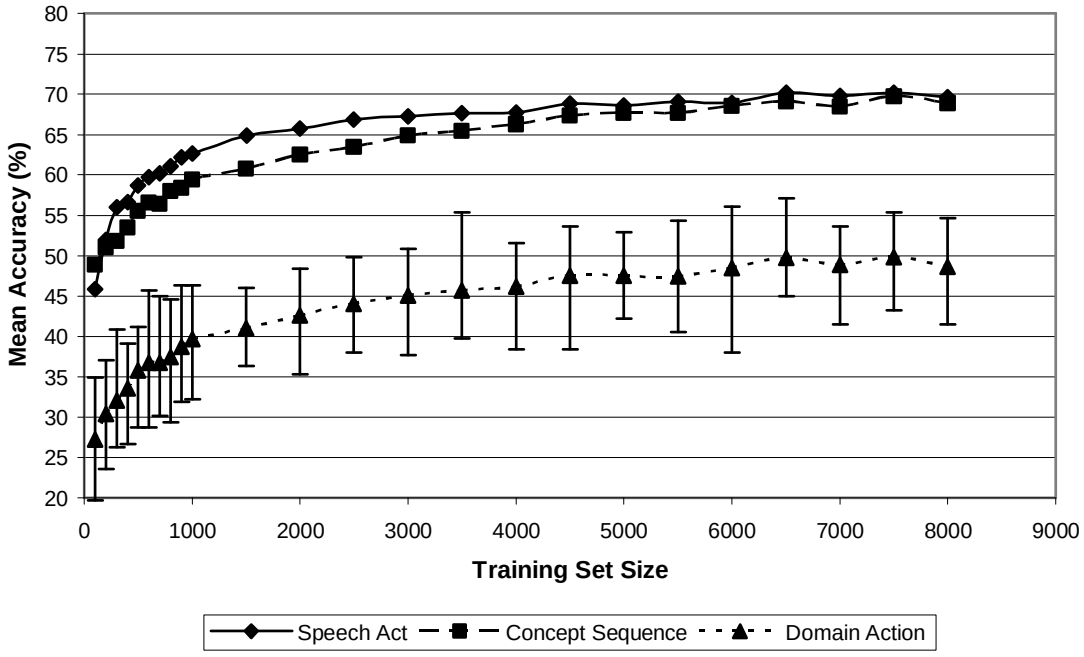

Figure 41: Mean speech act, concept sequence, and domain action classification accuracy with increasing amounts of training data for English Travel

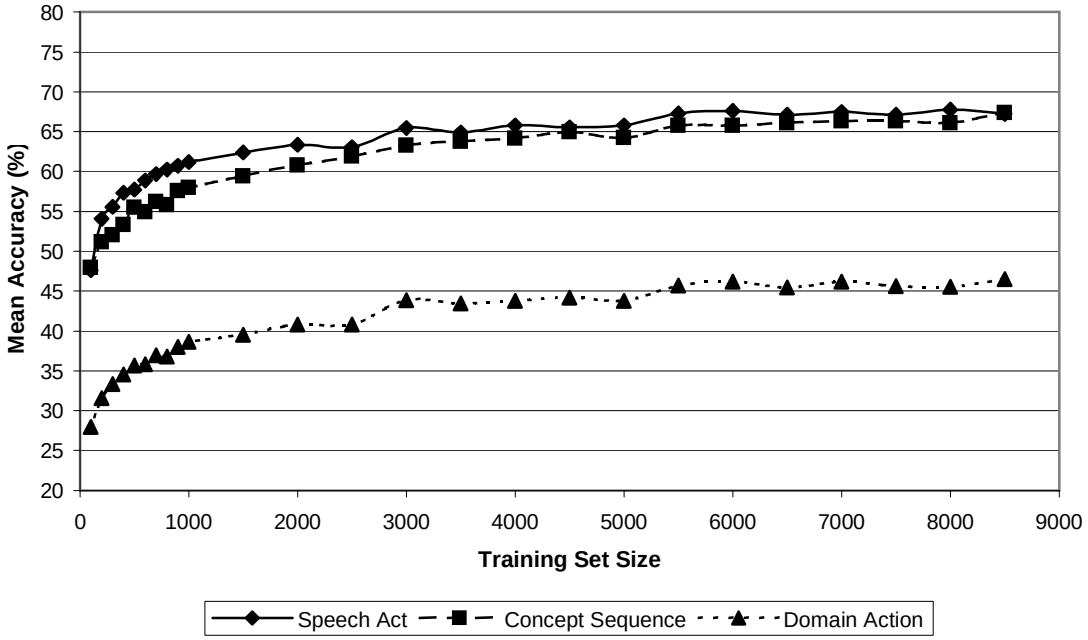

Figure 42: Mean speech act, concept sequence, and domain action classification accuracy with increasing amounts of training data for German Travel

Consider for example the case of using 4000 examples for the training set in the English Travel domain. First, a set of 289 examples was extracted for the speech act classifier, and the corresponding examples for the same semantic dialogue units were used as the test set for the concept sequence classifier. Next, 4000 examples were randomly selected from the 8000 remaining examples as the training set for the speech act classifier, and the corresponding set of examples were used for training the concept sequence classifier. Each of the classifiers was then trained and tested using the selected sets, and the outputs of the classifiers for each example in the test set were combined to produce the predicted domain action. The same process was then repeated using new training and test set samples.

Figure 41 and Figure 42 illustrate the results of the data ablation experiments for the English and German Travel & Tourism data. In each graph, the mean accuracy of the speech act (solid line with diamonds) and concept sequence (long-dashed line with squares) classifiers over 50 samples is plotted against the number of examples used in the training sets. The mean domain action classification accuracy using the combined outputs of the speech act and concept sequence classifiers is also shown (short-dashed line with triangles). The error bars in Figure 41 show the

minimum and maximum domain action classification accuracies observed for each training set size using the English Travel data. Similar trends are observed for both languages. As the graphs illustrate, the mean accuracy for each classification task increases quite rapidly initially. Around 3000-4000 training examples, the accuracy of the classifiers appears to begin leveling off. All of the accuracy curves appear to be basically flat after about 6000 examples. For both languages, an analysis of variance shows that differences among the mean accuracies for classifiers trained with at least 6000 examples for each classification task are not statistically significant. As additional training set sizes of less than 6000 examples are added to the analysis of variance, we begin to observe statistical significance with specific values depending on the language and classification task. Additionally, pairwise t-tests among training sets of different sizes show statistical significance between some pairs of training set sizes. Based on the pairwise t-tests, we observed a general tendency for the performance of classifiers trained with at least 6000 examples to be significantly better than that of classifiers trained with fewer examples.

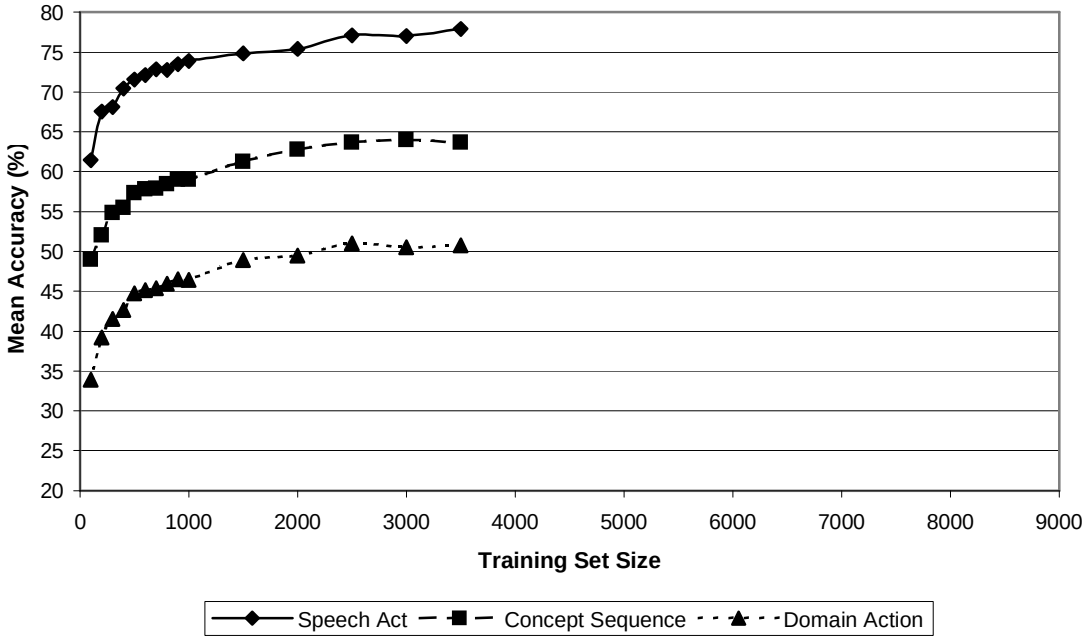

Figure 43: Mean speech act, concept sequence, and domain action classification accuracy with increasing amounts of training data for English Medical

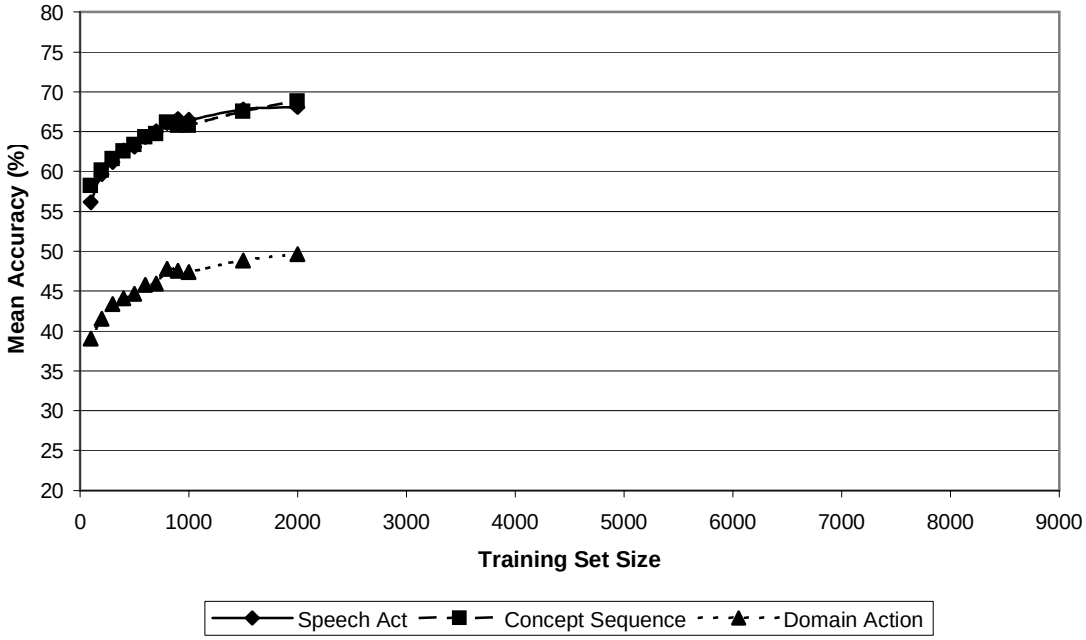

Figure 44: Mean speech act, concept sequence, and domain action classification accuracy with increasing amounts of training data for German Medical

Figure 43 and Figure 44 shows the results of the data ablation experiment for the English and German Medical Assistance data. Although there was much less data for the Medical domain, the graphs use the same scale as the Travel domain graphs for easier comparison. As the graphs illustrate, the trends for the Medical domain data appear to be similar to those for the Travel data. For both languages the performance of the classifiers appears to be in the phase of rapid growth observed for the Travel classifiers, although the improvements in accuracy for the English classifiers may be on the verge of slowing down. This is not surprising since the amount of data available for the Medical domain fell into the range in which classification accuracy was still growing rapidly for the Travel domain. One notable difference between the graph for the English Medical data and the graphs for the remaining language-domain pairs is the much larger gap between the line for speech act accuracy and the line for concept sequence accuracy. The larger difference is most likely attributable to the fact that the most common speech act occurred more frequently in the English Medical data as shown in Table 14 and Table 15 (Section 4.1). The most frequent speech act in the data for each language-domain pair was *give-information*. In the English Medical data, the *give-information* speech act covered about 60% of the semantic

dialogue units, whereas it only covered 35%-41% of the semantic dialogue units in the data for the other language-domain pairs. The higher speech act classification accuracy on the English Medical data than the on the other three data sets reflects this difference and accounts for the larger gap observed in Figure 43.

While the results of the data ablation experiments for the Medical domain may not be extremely informative due to the limited amount of data available, the results from the Travel domain experiments provide evidence of the portability of our hybrid analysis approach. It appears from these experiments that performance becomes relatively stable by around 6000 training examples. Although it is of course possible that the addition of more training examples may result in further improvements in accuracy, the trend beyond 6000 examples seems to be relatively small improvements in accuracy for relatively large increases in training set size. Recall that training the classifier used in our hybrid analysis approach requires training data that has been segmented into semantic dialogue units and tagged with domain actions. Based on the times for annotation reported in Table 76 for the Medical domain portability experiment described later in Section 5.3, tagging such training data would require about 0.7 minutes per semantic dialogue unit. Thus, assuming that a stable Interchange Format specification had been defined for a domain, a training set containing 6000 examples could be annotated for training classifiers used in the hybrid analyzer in about 70 person-hours. It is also worth noting that about 90-95% of the peak performance achieved by the Travel domain classifiers could be achieved with 3000 training examples (half the minimum training data for peak performance). A training set of that size could be annotated for training the classifiers in about 35 person-hours.

As we will see in the following sections, the level of classification performance achieved with the full training data sets is adequate to provide reasonable performance for end-to-end translation. The performance of the analysis classifiers used in the end-to-end translation experiments described in the following sections is similar to that observed in the data ablation experiments when the full training sets were used. The end-to-end results will demonstrate that the best performance levels achieved in the data ablation experiments are adequate for achieving reasonable end-to-end translation performance. In fact, the percentage of acceptable end-to-end translations will generally be higher than the percentage of canonical domain actions identified by the domain action classifiers.

5.2 End-to-End Evaluation of the NESPOLE! Travel & Tourism Domain

In this section, we describe the final end-to-end evaluation of the NESPOLE! translation system for the Travel & Tourism domain. The evaluation is described further in [NESPOLE!-D18, 2003]. The purpose of the end-to-end evaluation was to assess the overall translation performance of the full NESPOLE! system from source language input to target language output. Thus, the end-to-end performance reflects the combined performance of all components of the system. We focus on the aspects of the evaluation most relevant to our hybrid analysis module.

The NESPOLE! system was designed for use by an Italian-speaking agent and a English-, German-, or French-speaking client. Since our hybrid analyzer was used in the English and German translation servers, we focus on English and German input. For each source language, the test data for the end-to-end evaluation was extracted from 2 completely unseen dialogues. Prior to the evaluation, the test set dialogues had never been seen by the system developers and were never used to develop or train any component of the NESPOLE! system. Because the English and German translation servers were designed to support the client, only the client-side turns were extracted from the dialogues and used in the evaluation. The English test set contained 110 utterances (i.e. speaker turns) composed of 232 semantic dialogue units, and the German test set contained 246 utterances composed of 356 semantic dialogue units. The end-to-end evaluation included automatically recognized and manually transcribed versions of the utterances in the test sets.

The online setup of the hybrid analyzer (Section 2.6) was used for the evaluation. All of the classifiers were memory-based learners implemented using TiMBL. The segmentation classifier used the input features described in Section 2.4.2.2 and 5 neighbors (k=5). The dual classifier approach was used for domain action classification, and the input feature vectors for the speech act and concept sequence classifiers used only the binary features for the argument grammar and pseudo-argument grammar root labels in the argument parse. The speech act was also included as a feature for the concept sequence classifier. The data used to train the classifiers was the data for the English and German Travel domains described in Section 4.1. The segmentation classifiers were trained on the same corpus, which included 35417 segmentation examples for English and 45945 examples for German (including partial examples as described in Section 3.3).

5.2.1 Analyzer Performance

		Test Set	
Task		**Transcription**	**Speech Recognition**
Domain Actions	**P**	0.559	0.476
	R	0.530	0.427
	F_1	0.544	0.450
Speech Acts	**P**	0.714	0.673
	R	0.677	0.603
	F_1	0.695	0.636
Concept Sequences	**P**	0.677	0.596
	R	0.642	0.535
	F_1	0.659	0.564
Individual Concepts	**P**	0.576	0.385
	R	0.500	0.299
	F_1	0.535	0.337
Top-Level Arguments	**P**	0.593	0.408
	R	0.409	0.293
	F_1	0.484	0.341

Table 68: Performance of the English hybrid analyzer on full speaker turns from the Travel domain

		Test Set	
Task		**Transcription**	**Speech Recognition**
Domain Actions	**P**	0.706	0.399
	R	0.640	0.351
	F_1	0.672	0.374
Speech Acts	**P**	0.805	0.518
	R	0.730	0.455
	F_1	0.766	0.484
Concept Sequences	**P**	0.786	0.697
	R	0.714	0.612
	F_1	0.748	0.652
Individual Concepts	**P**	0.553	0.403
	R	0.472	0.348
	F_1	0.509	0.374
Top-Level Arguments	**P**	0.580	0.409
	R	0.488	0.343
	F_1	0.530	0.373

Table 69: Performance of the German hybrid analyzer on full speaker turns from the Travel domain

Table 68 and Table 69 show the performance of the English and German hybrid analyzers on five tasks when full speaker turns were used as input in the end-to-end evaluation. Thus, the results in the tables reflect the performance of the full online version of the analyzer including the use of the cross-domain grammar and the fallback strategy to guarantee that the representations produced by the analyzers adhered to the Interchange Format specification. The *Transcription* column contains the results when the input to the analyzer consisted of manually transcribed utterances, and the *Speech Recognition* column contains the results for automatically recognized utterances. After the NESPOLE! Travel system had been frozen for the end-to-end evaluation, the test sets were annotated with Interchange Format representations. For each analysis task shown in the tables, the precision and recall were computed at the turn level rather than the semantic dialogue unit level since the analyzers were not guaranteed to return the same semantic dialogue units as the annotated data.

One interesting observation regarding the results in the tables is the substantial difference in performance between the English and German analyzers. In particular, the German analyzer exhibited substantially higher performance than the English analyzer for domain actions, speech acts, and concept sequences on transcribed input but substantially lower performance for input from the speech recognizer. On the other hand, the analyzers for both languages exhibited similar performance for individual concepts and top-level arguments on both transcribed and recognized input. Although this phenomenon may appear puzzling at first, it can be explained by the relatively high occurrence of the back channel "mhm" in the German data. The correct Interchange Format representation for "mhm" is the *acknowledge* domain action with no concepts or arguments. The cross-domain grammars for both English and German contain a rule that produces the correct representation whenever "mhm" occurs in the input. Thus, for transcribed input, the German analyzer received a higher proportion of correct domain actions because "mhm" occurred more frequently in the German test set than in the English test set. The larger proportion of correct domain actions resulted in a correspondingly larger proportion of correct speech acts and full concept sequences (in this case the empty concept sequence). Since *acknowledge* contains no individual concepts and no arguments, the more frequent occurrence of "mhm" in German had no effect on these measures. Although the German analyzer benefited from the higher proportion of "mhm" back channels in transcribed input, it suffered on input from the speech recognizer. This occurred because the German recognizer often misrecognized

"mhm" as noise and thus produced no text for the analyzer to parse. Since there was no recognized input, no domain action was produced. In this case, the performance on domain actions, speech acts, and concept sequences was negatively impacted, but the individual concepts and arguments were again unaffected.

Additionally, as would be expected with the introduction of recognition errors, the performance of the analyzers on all tasks was lower for automatically recognized input than for manually transcribed input. For example, the F_1-measure for English recognized input dropped 9%-37% compared to the transcribed input, and the F_1-measure for German recognized input dropped 13%-44% compared to the transcribed input. Given that the word accuracy rates of the recognizers were only 56.4% and 51.0% respectively as shown in Table 72, these drops in performance are quite reasonable and demonstrate the robustness of the hybrid analysis approach. The larger relative drops for German are easily explained by the difference in frequency of the "mhm" back channels.

5.2.2 End-to-End Translation

In addition to assessing the performance of the hybrid analyzer components on full speaker turns, we evaluated the output of the complete NESPOLE! translation system. For each English and German input utterance, the NESPOLE! system was used to produce a translation into Italian as well as a paraphrase of the utterance back into the source language. The paraphrases were produced by applying the generator for the source language to the Interchange Format representation of the input utterance produced by the analyzer. For each source language, three bilingual human graders compared the translations and paraphrases produced by the NESPOLE! system to human transcriptions of the input utterances. A grade was assigned for each semantic dialogue unit in an utterance based on how well the meaning of the original semantic dialogue unit was conveyed in the translation output. Grades were not based on fluency or grammaticality except to the extent that the meaning of the original input was altered or corrupted in the output. In addition to grading the translations and paraphrases, the graders also evaluated the output of the speech recognizer as though it were a paraphrase produced by the system. The graders played no role in the development of the NESPOLE! system, and they were not aware of what conditions were used to produce each set of translations. Each grader graded all of the outputs for a particular source language.

	Grade	Description
Acceptable	***Very Good***	• All information in the original semantic dialogue unit is present in the translation and easy to understand.
	Good	• All *important* information in the original semantic dialogue unit is present in the translation. • Minor pieces of information or details that do not change the overall meaning may be missing from and/or added to the translation. • The important information may be expressed poorly but remains understandable.
Unacceptable	***Bad***	• Some of the important information in the original semantic dialogue unit is missing from the translation. • Information or details that change the overall meaning has been added to the translation. • Important information has been translated in a way that is not understandable.
	Very Bad	• Most of the important information in the original semantic dialogue unit is missing from the translation and/or is impossible to understand.

Table 70: Grading scale for end-to-end grading of translations

			very			very	*	no
			good	good	bad	bad	*	lang
Turn	3	*	*****	*****	*****	*****	*	*****
1	okay .	1	x				1	
2	i'm interested in a your summer packages ,	2	x				2	
3	preferably for campsites for	3		x			3	
Transl	Okay. I am interested in your summer packages. Campsites.	*	*****	*****	*****	*****	*	*****

Table 71: End-to-end grading format example

Grading was performed using the 4-point scale described in Table 70, and Table 71 shows an example of a single speaker turn in the grading format used by the graders. For each testing condition, a table such as the one shown in Table 71 was created for each utterance in the test set. The row in Table 71 labeled *Turn* contains a simple numeric ID for the speaker turn under consideration. Each of the following three rows in the example contains the manual transcription for one of the semantic dialogue units in the original speaker utterance. Thus, the original

utterance for the example in Table 71 was "*Okay. I'm interested in a your summer packages, preferably for campsites for*". As this example shows, any disfluency or ungrammaticality present in an utterance was preserved in the manual transcription. The row labeled *Transl* contains the output of the NESPOLE! system for the complete utterance. Because there was no guarantee that the analysis module would correctly segment the utterance into semantic dialogue units, the output for an entire utterance was shown in a single line. Of course, an incorrect segmentation also did not necessarily guarantee a bad translation since the meaning of the utterance may still have been understandable in the target language output. For each semantic dialogue unit in the original utterance, the graders compared the system output to the manual transcription and assigned a grade of *Very Good*, *Good*, *Bad*, or *Very Bad* based on their own judgment of how the translation or paraphrase fit the criteria for meaning preservation listed in Table 70.

Our primary concern in the end-to-end evaluation was whether or not the meaning of the original utterance could be understood in the output of the translation system. When the translation for a semantic dialogue unit was given a grade of *Very Good* or *Good*, the translation was considered *Acceptable* because all important information in the original semantic dialogue unit was conveyed understandably in the translation. When a grade of *Bad* or *Very Bad* was assigned, the translation was considered *Unacceptable*. For the purpose of this evaluation, we were interested in the percentage of semantic dialogue units that received an *Acceptable* grade.

English	German
56.4%	51.0%

Table 72: Speech recognition word accuracy rates for the English and German Travel end-to-end evaluation

	English Output	Italian Output
Speech Recognition Hypotheses	66.7%	--
Translation from Speech Recognition Hypotheses	50.4%	50.2%
Translation from Transcription	68.1%	69.7%

Table 73: Acceptable end-to-end translation for English Travel input

	German Output	Italian Output
Speech Recognition Hypotheses	61.6%	--
Translation from Speech Recognition Hypotheses	44.0%	53.4%
Translation from Transcription	39.7%*	51.7%*

Table 74: Acceptable end-to-end translation for German Travel input

Table 72 shows the word accuracy rates of the English and German speech recognizers on the test set data used in the end-to-end evaluation of the NESPOLE! Travel systems. Table 73 and Table 74 show the percentage of semantic dialogue units that received an *Acceptable* grade for English and German input respectively. The columns labeled *English Output* and *German Output* contain the grades for the paraphrases back into the source language, and the columns labeled *Italian Output* contain the grades for the translations into Italian. The results shown in the tables were computed using a majority vote among the grades assigned by the three graders for each source language. Under the majority-vote grading scheme, the translation of a semantic dialogue unit was considered to be *Acceptable* if the majority of the graders (in this case 2 out of 3) graded it as such.

As mentioned above, the output of the automatic speech recognizers was graded as though it were a paraphrase produced by the full translation system. The rows in Table 73 and Table 74 labeled *Speech Recognition Hypotheses* show the percentage of *Acceptable* grades for the automatic speech recognition output. As these results show, speech recognition errors can be a major source of *Unacceptable* translations since important components of meaning that are lost during speech recognition cannot be recovered during analysis. For example, if the name of a location were misrecognized, there would be no way for the analyzer to include that name in the Interchange Format representation. Thus, the grades for the speech recognition hypotheses provide a rough upper bound on the performance that can be expected from the translation system when translating the output of the automatic speech recognizer.

The rows labeled *Translation from Speech Recognition Hypotheses* contain the percentage of paraphrases and translations produced by the NESPOLE! translation system that were graded as *Acceptable* when speech recognition hypotheses were used as input. These grades reflect the combined performance of the speech recognizer, the analyzer, and the generator. The rows

labeled *Translation from Transcription* contain the percentage of *Acceptable* grades when manual transcriptions of the utterances were used as input. These grades abstract away from speech recognition errors and reflect the combined performance of the analyzer and the generator. Of course, given the word accuracy rates of the speech recognizers shown in Table 72, one would expect that the grades on transcribed input would be much higher than on recognized input. The analyzer performance results for English and German as well as the end-to-end grading results for English input support this expectation. However, the grading results for German input appear to indicate that performance on recognized input was actually higher than performance on transcribed input.

This anomalous result can be attributed to inconsistencies in the grading of back channels such as "mhm" that were especially prevalent in the German dialogues. In order to determine the reason for this anomaly, we examined the outputs of the translation system for German-to-German paraphrasing more closely. When the transcriptions were used as input to the translation system, each occurrence of "mhm" in the transcript was parsed by the cross-domain grammar as an *acknowledge* domain action, and "Okay" was generated in the output. According to the definitions of the domain actions and annotation conventions used for the NESPOLE! data, back channels such as "mhm" were annotated with the *acknowledge* domain action. Thus, the translation of "mhm" as "Okay" should have received a grade of *Acceptable* since this was exactly the behavior expected of the system. On the other hand, when the automatic speech recognition hypotheses were used as input, many instances of "mhm" were misrecognized as noise, and thus nothing appeared in the speech recognition hypotheses where "mhm" should have been. In such cases, the semantic dialogue unit containing "mhm" received an empty translation (i.e. no output) from the translation system. These empty translations should have been assigned a grade of *Unacceptable* since an important piece of information, namely the acknowledgement, was dropped. However, the graders who graded the paraphrases and translations for the German input assigned grades exactly opposite to those that should have been assigned. The graders graded "Okay" as *Unacceptable* and graded an empty translation as *Acceptable*. If we eliminate all of the turns for which the utterance was "mhm" from the German input, we can get a better picture of the relative performance using German translation from transcription and from speech recognition hypotheses. When the "mhm" turns are eliminated, the percentage of *Acceptable* German-to-German paraphrases becomes 60.6% for *Translation from*

Transcription and 51.2% for *Translation from Speech Recognition Hypotheses*. These results are clearly in line the results for English input and agree with our expectation that the translation system should perform better on transcribed input.

The results of the end-to-end evaluation also provide some indication that it is not always necessary to produce the canonical Interchange Format representation found in the manually annotated data in order to provide an acceptable translation. The recall measures shown in Table 68 and Table 69 indicate the percentage of semantic dialogue units from the test data for which the canonical representations in the annotated data were present in the analyzer output. The grading results in Table 73 and Table 74 indicate the percentage of the domain actions from the test data for which the output of the translation system was *Acceptable*. The recall and grading results are comparable because each measure was computed by comparing the output of the analyzer or full translation system for a whole turn against the semantic dialogue units present in the annotated test set data. For recall, the domain actions and arguments produced by the analyzer were compared with those from the annotated data to check if the system output for the turn included the annotated Interchange Format components for each semantic dialogue unit. For end-to-end grading, the target language text for the whole turn was compared with the original text for each semantic dialogue unit to determine if the translation (or paraphrase) preserved the meaning of each semantic dialogue unit. For each test set, the performance of the analyzer on identifying domain actions and top-level arguments was lower than the end-to-end grading results. For example, the English analyzer achieved a recall of 53.0% for domain actions and 40.9% for top-level arguments on manually transcribed input. Nevertheless, 68.1% of the semantic dialogue units were graded *Acceptable* in the English paraphrases, and 69.7% of the semantic dialogue units in the Italian translations were graded *Acceptable*.

It may still not be clear if the performance levels attained by the classifiers and end-to-end translation system are in some sense "good enough." An evaluation described in [Costantini *et al.*, 2002] of dialogue success rate using the NESPOLE! system sheds some light on this question. In the study, naïve users were given a set of goals relevant to the Travel & Tourism domain, such as identifying a hotel in the area with certain properties and convenient to certain activities. The users attempted to accomplish their assigned goals using the NESPOLE! translation system. The translation system used in the study included an earlier version of the hybrid analyzer than that used in the end-to-end evaluation described in this section, so the

results are of course not directly comparable. However, it is safe to assume that the performance of the NESPOLE! system would only have improved with additional development effort after the study. The study found that 86% of the attempted dialogues were completed successfully (i.e. the user accomplished their goals). We would argue that this result suggests that the end-to-end performance levels described in this section are "good enough" for the intended task.

5.3 Porting NESPOLE! to the Medical Assistance Domain

Following the development of the NESPOLE! system for the Travel & Tourism domain, the entire translation system was ported to the Medical Assistance domain. We evaluated the portability of our hybrid analysis approach within the context of the English translation server during this porting effort. A key question with respect to the portability of the hybrid analysis approach is how the development effort and performance of the hybrid approach compare with the development effort and performance of a purely grammar-based approach that uses handwritten full domain action grammars for analysis. Thus, we conducted an experiment in which both approaches were developed and tested using the same data. Due to limited resources, in terms of both the data and time available for development, the experiment was conducted on a scale that allowed for the development of the hybrid analysis approach as well as grammars written to parse full domain actions based on a small set of development dialogues.

Porting the NESPOLE! translation system from the Travel domain to the Medical domain required several steps. Dialogues representative of the new domain were collected and transcribed ([Burger *et al.*, 2003]). The Interchange Format specification was expanded to include new arguments and concepts ([Levin *et al.*, 2003a]). The vocabulary and language models of the speech recognizers were adapted to the new domain, and the analyzers and generators were updated to support the new expanded Interchange Format specification. The amount of effort required for porting the Interchange Format specification, speech recognizer, and generator is certainly relevant to the portability of the entire NESPOLE! translation approach. However, the development of those components would be necessary regardless of the analysis approach used and may therefore be considered constant with respect the portability of our hybrid analysis approach. Therefore, our portability experiment focused only on the development effort directly related to porting the analysis module. The two major elements of porting the analysis module were data annotation and grammar development. Annotated data

was used for reference during grammar writing and for training classifiers. Using a small set of English development dialogues, we measured the effort required to develop the resources for our hybrid analysis approach and compared the effort with that required for developing a full domain action grammar for a purely grammar-based approach.

5.3.1 Medical Domain Data

The data collection for the Medical domain was described in Section 1.5.5. As described there, monolingual dialogues between two native English speakers communicating using Microsoft® NetMeeting were recorded. All of the speakers who participated in the English data collection were doctors or nurses. The dialogues included scenarios in which a patient with either flu-like symptoms or chest-pain was seeking medical advice from a doctor. The "patients", who role-played by doctors or nurses, were provided with background information for their scenario (e.g. symptoms, age, risk factors, medical history, etc.), but the dialogues were otherwise unscripted. The content (i.e. semantic dialogue units, vocabulary, etc.) of the client sides of the Medical domain dialogues used for analyzer development in our portability experiment is shown in Table 80 (Section 5.3.4).

For the purpose of our portability experiment, the Medical domain dialogues were divided into three sets, each of which was used for a different aspect of development and evaluation. The first, and largest, data set was the Interchange Format Development set. Expanding the Interchange Format specification required the availability of dialogues representative of the new domain from which new domain actions, concepts, arguments, and values could be identified. The dialogues were tagged and retagged as the Interchange Format specification was updated to reflect the new domain. Thus, the primary purpose of the Interchange Format Development set was to support the expansion of the Interchange Format specification to cover the Medical domain. The Interchange Format Development set for English contained 8 dialogues that were originally collected monolingually in English. In addition, the set included 4 Italian dialogues that were manually translated into English as well as a small set of manufactured examples for addressing specific aspects of Interchange Format expansion. The Interchange Format Development set was also used for training the classifiers used by the hybrid analyzer in the NESPOLE! system.

The second data set was the Grammar Development set, which contained 4 English dialogues. The Grammar Development set served several roles in our portability experiment. First, after the Interchange Format specification for the Medical domain was stabilized, the Grammar Development set was used to measure the time required for annotating dialogues with Interchange Format representations. The Grammar Development set was also used to measure the times required for developing the grammars used by the hybrid analysis approach and the full domain action grammars for the purely grammar-based approach. Finally, the Grammar Development set was used as training data for the classifiers in the hybrid analysis approach.

The final data set extracted from the Medical domain dialogues for English was the Evaluation set, which consisted of 2 English dialogues. The client-side data from the Evaluation set consisted of 40 speaker turns containing a total of 78 semantic dialogue units. The Evaluation set was held out from the porting process as an unseen test set. It was not used in any way for system development. Following the completion of all grammar development, the Evaluation set was annotated with Interchange Format representations to allow for detailed evaluations of the analyzer components.

5.3.2 Data Annotation

One key element of development that affects the portability of an analysis approach is the amount of time required for annotating data. The process of annotating data with Interchange Format representations may be broken down into three phases. First, the utterances in the dialogue must be segmented into semantic dialogue units. After the semantic dialogue units have been identified, each semantic dialogue unit can be assigned a domain action. Finally, a complete Interchange Format representation requires that each semantic dialogue unit be annotated with arguments (including top-level arguments and values as well as any subarguments).

Table 75 illustrates the three steps required for annotating an utterance with Interchange Format representations. The *Utterance* row shows an example utterance in the format used for manually transcribing recorded input. The section labeled *Semantic Dialogue Unit Boundaries* shows how the utterance would be divided into 4 semantic dialogue units during the first phase of annotation. The *Domain Actions* section of the table shows the assignment of a domain action to each semantic dialogue unit. Finally, the section of the table labeled *Arguments* shows the annotation of each semantic dialogue unit with its corresponding arguments.

Utterance (Client)	*hi doctor . I'm <uh> Elizabeth . I'm% just having some headaches and some fever today . <B> wondering if you could +/h=/+ give me some advice*
Semantic Dialogue Unit Boundaries	*hi doctor .*
	I'm <uh> Elizabeth .
	I'm% just having some headaches and some fever today .
	<B> wondering if you could +/h=/+ give me some advice
Domain Actions	*hi doctor .* **c:greeting**
	I'm <uh> Elizabeth . **c:introduce-self**
	I'm% just having some headaches and some fever today . **c:give-information+experience+health-status**
	<B> wondering if you could +/h=/+ give me some advice **c:request-information+feasibility+action+information-object**
Arguments	*hi doctor .* c:greeting **(greeting=informal_hello, listener=(person-title=dr))**
	I'm <uh> Elizabeth . c:introduce-self **(who=(given-name=name-elizabeth))**
	I'm% just having some headaches and some fever today . c:give-information+experience+health-status **(experiencer=i,** **time=(relative-time=today),** **health-status=(operator=conjunct,** **[(headache, quantity=some), (fever, quantity=some)]))**
	<B> wondering if you could +/h=/+ give me some advice c:request-information+feasibility+action+information-object **(info-object=(advice, quantity=some), feasibility=feasible,** **who=you, action=e-give-4, to-whom=i)**

Table 75: Example of annotation steps required for tagging an utterance with Interchange Format representations

An important advantage of our hybrid analysis approach with respect to portability is that a full Interchange Format annotation is not required for training the classifiers. Because the features used by the semantic dialogue unit boundary detector (described in Section 2.4.2.2) are based only on words and argument parse output, only the first phase of annotation is required for training the segmentation classifier. Training the speech act and concept sequence classifiers requires only the domain action because all of the argument-based features used by the classifiers are extracted from automatically produced argument parses, not annotated arguments. Furthermore, it is possible to develop the argument grammars used in the hybrid analysis approach based on the Interchange Format specification without the need for additional data tagged with arguments. Developing argument-level grammars based on the Interchange Format

specification is feasible because the set of arguments is relatively small. As mentioned in Section 2.2, the final specification for the combined NESPOLE! Travel and Medical domains contained only 227 top-level arguments. Thus, assuming that a stable Interchange Format specification has been defined, only the first two phases of annotation are required to produce the resources necessary for the hybrid analysis approach.

On the other hand, the development of full domain action grammars generally requires fully annotated data. Although it would be possible in principle to develop domain action grammars based only on the Interchange Format specification, doing so is infeasible in practice due to the extremely large number of possible legal domain actions and the large number of ways that sets of arguments may be licensed by the domain actions. Therefore, in practice, the development of domain action grammars must be focused based on observation of domain actions in data representative of the desired domain of coverage. Since domain action grammar rules typically include top-level arguments on the right-hand side, the data must also be annotated with arguments in order to allow grammar writers to develop rules using the arguments observed with the domain actions. Thus, writing full domain action grammars requires all three stages of annotation.

After the expanded Interchange Format specification for the Medical domain was stabilized based on the Interchange Format Development data set, we measured the data annotation time for the Grammar Development set. Since full Interchange Format representations were not required for training the classifiers used in the hybrid analysis approach, it was important to measure the time required for tagging only the domain action as well as the time required for tagging the complete Interchange Format representation. In order to measure both times while minimizing duplication of effort, the data was annotated in three stages corresponding to the three phases mentioned above. The Grammar Development dialogues were first segmented into semantic dialogue units, then tagged with domain actions, and finally tagged with arguments. The annotators noted the tagging time for each stage in the database file containing the tagged data. Segmentation annotation time is simply the time spent on the first stage. The time required for annotating domain actions for training classifiers in the hybrid analysis approach is the sum of the times for the first two stages, and the time for tagging with complete Interchange Format representations is the sum of all three stages.

	Utterances	Semantic Dialogue Units
Dialogue 1	21	55
Dialogue 2	25	74
Dialogue 3	37	89
Dialogue 4	12	28
Total	95	246

Table 76: Data set sizes for client-side utterances in the Grammar Development set

	Segmentation*	Domain Actions	Arguments
Dialogue 1	5	25 (30)	225 (255)
Dialogue 2	5	69 (74)	150 (224)
Dialogue 3	8	31 (39)	75 (114)
Dialogue 4	3	25 (28)	45 (73)
Total	21	150 (171)	495 (666)

Table 77: Annotation times (minutes) for client-side utterances in the Grammar Development set

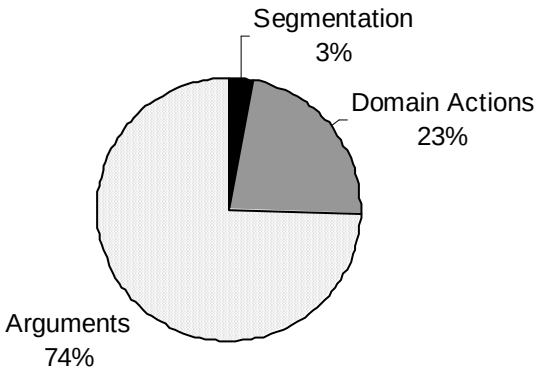

Figure 45: Distribution of annotation times for Grammar Development set client-side utterances

Due to time limitations at the end of the NESPOLE! project and because the focus of the English translation engine in the NESPOLE! system was on the client (a sick traveler for the Medical domain), only the client-side utterances were initially annotated with Interchange Format representations during the port to the Medical domain. Thus, Table 76 lists the sizes of the client side of the Grammar Development dialogues (i.e. number of utterances and semantic dialogue units), and Table 77 shows the tagging times for each stage of annotation for the client side of the dialogues. The first number in each column of Table 77 shows the annotation time for the individual stage, and the numbers in parentheses show the cumulative annotation time including preceding stages. Figure 45 shows the distribution of the client-side annotation times for the Grammar Development set. Although only the client side was tagged with Interchange Format representations, the full dialogue was segmented into semantic dialogue units. Thus, the times reported for segmentation in Table 77 are half of the times noted by the annotators. The client and agent (doctor) sides of the dialogues had a similar number of utterances (client: 95, agent: 101) and a similar number of semantic dialogue units per utterance (client: 2.6, agent: 2.8). Since the client-side utterances made up about 48% of the utterances in the data and about 47% of the semantic dialogue units, the estimate of half the tagging time is reasonable. As the data in the table show, the first phase of annotation, segmenting utterances into semantic dialogue units, required a very small portion (about 3%) of the total annotation time. Additionally, annotating the data with domain actions required much less time than annotating with arguments. In fact, the first two phases of annotation, those required for the hybrid analysis approach, required only about one quarter of the total time required for tagging the data with complete Interchange Format representations.

5.3.3 Grammar Development

Writing grammars is the second major aspect of analyzer development that affects the portability of the analysis approach. For our hybrid analysis approach, argument and pseudo-argument grammars had to be developed. For the purely grammar-based analysis approach, a full domain action grammar designed to parse utterances all the way to the domain action level had to be written. After data annotation was completed, we measured the time required for grammar development for each analysis approach. Since the time available for porting the NESPOLE! system to the Medical domain was limited and the hybrid analysis approach served as the

analysis module for the English translation server, it was necessary to impose restrictions on the intended coverage of the full domain action grammar. Thus, the full domain action grammar was only written to cover domain actions found in the client side of the dialogues in the Grammar Development set.

In order to measure the effort required for porting the analysis grammars to a new domain, we recorded the time spent developing the argument, pseudo-argument, and full domain action grammars for the Medical domain. The grammar writers were allowed to start grammar development using the existing grammars from the final version of the NESPOLE! Travel & Tourism analyzer in order to minimize duplication of effort and fit full grammar development into the limited time available. The Travel grammars were expanded as necessary to provide coverage for the Medical domain.

Because we wanted to isolate the time spent on each grammar as much as possible, the argument, pseudo-argument, and full domain action grammars were developed sequentially rather than in parallel, and the grammar writers logged the time spent developing each grammar. The sequence of grammar development was determined by dependencies among the grammars. The argument grammar for the Medical domain was developed first. As described in Sections 1.5.4 and 2.2, Interchange Format arguments are represented using a tree-structured feature-value representation. Thus, in addition to the time required for adding rules to cover new top-level arguments for the Medical domain, argument grammar development included time required for writing rules for new Medical values and subarguments. Since the phrases parsed by the pseudo-argument grammar sometimes contained arguments, the pseudo-argument grammar was developed second. Development of the pseudo-argument grammar primarily involved writing top-level rules to parse groups of similar phrases, which sometimes included arguments provided by the rules in the argument grammar. Finally, since argument-level grammar rules are required for parsing full domain actions and the phrases parsed by the pseudo-argument grammar can be useful for domain action parsing, the full domain action grammar was written last. Thus, the grammar development time for the purely grammar-based analysis approach would be the sum of the times for developing all three subgrammars.

	Grammar Development Set	Additional	Total	Percent of Total
Argument Grammar	31.25	46.25	77.5	70.6%
Pseudo-Argument Grammar	8.5	0.5	9	8.2%
Full Domain Action Grammar	23.25	--	23.25	21.2%
Total	55.0	46.75	109.75	

Table 78: Grammar development times (hours) for the NESPOLE! Medical Assistance domain

Table 78 shows the grammar development times for the argument, pseudo-argument, and full domain action grammars used in our portability experiment. The grammar writers first wrote argument and pseudo-argument grammars to cover the dialogues in the Grammar Development data set. Although they were not restricted from generalizing rules to include phrasings not seen in the data when they noticed opportunities to do so, they were instructed not to try to develop rules to cover aspects of the NESPOLE! Medical Assistance domain not observed in the Grammar Development dialogues. The times in the *Grammar Development Set* column of Table 78 show the initial development times for the argument and pseudo-argument grammars based on data in the Grammar Development set. After development on the Grammar Development set was complete, the grammar writers spent additional time, shown in the *Additional Time* column, expanding the coverage of the argument grammar and pseudo-argument grammars. The vast majority (more than 90%) of the additional time for the argument grammars was spent developing rules based on the Interchange Format specification. A small amount of the time was also spent debugging the grammars. Finally, after argument and pseudo-argument grammar development were complete, the grammar writers wrote rules for the full domain action grammar to cover the domain actions observed in the Grammar Development set dialogues. The grammar writers used the argument-level rules from the fully developed argument grammar and phrase rules from the pseudo-argument grammar while writing the full domain action grammar. The time shown in Table 78 for the full domain action grammar was spent writing the top-level domain action rules using the existing grammars for lower level rules. The times in the *Total* column of Table 78 show the sum of the development times spent on each grammar based on the Grammar Development set and including any additional time.

5.3.4 Evaluation

In this section, we compare the performance of the hybrid analysis approach and the purely grammar-based approach. We evaluated the performance of both approaches on the tasks of argument parsing and domain action classification. In addition, we evaluated the end-to-end translation performance of the NESPOLE! translation system for the Medical Assistance domain using each approach for the analysis module.

We tested five different versions of the analysis module. All of the analyzers made use of the cross-domain grammar. The first version of the analysis module used a purely grammar-based approach with the full domain action grammar developed as described above, and parsing was performed using the SOUP parser (Section 2.3.1). We will refer to this analyzer as the *FullDA* analyzer. We also tested four different versions of our hybrid analyzer using different combinations of argument grammars and training data.

		Training Set	
		AllData	**GraDevData**
Grammars	**AllDevGra**	AllDevGra+AllData	AllDevGra+GraDevData
	GraDevGra	GraDevGra+AllData	GraDevGra+GraDevData

Table 79: Hybrid analysis approach configurations tested in the portability experiment

Table 79 summarizes the resources used by each of the four different hybrid analyzer versions tested. Two different versions of the argument and pseudo-argument grammars are listed in the *Grammars* rows of the table. The *AllDevGra* grammar refers to the final versions of the argument and pseudo-argument grammars after development on the Grammar Development set data and expansion based on the Interchange Format specification. These were the grammars used during the development of the full domain action grammar. The *GraDevGra* grammar refers the versions of the argument and pseudo-argument grammars that were developed only on the Grammar Development set data without any additional development. Table 79 also lists two different training data sets in the *Training Set* columns. The *AllData* training set included data from the Grammar Development set as well as data from the Interchange Format Development set. The *AllData* set provided 3302 training examples for the segmentation classifier and 1096

training examples for the speech act and concept sequence classifiers. The *GraDevData* training set included only data extracted from the Grammar Development set. The *GraDevData* training set provided 687 training examples for the segmentation classifier and 241 training examples for the speech act and concept sequence classifiers. Table 79 also shows the names that will be used to refer to the four different versions of our hybrid analyzer created by combining the two different grammars with the two different training sets.

	GraDevData	AllData
Speaker Turns	95	387
Semantic Dialogue Units	246	1110
Domain Actions	70	168
Speech Acts	18	26
Concept Sequences	53	129
Individual Concepts	33	49
Top-Level Arguments	56	91
Words	328	669

Table 80: Contents of the *GraDevData* and *AllData* training sets used in the portability experiment

Since the full domain action grammar for the *FullDA* analyzer was only developed to cover the client side of the Grammar Development set dialogues, all of the classifiers used in the hybrid analyzers were also trained only on client-side semantic dialogue units. Table 80 shows the contents of the client-side data used in the *GraDevData* and *AllData* training sets. All of the classifiers were implemented using TiMBL with the IB1 k-nearest neighbor algorithm, and all of the hybrid analyzers used the same parameter settings and feature sets. The segmentation classifiers used the 10 features described in SCTION-REF with *k*=5, Gain Ratio feature weighting, and unweighted voting. The speech act and concept sequence classifiers used *k*=1, Gain Ratio feature weighting, and unweighted voting. The feature sets for the speech act and concept sequence classifiers were selected using a process similar to the one described in Section 4.5 based on the classifiers for the *AllDevGra+AllData* analyzer. The only difference in the feature set selection process was that a feature set was only removed from the input vector if its removal produced an increase in accuracy (rather than an increase or non-significant decrease as in the previous experiment). The probabilities computed by the Rainbow naïve Bayes word

bigram models (*SAProbs* and *ConcProbs*) were not considered for inclusion in the feature sets for reasons discussed in Section 4.2.6. Also, since only client-side data was used for training the classifiers, the *Side* feature was unnecessary. Thus, we considered only the argument grammar labels (*ArgGra*), the pseudo-argument grammar labels (*PseudoGra*), the top 250 words based on mutual information with the class (*Words*), and the top-level Interchange Format arguments (*IFArgs*) for inclusion in the input feature vector. The speech act (*SA*) was also considered for the concept sequence classifier. After the feature set selection process, the *Words*, *IFArgs*, and *PseudoGra* feature sets were used in the input vector for the speech act classifiers. The concept sequence classifiers used the *IFArgs*, *PseudoGra*, and *SA* feature sets.

Analyzer	Annotation	Grammar Development	Total	Percent of FullDA
FullDA	11.10	109.75	120.85	100%
AllDevGra+AllData	12.86	86.50	99.36	82%
AllDevGra+GraDevData	2.85	86.50	89.35	74%
GraDevGra+AllData	12.86	39.75	52.61	44%
GraDevGra+GraDevData	2.85	39.75	42.60	35%

Table 81: Development times (in hours) for analyzers tested in the portability experiment

Table 81 lists the development time (in hours) for each of the 5 analyzers tested. For the *FullDA* analyzer, the annotation time was the time required for tagging the Grammar Development dialogues with full Interchange Format representations, and the grammar development time was the sum of all grammar development times for the Medical domain, including argument, pseudo-argument, and full domain action grammar development. For each of the hybrid analyzers, the annotation time included the time required for identifying the semantic dialogue unit boundaries and tagging the semantic dialogue units with domain actions. The time for annotating the *GraDevData* training set was measured directly as described in Section 5.3.2. We estimated the time required for annotating the Interchange Format Development set with domain actions based on the average annotation time per semantic dialogue unit for the Grammar Development set. Based on this estimate, annotating the additional 864 semantic dialogue units from the Interchange Format Development set would

have taken approximately 10.01 hours. The annotation time for the *AllData* training set reported in Table 81 is the sum of the annotation time for the *GraDevData* training set and the estimated annotation time for the Interchange Format Development set data. The development times in Table 81 provide one indication of the improved portability of our hybrid analysis approach. The most fully developed hybrid analyzer (*AllDevGra+AllData*) required about 18% less development time than the purely grammar-based analyzer (*FullDA*), and the minimally developed hybrid analyzer (*GraDevGra+GraDevData*) required about 65% less development time.

Test Set Name	Description
GraDevTCT	Manually transcribed utterances from the Grammar Development set
EvalTCT	Manually transcribed utterances from the Evaluation set
EvalSR	Automatically recognized utterances from the Evaluation set

Table 82: Test sets used in the portability experiment

The performance of each of the five analyzers was evaluated using the three different test sets listed in Table 82. The *GraDevTCT* test set included manually transcribed utterances from the Grammar Development set, and the *EvalTCT* test set contained manually transcribed utterances from the unseen Evaluation set. Finally, the *EvalSR* test set included automatic speech recognition output for the utterances in the unseen Evaluation set. Each of the test sets contained only the client-side utterances from the dialogues in the respective data sets, and each utterance corresponded to a complete speaker turn. The *Eval* test sets contained 40 speaker turns with a total of 78 semantic dialogue units. In each of the test sets, there were a few turns that contained only noise and no real linguistic content. Since the recognizer did not erroneously produce any words for those turns (which could have resulted in the insertion of spurious meaning in the paraphrases), those turns were excluded from the end-to-end grading results presented later in this chapter. Thus, the *GraDevTCT* test set contained 241 semantic dialogue units with real linguistic content, and the *EvalTCT* and *EvalSR* test sets contained 76 semantic dialogue units with real linguistic content. We conducted our evaluation by running the full utterances, rather than individual semantic dialogue units, through the analysis and generation modules of the

NESPOLE! English translation server as in the end-to-end evaluation for the NESPOLE! Travel domain described in Section 5.2.

Since the input to the analyzers consisted of full speaker turns, there was no guarantee that the analyzers would correctly segment the turns into semantic dialogue units nor that the analysis for each turn would even contain the correct number of semantic dialogue units. Therefore, the performance measures for argument parsing and domain action classification reported in the following sections were computed on the basis of full turns without regard to semantic dialogue unit boundaries. The precision measures thus reflect the percentage of Interchange Format elements (i.e. top-level arguments, domain actions, etc.) in the analyzer output for the turn that were present in the manual annotation for the turn. Likewise, the recall measures reflect the percentage of elements from the manually annotated turn that were present in the analyzer output for the whole turn. Additionally, the results were computed based on the final output of the analysis module. Thus, the results for the hybrid analyzers included the use of the cross-domain grammar as well as the application of the fallback strategy described in Section 4.6.1 to ensure that legal Interchange Format representations were produced.

	Domain Action Component	**Coverage**
Domain Action	*give-information+experience+health-status*	16.8%
Speech Act	*give-information*	63.9%
Concept Sequence	*+experience+health-status*	17.0%

Table 83: Coverage of *give-information+experience+health-status* in the *AllData* training data set

	Domain Action	**Speech Act**	**Concept Sequence**	**Individual Concepts**
Precision	0.198	0.605	0.198	0.216
Recall	0.205	0.628	0.205	0.340
F_1	0.201	0.616	0.201	0.264

Table 84: Performance of the baseline analyzer on the *EvalTCT* test set

In order to provide some context for the results of the evaluations described in the following sections, we first look at the performance that could be achieved using a very simple baseline classifier that always returns the most common domain action from the training data. For the purpose of establishing this baseline, we replaced the domain actions produced by the *AllDevGra+AllData* hybrid analyzer with the most frequent domain action found in the *AllData* training set, and we examined the performance of this baseline analyzer on the *EvalTCT* test set. *give-information+experience+health-status* was the most frequent domain action in the *AllData* training set. Table 83 shows the coverage of this domain action in the *AllData* training data set, and shows the precision, recall, and F_1-measure on the *EvalTCT* test set when this domain action is used for every semantic dialogue unit found by the *AllDevGra+AllData* analyzer. As the results in the following sections will show, the *AllDevGra+AllData* hybrid analyzer provides much better performance on the *EvalTCT* test set than this simple baseline analyzer.

5.3.4.1 Argument Parsing

		Test Set		
Analyzer		**GraDevTCT**	**EvalTCT**	**EvalSR**
FullDA	**P**	0.764	0.549	0.428
	R	0.843	0.532	0.411
	F_1	0.801	0.540	0.419
AllDevGra + AllData	**P**	0.790	0.586	0.490
	R	0.749	0.519	0.443
	F_1	0.769	0.550	0.465
AllDevGra + GraDevData	**P**	0.788	0.580	0.496
	R	0.753	0.506	0.405
	F_1	0.770	0.541	0.446
GraDevGra + AllData	**P**	0.747	0.541	0.467
	R	0.715	0.462	0.399
	F_1	0.731	0.498	0.430
GraDevGra + GraDevData	**P**	0.750	0.545	0.457
	R	0.719	0.462	0.373
	F_1	0.734	0.500	0.411

Table 85: Argument parsing performance on identification of top-level arguments for analysis of full speaker turns in the portability experiment

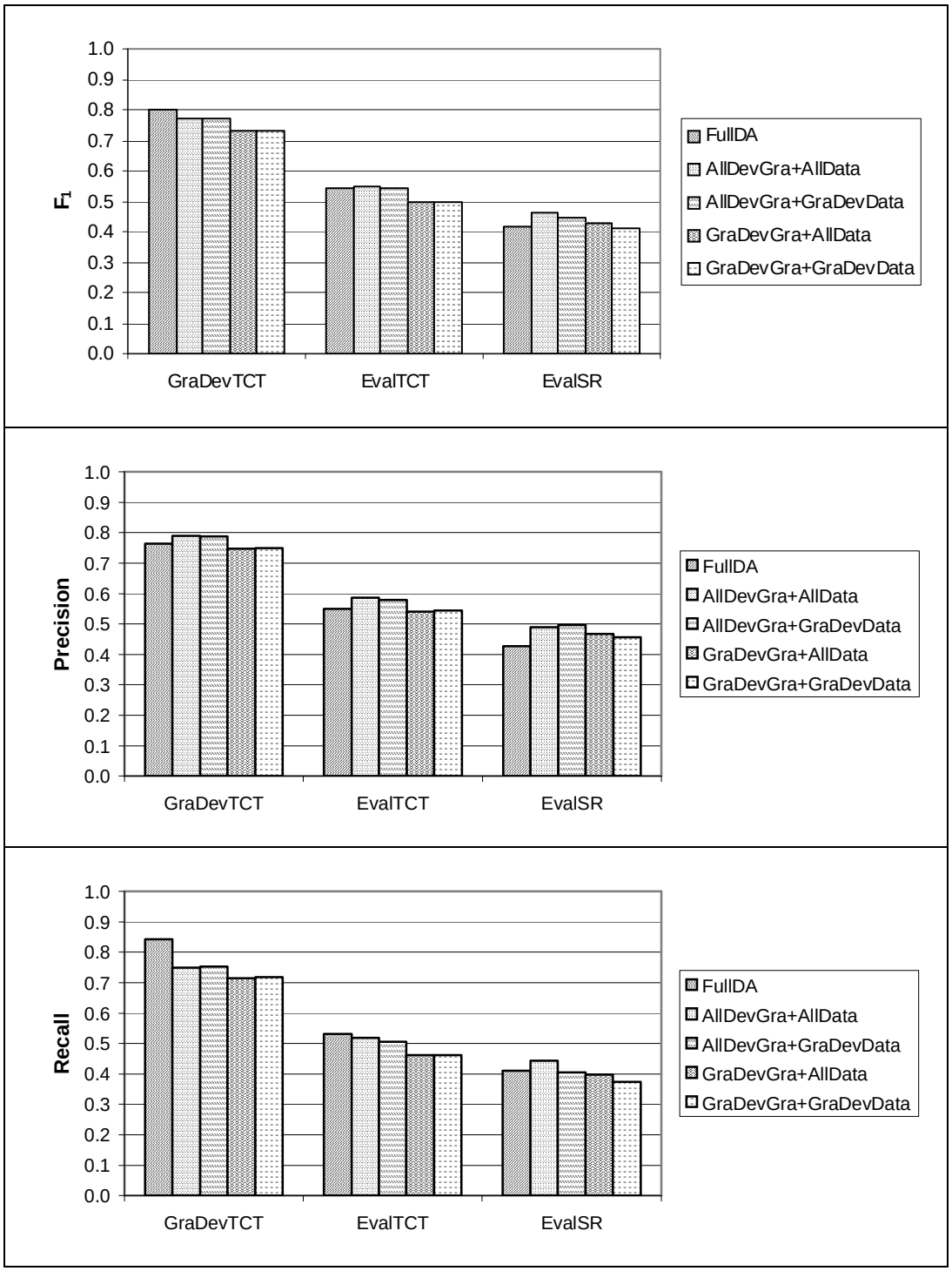

Figure 46: Argument parsing performance on identification of top-level arguments for analysis of full speaker turns in the portability experiment

Table 85 contains the results for argument parsing for the five analyzers tested in the evaluation based on full-turn input. The table shows the precision, recall, and F_1-measure for identification of top-level arguments for each analyzer on each test set. The performance measures were computed by comparing the set of top-level arguments present in the manually annotated Interchange Format representations for each turn with the set of top-level arguments produced by an analyzer. Each of the performance measures was computed over the complete test set (as opposed to averaging the measures for individual turns). Figure 46 illustrates the performance measures reported in the table graphically.

The set of columns on the left side of each chart shows the performance of the analyzers on the *GraDevTCT* test set, which contained manually transcribed utterances on which the analyzers were developed. As expected, performance on this test set was highest for all of the analyzers. The middle set of columns shows performance on the *EvalTCT* test set, which included manually transcribed unseen data. The performance of the analyzers was next best on this data set. Finally, the set of columns on the right side of each chart shows analyzer performance on the *EvalSR* test set, which contained automatic speech recognition output for the utterances in the *EvalTCT* test set. Since performance on the *EvalSR* set reflects speech recognition errors as well as analysis errors, it was expected that performance would be worst on this test set.

A closer inspection of the results allows us to make several observations about the performance of the various analyzers. First, the *AllDevGra* hybrid analyzers that used the fully developed argument grammars (*AllDevGra+AllData* and *AllDevGra+GraDevData*) outperformed the *GraDevGra* hybrid analyzers that used the argument grammars developed only on the Grammar Development data (*GraDevGra+AllData* and *GraDevGra+GraDevData*). Since both of the *AllDevGra* hybrid analyzers used the same argument grammars, the best argument parse for each utterance would have been the same for both analyzers. Thus, performance differences between the analyzers should be small and would have been caused by arguments dropped during fallback processing using the Interchange Format specification. The same holds true for the two *GraDevGra* hybrid analyzers.

The most interesting comparisons are those regarding the performance of the purely grammar-based analyzer (*FullDA*) with the hybrid analyzers. We first consider the performance of the analyzers on the *GraDevTCT* test set. Based on the F_1-measure, the *FullDA* analyzer outperformed all of the hybrid analyzers. The higher F_1-measure was the result of higher recall of

top-level arguments. The recall of the *FullDA* analyzer was 9% absolute higher than that of the best hybrid analyzer. However, the precision of the *AllDevGra* analyzers was better, a fact that held true across all three test sets. The superior performance of the purely grammar-based approach on the seen test set was not surprising since the *FullDA* grammars were developed specifically to cover the *GraDevTCT* data. The context provided by the full domain action grammar rules should have led to less ambiguity at the argument level for the *FullDA* analyzer. Nevertheless, the performance of the *AllDevGra* analyzers, which used the same argument grammars as the *FullDA* analyzer without the benefit of domain-action-level rules, was still quite strong. The F_1-measures for those analyzers were only about 3% absolute lower than the *FullDA* grammar.

We next look at the performance of the analyzers on the *EvalTCT* test set. Since the data was unseen, the performance of the analyzers on this test set relative to performance on the *GraDevTCT* test set serves as an indicator of the domain robustness of the analyzers (i.e. robustness to unforeseen in-domain inputs). Although the *FullDA* analyzer again had the highest recall, the advantage over the *AllDevGra+AllData* hybrid analyzer fell from 9% to 3% absolute. Furthermore, the *AllDevGra+AllData* outperformed the *FullDA* analyzer on the F_1-measure. An additional point worth noting is that the *FullDA* analyzer suffered the largest absolute and relative drops in performance on all three measures. These results provide evidence of the superior domain robustness of the hybrid analysis approach over the purely grammar-based approach on the task of argument parsing.

Finally, we consider the performance of the analyzers on the *EvalSR* test set. Since the only difference between this test set and the *EvalTCT* test set was that an automatic speech recognizer generated the inputs, the performance of the analyzers on this test set is indicative of their robustness to speech recognition errors. On the *EvalSR* test set, the *FullDA* analyzer had the worst precision of all the analyzers, and the recall of the *FullDA* analyzer fell below that of the *AllDevGra+AllData* analyzer by about 3% absolute. On the F_1-measure, the *AllDevGra+AllData* outperformed the *FullDA* analyzer by about 4.5% absolute. In fact, 3 of the 4 hybrid analyzers outperformed the *FullDA* analyzer on the F_1-measure. Only the *GraDevGra+GraDevData* analyzer, the least developed hybrid analyzer, had a lower F_1-measure, and the difference was less than 1% absolute. As in moving from the *GraDevTCT* test set to the *EvalTCT* test set, the *FullDA* analyzer again suffered the largest absolute and relative drops in performance. Thus, the

results provide evidence that the hybrid analysis approach is also more robust to speech recognition errors than the purely grammar-based approach for identification of top-level arguments.

5.3.4.2 Domain Action Classification

In this section we present performance results for the *FullDA* analyzer and each of the four hybrid analyzers on the task of domain action identification in full speaker turns. We also present results for the identification of speech acts, complete concept sequences, and individual concepts. The results are presented in the same format as the results for argument parsing presented in the previous section. Thus, for each task, we first present a table containing the precision, recall, and F_1-measure for each of the five analyzers on each of the three test sets, and we then present charts depicting the performance measures graphically. Since the inputs were full speaker turns, the results reflect the performance of the complete online version of the hybrid analyzers, including use of the cross-domain grammar and the fallback strategy to guarantee that legal Interchange Format representations were produced.

As we observed in the results for argument parsing, the performance of all analyzers on the domain action identification tasks was generally best for the *GraDevTCT* test set followed by the *EvalTCT* test set and finally the *EvalSR* test set. Given that this was the anticipated order of increasing difficulty of the test sets, these results were expected. Additionally, the *AllDevGra* analyzers generally outperformed the *GraDevGra* analyzers. This was not surprising since the argument grammars used by those analyzers were more fully developed and provided better argument parses upon which to base the identification of domain actions.

Table 86 contains the performance results for domain action identification for the five analyzers tested in the portability evaluation, and Figure 47 illustrates the performance measures graphically. The performance measures for domain actions were computed by comparing the set of domain actions for each turn from the annotated data with the set of domain actions produced by each analyzer. A domain action produced by an analyzer had to exactly match an annotated domain action in order to be counted as correct. One interesting observation is that each of the *GraDevData* hybrid analyzers outperformed the corresponding *AllData* hybrid analyzer that used the same grammars on the *GraDevTCT* test set. This was most likely due to the fact that the *GraDevData* analyzers could only produce speech acts and concept sequences that were seen in the *GraDevTCT* data, whereas the *AllDevData* analyzers could produce speech acts and concept

sequences not in the *GraDevTCT* data because they were also trained on the Interchange Format Development set. This advantage disappeared for the *AllDevGra+GraDevData* analyzer for both *Eval* test sets and for the *GraDevGra+GraDevData* analyzer for the *EvalSR* test set showing that the larger *AllData* training set provided an advantage on unseen test data.

		Test Set		
Analyzer		**GraDevTCT**	**EvalTCT**	**EvalSR**
FullDA	**P**	0.624	0.339	0.202
	R	0.687	0.513	0.346
	F_1	0.654	0.408	0.255
AllDevGra + AllData	**P**	0.730	0.543	0.384
	R	0.691	0.564	0.423
	F_1	0.710	0.554	0.402
AllDevGra + GraDevData	**P**	0.764	0.524	0.384
	R	0.736	0.564	0.423
	F_1	0.750	0.543	0.402
GraDevGra + AllData	**P**	0.660	0.457	0.372
	R	0.655	0.474	0.410
	F_1	0.657	0.465	0.390
GraDevGra + GraDevData	**P**	0.685	0.488	0.365
	R	0.699	0.539	0.397
	F_1	0.692	0.512	0.380

Table 86: Domain action identification performance for analysis of full speaker turns in the portability experiment

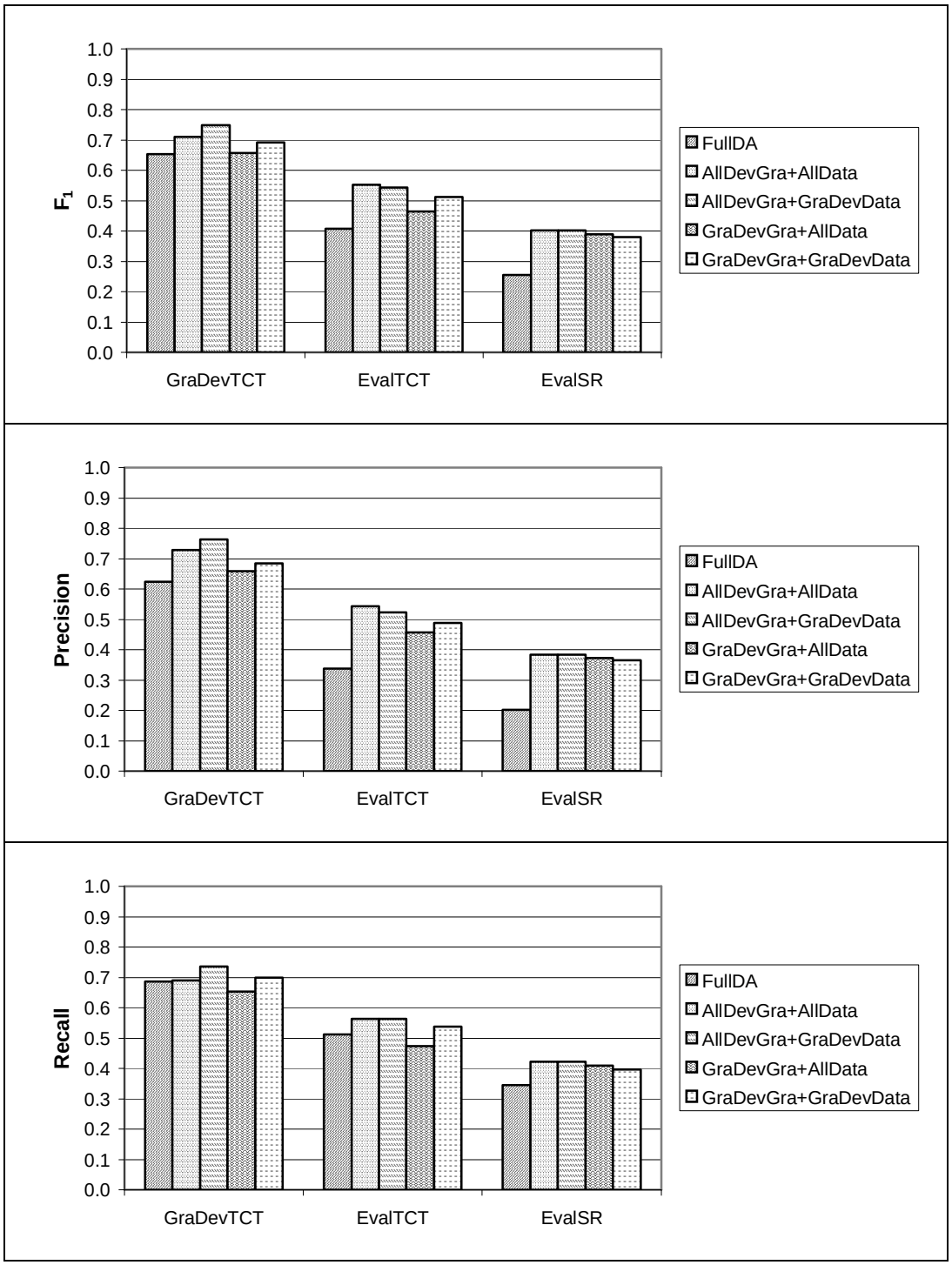

Figure 47: Domain action identification performance for analysis of full speaker turns in the portability experiment

The most important point to note about the performance results for domain action identification is that all of the hybrid analyzers outperformed the purely grammar-based *FullDA* analyzer on the F_1-measure and precision for all three test sets, and at least 3 of the 4 hybrid analyzers provided higher recall. Furthermore, the advantage of the hybrid analyzers over the *FullDA* analyzer grew larger as the test sets became more difficult. For example, the advantage of the *AllGraDev+AllData* hybrid analyzer over the *FullDA* analyzer for the F_1-measure was about 5.5% absolute on the *GraDevTCT* test set, and about 14.5% on the *EvalTCT* and *EvalSR* test sets. The *FullDA* analyzer also exhibited the largest absolute and relative decreases in F_1-measure when moving from seen (*GraDevTCT*) to unseen (*EvalTCT*) data and from manual transcriptions (*EvalTCT*) to automatic recognitions (*EvalSR*). Taken together, these results provide additional evidence of the improved portability of the hybrid analysis approach since it provided superior performance with less development time. Furthermore, the smaller decreases in performance on unseen and automatically recognized data provide evidence that the hybrid analysis approach is more robust than the purely grammar-based approach.

		Test Set		
Analyzer		**GraDevTCT**	**EvalTCT**	**EvalSR**
FullDA	**P**	0.779	0.610	0.542
	R	0.858	0.923	0.910
	F_1	0.816	0.735	0.679
AllDevGra + AllData	**P**	0.884	0.852	0.814
	R	0.837	0.885	0.897
	F_1	0.860	0.868	0.854
AllDevGra + GraDevData	**P**	0.916	0.810	0.729
	R	0.882	0.872	0.795
	F_1	0.899	0.840	0.761
GraDevGra + AllData	**P**	0.861	0.827	0.756
	R	0.854	0.859	0.833
	F_1	0.857	0.843	0.793
GraDevGra + GraDevData	**P**	0.873	0.791	0.729
	R	0.890	0.872	0.795
	F_1	0.881	0.829	0.761

Table 87: Speech act identification performance for analysis of full speaker turns in the portability experiment

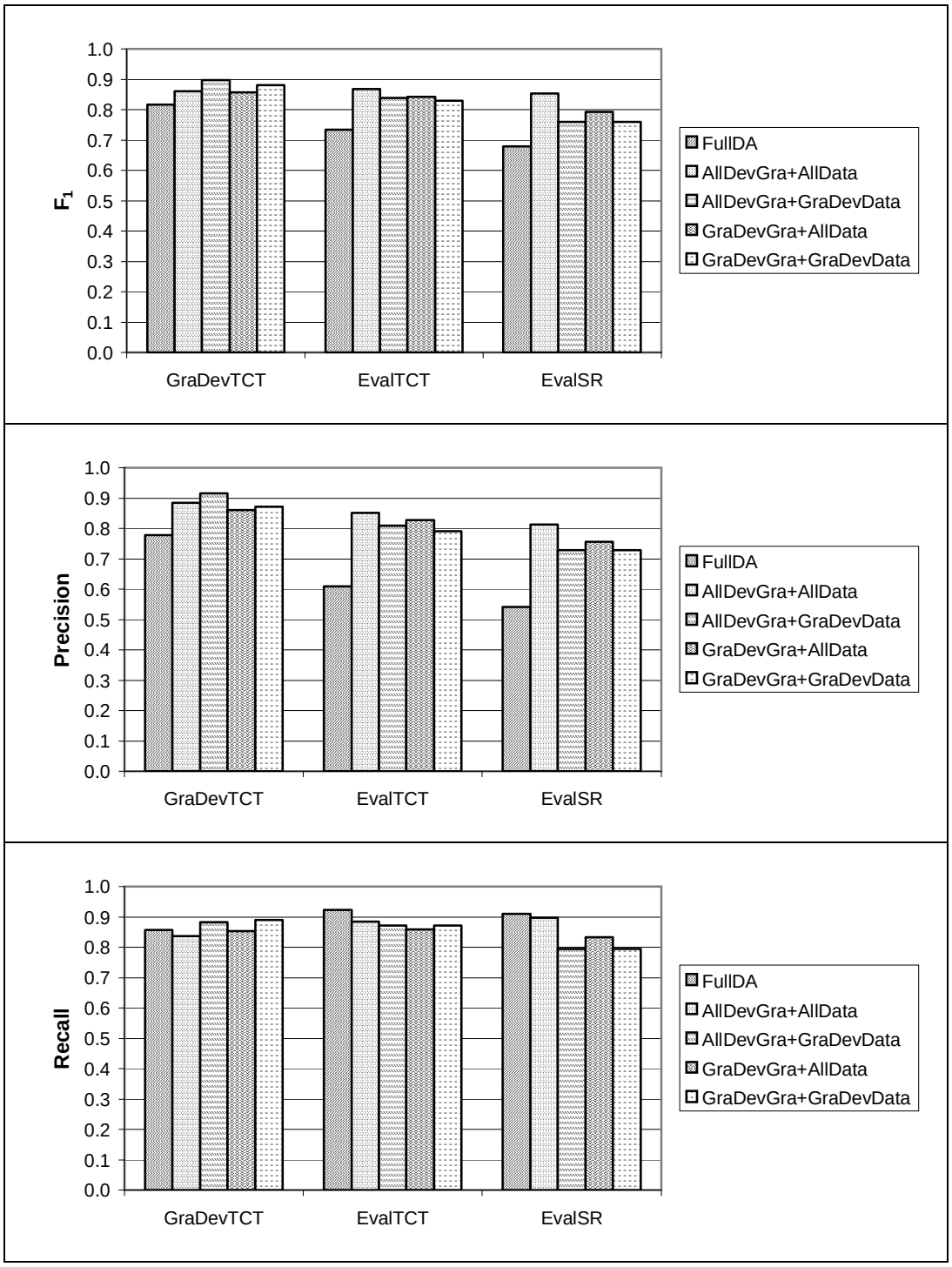

Figure 48: Speech act identification performance for analysis of full speaker turns in the portability experiment

The precision, recall, and F_1-measures for identification of speech acts by each of the analyzers on each of the test sets are listed in Table 87 and illustrated in Figure 48. The performance of all of the analyzers on the speech act identification task was higher than for any other task. This was not surprising since speech acts present the least difficult classification problem, having the smallest number of classes and the least sparse data. As was the case for domain action identification, the *GraDevData* analyzers outperformed the corresponding *AllDevData* analyzers on the *GraDevTCT* test set, but the analyzers trained with more data performed better on the unseen test sets. One other interesting result from speech act identification was that the recall of the *FullDA* analyzer and the *AllDevGra+AllData* hybrid analyzer was actually higher on the unseen *Eval* test sets than on the seen *GraDevTCT* test set. This was likely due to the fact that, on average, the analyzers produced more semantic dialogue units per turn for the *Eval* test sets.

	GraDevTCT	EvalTCT	EvalSR
Annotated Data	2.59	1.95	1.95
FullDA Analyzer	2.85	2.95	3.28
AllDevGra+AllData Analyzer	2.45	2.03	2.15

Table 88: Mean number of semantic dialogue units per turn

Table 88 shows the mean number of semantic dialogue units per turn in the annotated data for each test set as well as for the output of the two analyzers. On the *GraDevTCT* set, the average number of semantic dialogue units per turn produced by the analyzers was similar to the annotated average, with the *FullDA* analyzer somewhat higher than the annotated data and the *AllDevGra+AllData* analyzer slightly lower. On the other hand, the average for both analyzers on the *Eval* test sets was higher than in the annotated data. Although the average for the hybrid analyzer was still very close to the annotated average, the *FullDA* analyzer averaged at least 1 extra semantic dialogue unit per turn for the *Eval* sets. This means that the *FullDA* analyzer essentially had an extra opportunity for each turn to match a speech act from the annotated turn, which probably boosted recall. Of course, it would also explain the severe drops in precision observed for the *FullDA* analyzer since it would mean that, on average, at least one of the speech acts in the analyzer output for each turn could never match the annotated turn. Not surprisingly, this phenomenon of improved recall on the *Eval* sets did not occur for any of the other

classification tasks or for argument parsing. Unlike speech acts, which are basically atomic labels, domain actions and concept sequences may be broken down into smaller components. If any of the components were incorrect, the whole label was considered incorrect. Breaking up a turn into too many semantic dialogue units would be likely to split up concept sequences at best and cause missing or incorrect concepts at worst. Since the performance measures for argument parsing and individual concepts were computed independently of semantic dialogue units, the number of semantic dialogue units in a turn was not a factor in performance.

Two aspects of the speech act identification results provide additional support for the effectiveness of our hybrid analysis approach. First, all of the hybrid analyzers outperformed the *FullDA* analyzer on the F_1-measure and precision across all test sets. Second, the *AllDevGra+AllData* analyzer, the most fully developed hybrid analyzer, performed particularly well on speech act identification. In fact, although the F_1-measure for the other analyzers dropped as the test sets became more difficult, the F_1-measure for the *AllDevGra+AllData* analyzer remained essentially constant.

		Test Set		
Analyzer		**GraDevTCT**	**EvalTCT**	**EvalSR**
FullDA	**P**	0.661	0.364	0.237
	R	0.728	0.551	0.397
	F_1	0.693	0.439	0.297
AllDevGra + AllData	**P**	0.781	0.556	0.407
	R	0.740	0.577	0.449
	F_1	0.760	0.566	0.427
AllDevGra + GraDevData	**P**	0.806	0.536	0.412
	R	0.776	0.577	0.449
	F_1	0.791	0.556	0.429
GraDevGra + AllData	**P**	0.709	0.469	0.407
	R	0.703	0.487	0.449
	F_1	0.706	0.478	0.427
GraDevGra + GraDevData	**P**	0.725	0.500	0.388
	R	0.740	0.551	0.424
	F_1	0.732	0.524	0.405

Table 89: Concept sequence identification performance for analysis of full speaker turns in the portability experiment

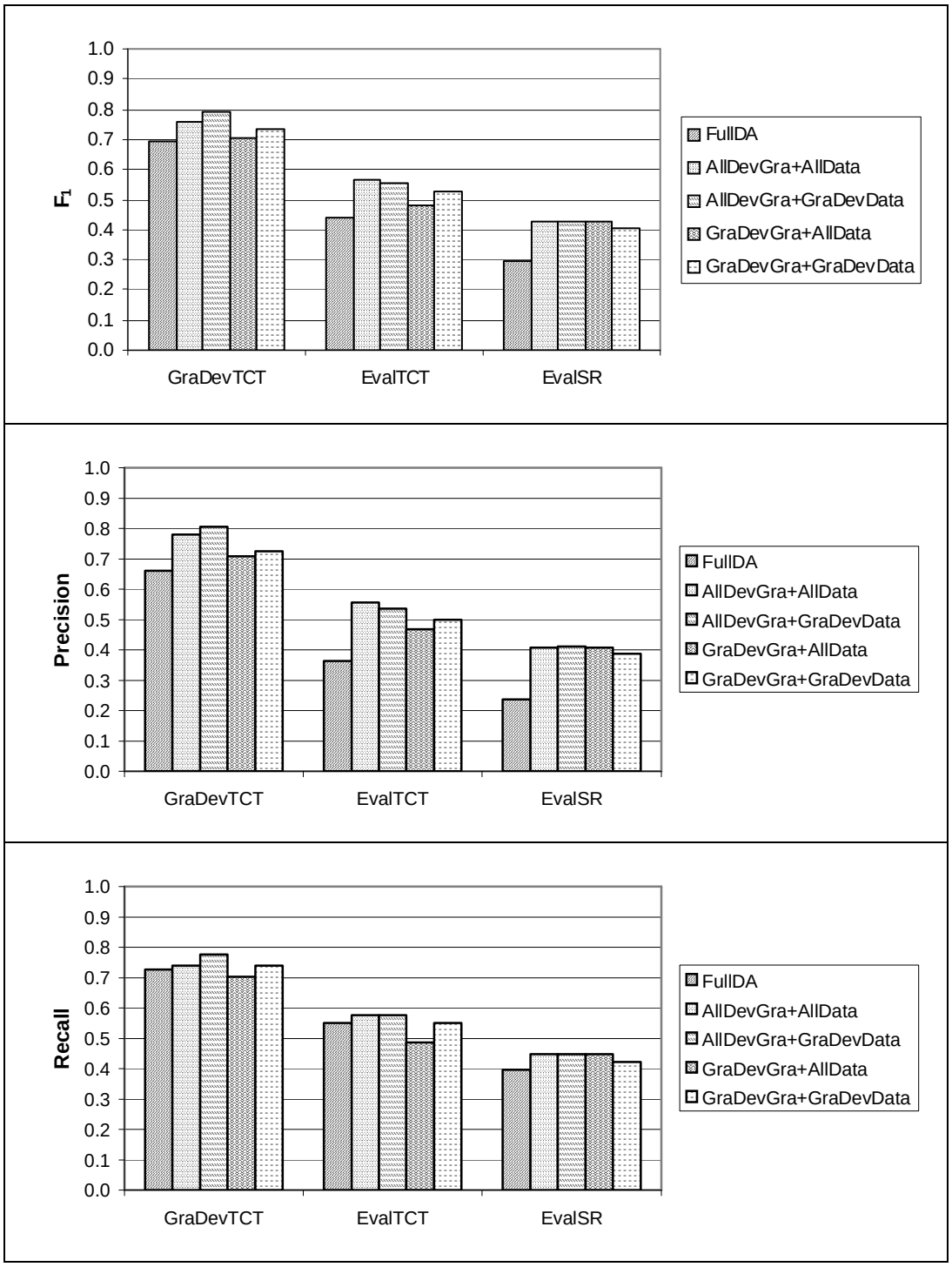

Figure 49: Concept sequence identification performance for analysis of full speaker turns in the portability experiment

Table 89 contains the results for concept sequence classification for the five analyzers tested in the portability evaluation based on full-turn input. The table shows the precision, recall, and F_1-measure for identification of concept sequences for each analyzer on each test set. Figure 49 illustrates the performance measures shown in the table for each analyzer and test set graphically. The concept sequence performance measures reflect the ability of the analyzers to identify complete concept sequences and were computed in the same manner as the domain action performance measures. Empty concept sequences were treated the same as any other complete concept sequence since a concept sequence containing no concepts can be a valid Interchange Format representation. The trends found in the concept sequence performance results were very similar to those for domain action identification, although the specific values of course differed and were generally somewhat higher. All of the hybrid analyzers outperformed the *FullDA* analyzer on F_1-measure and precision for all three test sets, and most of the hybrid analyzers also provided higher recall. The *FullDA* analyzer again suffered the largest absolute and relative decreases in F_1-measure moving from seen to unseen data and from transcriptions to automatic recognitions.

		Test Set		
Analyzer		**GraDevTCT**	**EvalTCT**	**EvalSR**
FullDA	**P**	0.736	0.393	0.219
	R	0.801	0.447	0.252
	F_1	0.767	0.418	0.234
AllDevGra + AllData	**P**	0.790	0.548	0.412
	R	0.734	0.495	0.408
	F_1	0.761	0.520	0.410
AllDevGra + GraDevData	**P**	0.785	0.535	0.405
	R	0.762	0.447	0.330
	F_1	0.773	0.487	0.364
GraDevGra + AllData	**P**	0.720	0.506	0.416
	R	0.656	0.437	0.408
	F_1	0.687	0.469	0.412
GraDevGra + GraDevData	**P**	0.742	0.534	0.455
	R	0.702	0.456	0.388
	F_1	0.721	0.492	0.419

Table 90: Individual concept identification performance for analysis of full speaker turns in the portability experiment

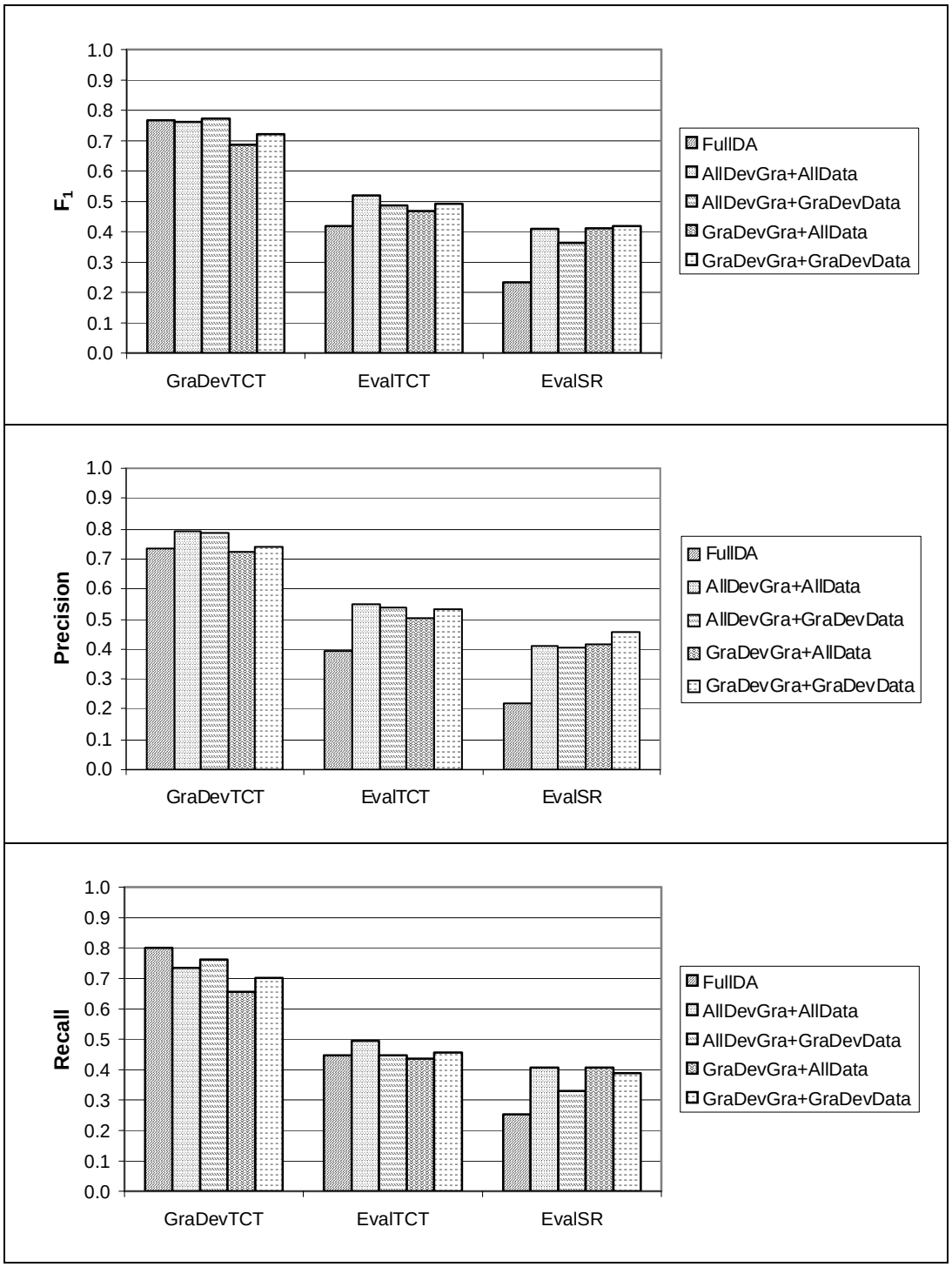

Figure 50: Individual concept identification performance for analysis of full speaker turns in the portability experiment

The precision, recall, and F_1-measure performance results for identification of individual concepts for the five analyzers tested in the portability evaluation are listed in Table 90 and illustrated in Figure 50. The results indicate how well the analyzers performed the task of identifying individual concepts present in the annotated data for the test sets. The full concept sequence assigned to a semantic dialogue unit depends heavily on the arguments present in the semantic dialogue unit. Thus, if the analyzers segmented a turn such that the arguments were not correctly grouped into semantic dialogue units, there would be a good chance that the resulting concept sequences would be incorrect. However, even if arguments were shifted to incorrect semantic dialogue units, the analyzer could still be able to find correct concepts to go along with the arguments.

In order to examine the performance of the analyzers on the identification of individual concepts, we computed the performance measures based on the whole turn ignoring semantic dialogue unit boundaries in a manner similar to that used for top-level arguments. Each concept sequence present in a turn was broken down into single concepts, and the set of concepts present in an analyzed turn was compared with the set of concepts in the annotated turn. Thus, if an annotated concept sequence were split across more than one semantic dialogue unit by an analyzer, the individual concepts would still be considered correct. Since empty concept sequences obviously contained no concepts, they were not a factor in the individual concept results.

The F_1-measure performance of the *FullDA* analyzer and the two *AllData* hybrid analyzers was similar on the *GraDevTCT* test set. The precision of the hybrid analyzers was higher than that of the *FullDA* analyzer, but recall was lower. For both *Eval* test sets, the F_1-measure performance of all four hybrid analyzers was superior to the performance of the *FullDA* analyzer. The F_1-measure of the *AllDevGra+AllData* hybrid analyzer was higher by about 10% absolute for the *EvalTCT* test set and by about 17.5% absolute for the *EvalSR* test set. Furthermore, the *FullDA* analyzer suffered much larger absolute and relative drops across all performance measures in moving from the *GraDevTCT* set to the *EvalTCT* set and from the *EvalTCT* set to the *EvalSR* set.

5.3.4.3 End-to-End Translation

In addition to measuring the performance of the individual analysis components, we also conducted an end-to-end translation evaluation for the Medical domain similar to the evaluation

described in Section 5.2.2 for the Travel domain. The end-to-end evaluation was conducted using the *GraDevTCT*, *EvalTCT*, and *EvalSR* test sets described previously. Because the NESPOLE! system was set up to support English, German, or French clients and Italian agents, the only meaningful end-to-end translation for our experiments would have been English-to-Italian. However, since there were no English-Italian bilingual speakers available for grading the outputs of our Medical system, we graded only English-to-English paraphrases. As the Travel domain end-to-end results for English input in Table 68 show, the acceptability of the English paraphrases and Italian translations was similar. Thus, the grades for English paraphrases should be indicative of the end-to-end translation performance for the Medical domain as well.

We tested the end-to-end paraphrase quality of the English NESPOLE! Medical domain system using three different analyzers. All other components of the NESPOLE! system were held constant across all of the testing conditions. First, we tested the quality of the translations produced using the purely grammar-based *FullDA* analyzer. Since grader time was limited, we only tested 2 of the 4 hybrid analyzer configurations. We tested translation quality using the *AllDevGra+AllData* analyzer, the most fully developed hybrid analyzer, and the *GraDevGra+GraDevData* analyzer, the hybrid analyzer with the least developed resources. Each of the analyzers was tested on all three test sets, for a total of 9 different output sets.

Grading was conducted using the process that was described in Section 5.2.2 for the NESPOLE! Travel end-to-end evaluation. The same 4-point scale (*Very Good*, *Good*, *Bad*, *Very Bad*) was used. Grades of *Very Good* and *Good* were again considered to be *Acceptable* for the purpose of our evaluation, and grades of *Bad* and *Very Bad* were considered *Unacceptable*. The graders were instructed to judge how well the meaning of each semantic dialogue unit from an utterance was preserved in the paraphrase produced by the translation system using the criteria for meaning preservation listed in Table 70. In addition to the 9 sets of paraphrases, the output of the automatic speech recognizer was also graded as in the Travel domain evaluation. Each of the 10 sets of paraphrases was graded by 3 graders. The graders were all staff members who had no direct affiliation with the NESPOLE! project. The 10 output sets were randomly shuffled for each grader so that the order of grading was not a factor, and the graders were not aware of what system configuration was used to produce each set.

	GraDevTCT	EvalTCT	EvalSR
FullDA	88.4%	59.2%	48.7%
AllDevGra+AllData	86.3%	54.0%	40.8%
GraDevGra+GraDevData	79.3%	50.0%	32.9%
Speech Recognition Hypotheses	--	--	69.7%

Table 91: Percentage of acceptable end-to-end paraphrases for English Medical domain input using majority vote grading

Table 91 shows the percentage of semantic dialogue units graded as *Acceptable* by at least 2 of 3 graders for each test set and analyzer. The results reported in the table were computed using the same majority vote among the graders that was used to produce the results shown in Table 73 and Table 74 for the end-to-end evaluation for the NESPOLE! Travel domain. The first three rows in the table show the grades for the *FullDA*, *AllDevGra+AllData*, and *GraDevGra+GraDevData* analyzers. The row labeled *Speech Recognition Hypotheses* shows the majority grades when the output of the speech recognizer was graded as a paraphrase. The columns show the grades for each test set.

	GraDevTCT	EvalTCT	EvalSR
FullDA	88.5%	59.2%	47.8%
AllDevGra+AllData	86.7%	58.8%	44.7%
GraDevGra+GraDevData	80.1%	51.3%	35.5%
Speech Recognition Hypotheses	--	--	71.5%

Table 92: Mean percentage of acceptable end-to-end paraphrases for English Medical domain input

An alternative to using a majority vote for combining grading results from different graders is to simply average the percentage of acceptable grades from each grader. Table 92 shows the percentage of acceptable translations as an average of the grades from the three graders rather than a majority vote. The general trends in the average grades are similar to those in the majority grades. The main difference between the two grade combination methods is that the grades for the hybrid analyzers on the *Eval* test sets were higher with the average method than with the majority method, particularly for the *AllDevGra+AllData* analyzer. The difference

is likely due to the fact that one of the graders graded the paraphrases produced using the *AllDevGra+AllData* analyzer for the *EvalTCT* and *EvalSR* sets higher than those produced by the *FullDA* analyzer, whereas the other two graders did the opposite.

As expected given the relative difficulties of the test sets, end-to-end performance for all three analyzers under both grading methods was best on the *GraDevTCT* data (seen, transcribed) followed by the *EvalTCT* data (unseen, transcribed) and finally the *EvalSR* data (unseen, automatically recognized). Under both grading methods, the performance of the *GraDevGra+GraDevData* analyzer was worse than both of the other two analyzers. This was also expected since the *GraDevGra+GraDevData* analyzer was trained on much less data and had much less grammar development time than either of the other analyzers. The most interesting comparisons are between the *FullDA* analyzer and the *AllDevGra+AllData* analyzer. The *AllDevGra+AllData* analyzer was the hybrid analyzer with the most grammar development time combined with the largest training set size. Under both grading methods, the *FullDA* analyzer had a higher percentage of acceptable paraphrases than the *AllDevGra+AllData* analyzer for each test set. Although the samples are very small, we use a t-test to test the differences between the average percentages of acceptable translations using the two analyzers for significance. We use a matched pair test since grades were assigned to outputs for each test set produced under two different conditions, and we use a two-tailed test because we had no expectation before testing of which analyzer would produce a better paraphrase. The significance tests show that the differences between the *FullDA* and *AllDevGra+AllData* analyzer were not significant for any of the test sets (*GraDevTCT*: t=1.07, p=0.398; *EvalTCT*: t=1.05, p=0.926; *EvalSR*: t=1.32, p=0.318).

In order to validate the portability results for both the analyzer components and end-to-end evaluations, we also examined the results when semantic dialogue units that were annotated with the *acknowledge* domain action were excluded from the results. The *acknowledge* domain action is generally easy to identify, and as Table 14 and Table 15 show, it is very frequent in some data sets. When *acknowledge* domain actions comprise a large portion of the data, their presence can sometimes obscure the results for the remainder of the domain actions. However, in the case of the data sets used in the experiments described in this section, *acknowledge* domain actions did not make up an overwhelming proportion of the data. Only about 3-4% of the semantic dialogue units in the data sets used for training the hybrid analyzers were tagged with the *acknowledge*

domain action. Similarly, in the evaluation data, only 4 of the 78 semantic dialogue units, about 5% of the data, were tagged with the *acknowledge* domain action. Furthermore, all 5 of the analyzers that we evaluated correctly identified all of the *acknowledge* domain actions. Thus, when we eliminated those semantic dialogue units from the evaluation results, the absolute levels of the various performance measures dropped a few percentage points, but the ranking of the various systems remained exactly the same.

Although the *AllDevGra+AllData* hybrid analyzer generally outperformed the *FullDA* analyzer based on the evaluation of the components of the Interchange Format representations produced, the end-to-end grading results appear to indicate that the paraphrases produced when the *FullDA* analyzer was used were somewhat better. Even though the differences between the analyzers based on the average grades were not significant, it was surprising that the *FullDA* analyzer would perform better than the hybrid analyzer given the component results. Using either grading method, the performance of the two analyzers was fairly close on the *GraDevTCT* data. Since the full domain action grammars were written specifically to cover that data set, it was not very surprising that the *FullDA* analyzer would produce somewhat better paraphrases even though the hybrid analyzer was also trained on the data. The results were more surprising for the *EvalTCT* and *EvalSR* test sets since the component results showed evidence that the hybrid analyzer was more robust than the *FullDA* analyzer on unseen data. Thus, we took a closer look at the output of the analyzers on the *EvalTCT* data set to see if we could identify reasons for the differences between the analyzers.

We focused our analysis of the end-to-end output on the semantic dialogue units for which the majority grades for *AllDevGra+AllData* hybrid analyzer and the *FullDA* analyzer were different. Of the 76 semantic dialogue units in the *EvalTCT* test set, 12 received different grades. For the output produced using the *AllDevGra+AllData* analyzer, 8 of those semantic dialogue units were graded *Unacceptable* and 4 were graded *Acceptable*. Thus, the net advantage for the *FullDA* analyzer was 4 extra *Acceptable* paraphrases. 3 of the 12 semantic dialogue units that received different grades in fact produced exactly the same paraphrase, indicating a small degree of inconsistency in the grading. Ignoring those semantic dialogue units leaves 9 semantic dialogue units with different grades, 6 graded as *Unacceptable* and 3 as *Acceptable* for the *AllDevGra+AllData*.

We noticed one additional inconsistency regarding 1 of the 6 semantic dialogue units that were graded *Unacceptable* for the *AllDevGra+AllData* analyzer. In one of the dialogues, the utterance "yes i am" occurred two times at different points in the dialogue. Although the output of each analyzer for the "i am" portion of the utterance was different, both of course produced the same output for each occurrence of the phrase. Because of a domain action misclassification, the output of the *AllDevGra+AllData* analyzer was "I give some information", which was graded each time as *Unacceptable*. The output of the *FullDA* analyzer was simply "I". However, one of the occurrences received an *Acceptable* grade and one received an *Unacceptable* grade, which accounted for one of the differences between the analyzers.

An important observation that we made in inspecting the remaining data was that the *FullDA* analyzer frequently parsed each top-level argument under a single parse tree rather than grouping the arguments into meaningful semantic dialogue units. Each argument was then assigned a trivial domain action, most often *give-information+concept* which is typically used for identifying fragments when there is insufficient information available in an utterance for identifying a more meaningful domain action. The result was a set of fragments that conveyed most of the detailed information present in the semantic dialogue unit but little of the information connecting the details. Because the grading system that we used focused on the preservation of meaning and not on fluency, translating a list of top-level arguments sometimes preserved enough meaning to receive an *Acceptable* grade. This behavior also explains the fact that the *FullDA* analyzer produced many more semantic dialogue units per turn on average than were contained in the annotated data, as noted in Table 88. The behavior also helps explain the severe degradation on concept identification observed for the *FullDA* analyzer, since many of the meaningful semantic dialogue units and concepts were replaced with fragments.

The *AllDevGra+AllData* hybrid analyzer also sometimes made use of the *+concept* concept but in a different way. Rather than splitting arguments into individual semantic dialogue units, the hybrid analyzer sometimes used *+concept* when it could not find a more meaningful domain action that licensed all of the top-level arguments in a semantic dialogue unit. In principle, the effect should have been similar to the effect created by the *FullDA* analyzer, although the English generator does attempt to order arguments based on the roles that they usually fill (i.e. subject, verb, etc.) when multiple arguments are present. However, for 1 of the 6 semantic dialogue units for which the *AllDevGra+AllData* analyzer received an *Unacceptable*

grade, the hybrid analyzer parsed exactly the same arguments that were parsed by the *FullDA* analyzer, but some of the arguments were not generated. This type of error is indicative of problems in the generator or generation grammars, a source of error over which the analyzer has no control. The following tables show the resulting Interchange Format representation and the end-to-end output for the semantic dialogue unit that received a different grade using the two analyzers.

Analyzer	**FullDA**
Input	but it also goes over like this

Arguments Parsed	`rhetorical=contrastive`
	`object-spec=pronoun`
	`focalizer=additive`
	`experience=e-move-7`
	`concept-spec=(modifier=like_this)`

Interchange Format	`c:give-information+concept`
	`(rhetorical=contrastive,`
	`object-spec=pronoun,`
	`focalizer=additive)`
	`c:give-information+experience`
	`(experience=e-move-7)`
	`c:give-information+concept`
	`(concept-spec=(modifier=like_this))`

Generation	But it also. It moves. Like this.

Table 93: End-to-end example from the *EvalTCT* test set using the *FullDA* analyzer

Analyzer	**AllDevGra+AllData**
Input	but it also goes over like this

Arguments Parsed	`rhetorical=contrastive`
	`object-spec=pronoun`
	`focalizer=additive`
	`experience=e-move-7`
	`concept-spec=(modifier=like_this)`

Interchange Format	`c:give-information+concept`
	`(rhetorical=contrastive,`
	`object-spec=pronoun,`
	`focalizer=additive,`
	`concept-spec=(modifier=like_this))`

Generation	But it also.

Table 94: End-to-end example from the *EvalTCT* test set using the *AllDevGra+AllData* analyzer

Table 93 and Table 94 show the end-to-end output produced by the *FullDA* and *AllDevGra+AllData* analyzers for the semantic dialogue unit that received a different grade. The *Input* row shows the original text of the semantic dialogue unit, and the *Output* row shows the generated paraphrase. The *Arguments Parsed* column shows the arguments contained in the parse produced by SOUP. As shown in the tables, both analyzers parsed exactly the same set of arguments for the semantic dialogue unit. The *Interchange Format* row shows the final representation produced by each analyzer. Although the Interchange Format representations produced by the analyzers did not match the annotated representation, it is clear from the output produced for the *FullDA* analysis that the arguments contained enough information to convey the original meaning. There were two differences between the output representations produced by the analyzers. First, the *experience=* argument was dropped during fallback processing by the *AllDevGra+AllData* analyzer. Since the dropped argument clearly contained important information, this shows a weakness of the fallback strategy. Second, the *AllDevGra+AllData* analyzer placed all of the arguments into a single semantic dialogue unit, whereas the *FullDA* analyzer split the arguments among 3 semantic dialogue units. Although the 4 remaining arguments were licensed by the domain action selected by the *AllDevGra+AllData* analyzer, there was no generation for the *concept-spec=* argument. Situations such as this are indicative of generation problems, or at least a difference in the behavior of the generator for the parses produced by the two analyzers. Although such issues are clearly important for translation performance, they are beyond the control of the analyzer.

Among the remaining semantic dialogue units with different grades, several were simply the result of better parses by one of the analyzers. However, we did observe three other points worth noting in the semantic dialogue units that were graded *Unacceptable* for the *AllDevGra+AllData* analyzer. First, one of the *Unacceptable* grades occurred when there was a repetition at the beginning of an utterance. The utterance in question was “it is it is pretty painful”. In the parse produced by the *AllDevGra+AllData* analyzer, a cross-domain grammar rule parsed “is it is” as *request-verification*. The effect was that “Is that right?” was inserted into the generated output, which apparently led the graders to give an *Unacceptable* grade due to the insertion of spurious meaning not in the original utterance. The *FullDA* analyzer did not suffer from this problem because it contained a domain action level rule that parsed “it is” with the following argument for “pretty painful”.

The second point worth noting had to do with resolving pronouns properly. The text for the semantic dialogue unit was “when it happens”. The *AllDevGra+AllData* analyzer parsed the pronoun “it” as *health-status=pronoun*, which was incorrect for that particular semantic dialogue unit and should have been *object=pronoun* according to the annotation. The incorrect argument led to the selection of an incorrect domain action. *give-information+experience+health-status* was identified as the domain action rather than *give-information+occurrence+object*, and the resulting output was “When there it is”. On the other hand, the *FullDA* grammar was able to parse the pronoun correctly because it had a rule that provided enough context to resolve the ambiguity.

One final observation that we made was that the *AllDevGra+AllData* analyzer inserted the *+negation* concept into the domain action for a semantic dialogue unit that clearly gave no indication for negation. The original text for the semantic dialogue unit was “i think i can do that”, but with *+negation* inserted in the domain action, the output was “I can not do that.” This error was caused by a combination of pronoun ambiguity and the fallback strategy. In the parse, “that” was placed under an incorrect argument (*object-spec=* instead of *general-action=*). As a result, the best domain action did not license the argument, and fallback processing was used. The concept sequence from the training data that licensed the most arguments happened to have had the *+negation* concept in it, although it was not necessary for licensing the arguments in this case. Since inserting a spurious “not” into a paraphrase clearly changed the meaning, an *Unacceptable* grade was assigned. This case represents a more serious problem that suggests that a more sophisticated solution during domain action verification and/or fallback processing might be useful. For the particular case of the *+negation* concept, it may be possible to define a small set of words (e.g. “not”) and/or arguments that must be present in a semantic dialogue unit in order for the *+negation* concept to be used in the domain action. Otherwise, the *+negation* concept could be dropped. It may also be possible to include a mechanism in fallback processing for checking if each concept licenses at least one argument. It is not immediately clear whether the overall effects of such a solution would be positive or negative, and any such solution would have to be tested experimentally.

5.3.5 Discussion

The results of the end-to-end evaluation for the NESPOLE! Medical Assistance domain using the hybrid analysis approach are not directly comparable to the results for the Travel & Tourism domain reported in Section 5.2.2 since the evaluations involved different domains, different data, and different graders. Nevertheless, we observe that the percentage of acceptable English-to-English paraphrases in the Travel domain was much higher than in the Medical domain. For the Travel domain, the percentage of acceptable paraphrases was about 68% for transcribed input and about 50% for automatically recognized input. For the Medical domain, the percentages of acceptable paraphrases for transcribed and recognized inputs were about 54-59% and 41-45%, respectively. This was not surprising since much more time was spent on the development of the Travel system than the Medical system. The domain for the Travel system was an expansion of domains that had been developed in previous systems, so there was a richer set of existing resources and experience for the Travel domain than for the Medical domain. Furthermore, more than two years were spent developing the translation servers for the Travel domain, whereas the Medical domain servers were developed in the final few months of the NESPOLE! project.

Although the end-to-end grading results for the NESPOLE! Medical domain seem to indicate that the purely grammar-based approach with full domain action grammars produced slightly better paraphrases, the results of all of the experiments together demonstrate the improved portability of our hybrid analysis approach. Several of the different grades between the two approaches for the *EvalTCT* test set appear to have been the result of inconsistent grading, which means that the difference between the approaches was not quite as large as it first appeared. Additionally, based on the average grading results, it appears that none of the differences between the *FullDA* analyzer and the *AllDevGra+AllData* analyzer were statistically significant. Furthermore, our evaluations showed that the hybrid analyzers generally did a better job of identifying components of the Interchange Format representation than the *FullDA* analyzer. This was true despite the fact that the *AllGraDev+AllData* analyzer was trained with a very small amount of data (about 1100 examples) that would still have been in the region of relatively rapid performance growth based on the results from the data ablation experiment described in Section 5.1. Some of the difference in end-to-end performance between the two analyzers may also be attributable to weaknesses in generation. Just as a relative small amount of time was spent developing the analysis grammars for the Medical domain, a correspondingly

small amount of time was spent developing the generation grammars. In the end-to-end translation, generation errors may certainly contribute to some unacceptable translations. We found at least one example in which a generation error resulted in an *Unacceptable* grade for a semantic dialogue unit for which the hybrid analyzer produced a reasonable representation. Additionally, as mentioned previously, there may be more than one reasonable interlingua representation for a given semantic dialogue unit. Because the hybrid approach may produce domain actions that were not seen in the data, it possible that it may produce valid interlingua representations that were not anticipated by the generation grammar developers. In such cases, the end-to-end translation performance using the hybrid approach may be somewhat lower than that of the full domain action grammar approach even if the identification of interlingua elements appears to be superior. Given that the hybrid analyzers required less development time than the analyzer that used full domain action grammars and achieved similar or superior performance, there is clear evidence for improved portability.

We also have evidence that the full domain action grammars would require more effort to maintain on an ongoing basis as new data was encountered. As shown in Figure 13 and discussed in Section 2.2, new domain actions continued to occur at a steady rate even with nearly 10 times as much data as was used in the portability experiments. The domain action grammar approach would require the development of new rules for each new domain action encountered. Based on the times reported in Table 78, developing rules at the domain action level took approximately 0.5 hours per domain action. On the other hand, the set of arguments remained relatively fixed after a few thousand examples. New arguments occurred much less frequently, and even after the complete set of data had been explored in Figure 14, the set of arguments was still smaller than the set defined in the Interchange Format specification. Once a relatively comprehensive argument grammar had been written to cover the Interchange Format specification, new rules for top-level arguments would only have to be added if the specification were updated. Minor phrasing and vocabulary additions would also sometimes be required. Of course, all of the maintenance required for the argument grammar would also be required for the domain action grammar approach.

In addition to grammar writing, annotation time is a factor in favor of the portability of the hybrid analysis approach. Whereas the hybrid approach only requires data to be segmented into semantic dialogue units and tagged with domain actions, the domain action grammar approach

requires annotation with full Interchange Format representations. Since tagging arguments comprised about 75% of the time required for annotating complete representations, annotating only with domain actions saves a considerable amount of time. Some time could be saved for the domain action grammar approach if data were annotated only with domain actions in a first pass. Then new or infrequent domain actions could be fully annotated and used for developing new domain action grammar rules. However, after the data was annotated with domain actions, the work would be done for the hybrid analysis approach. No further grammar development would be required, and the classifiers could be trained immediately. It might also be possible to use the classifiers from the hybrid analysis approach to perform an initial annotation of the data. Then human annotators would only be required to verify and sometimes correct domain action labels rather than annotating from scratch. Of course, it would have to be determined experimentally whether prelabeling the data in such a way actually reduced the human effort required for annotating the data.

		GraDevTCT	EvalTCT	Absolute Change	Relative Change
Domain Actions	**AllDevGra+AllData**	0.710	0.554	-0.156	-22.0%
	FullDA	0.654	0.408	-0.246	-37.6%
Speech Acts	**AllDevGra+AllData**	0.860	0.868	0.008	0.9%
	FullDA	0.816	0.735	-0.081	-10.0%
Concept Sequences	**AllDevGra+AllData**	0.760	0.566	-0.194	-25.5%
	FullDA	0.693	0.439	-0.254	-36.6%
Individual Concepts	**AllDevGra+AllData**	0.761	0.520	-0.241	-31.6%
	FullDA	0.767	0.418	-0.349	-45.5%
Top-Level Arguments	**AllDevGra+AllData**	0.769	0.550	-0.219	-28.4%
	FullDA	0.801	0.540	-0.261	-32.6%

Table 95: Changes in F_1-measure on identification of Interchange Format components on unseen versus seen input

In addition to the evidence for the domain portability of our hybrid analysis approach, the evaluations on the Grammar Development data set and the Evaluation data set provide evidence that the hybrid approach was more robust to unseen in-domain data than the full domain action grammar approach. We found evidence of the superior robustness of the hybrid analysis

approach to unseen data in the intended domain of coverage in the evaluation results for the identification of Interchange Format components. Table 95 lists the previously reported F_1-measures for the *AllDevGra+AllData* and *FullDA* analyzers on the *GraDevTCT* seen data set and the *EvalTCT* unseen data set. The table also shows the absolute and relative changes in moving from the seen data to the unseen data. The table shows that the *FullDA* analyzer suffered larger degradations in performance than the hybrid *AllDevGra+AllData* analyzer on all of the component identification tasks.

		EvalTCT	EvalSR	Absolute Change	Relative Change
Domain Actions	**AllDevGra+AllData**	0.554	0.402	-0.152	-27.3%
	FullDA	0.408	0.255	-0.153	-37.6%
Speech Acts	**AllDevGra+AllData**	0.868	0.854	-0.014	-1.6%
	FullDA	0.735	0.679	-0.056	-7.5%
Concept Sequences	**AllDevGra+AllData**	0.566	0.427	-0.139	-24.6%
	FullDA	0.439	0.297	-0.142	-32.4%
Individual Concepts	**AllDevGra+AllData**	0.520	0.410	-0.110	-21.3%
	FullDA	0.418	0.234	-0.184	-44.0%
Top-Level Arguments	**AllDevGra+AllData**	0.550	0.465	-0.085	-15.5%
	FullDA	0.540	0.419	-0.121	-22.4%

Table 96: Changes in F_1-measure on identification of Interchange Format components on automatically recognized versus manually transcribed input

We also observed similar evidence that the hybrid analysis approach was more robust to speech recognition errors in comparing performance on the transcribed *EvalTCT* test set and the automatically recognized *EvalSR* test set. Table 96 is similar to Table 95 but lists the changes in the F_1-measures for the *AllDevGra+AllData* and *FullDA* analyzers in moving from the *EvalTCT* test set to the *EvalSR* test set. Again the *FullDA* analyzer exhibited larger drops in performance on each task than the hybrid *AllDevGra+AllData* analyzer.

One additional consideration in comparing the hybrid analysis approach (*AllDevGra+AllData*) with the full domain action grammar approach (*FullDA*) was the system resources required for running the analyzers. All of the portability experiments for the Medical domain were run on a 2.8GHz Pentium4 PC with 512GB of memory running RedHat Linux 7.1.

For each test set used in the Medical domain portability evaluation, the translation server that used the *FullDA* analyzer required about 30MB of memory for the analyzer and approximately 0.6 seconds per speaker turn for analysis and generation. The translation server that used the *AllDevGra+AllData* analyzer required about 45MB of memory for the analyzer and approximately 0.7 seconds per speaker turn for analysis and generation. Since the hybrid analysis approach requires several classification steps and sometimes fallback processing, it was not surprising that it would require more processing time. However, the average time per speaker turn for analysis and generation was quite close to the full domain action grammar approach and was still under 1 second per utterance. The additional memory required by the hybrid analyzer was used for running the segmentation, speech act, and concept sequence classifiers. Although the *AllDevGra+AllData* analyzer required more system resources than the *FullDA* analyzer, it appears that there should be no problem using the hybrid analyzer in a real-time online translation system, which agrees with our experience in using the NESPOLE! translation system with the hybrid analyzer.

In conclusion, improving the portability and robustness of the analyzer used in our interlingua-based machine translation system was one of the primary motivations for developing our hybrid analysis approach. We found evidence in the experiments described in this chapter that our hybrid approach was both more portable and more robust than a strictly grammar-based approach that utilized domain-action-level rules. We first demonstrated that reasonable levels of domain action classification performance could be achieved with relatively small amounts of training data that could be annotated for training domain action classifiers in a few person weeks. We also showed that the hybrid analysis approach reduced the human effort required for data annotation as well as for grammar development and maintenance. We evaluated the performance of both analysis approaches in isolation and as components of end-to-end translation in the NESPOLE! system. Although the hybrid analysis approach resulted in slightly lower end-to-end performance, the performance of the individual analyzer components on identifying elements of the interlingua representation was generally superior to that of the domain action grammar approach. Thus, the hybrid approach provides at least comparable performance while reducing the human effort required for analyzer development relative to the domain action grammar approach. Together these results indicate the superior portability of the hybrid approach. Furthermore, our hybrid approach suffers smaller degradations in performance when unseen or

automatically recognized input is encountered, demonstrating the improved robustness compared to the strictly grammar-based approach.

Chapter 6 Conclusion

6.1 Summary

In this dissertation, we described an approach to natural language analysis for use in interlingua-based speech-to-speech machine translation. Our approach uses a combination of phrase-level parsing with handwritten semantic grammars and automatic domain action classification using machine learning techniques to transform spoken utterances into a shallow semantic task-oriented interlingua representation called Interchange Format. The analyzer operates in three stages, first using handwritten grammars to parse an utterance for semantic arguments and phrases. Since utterances are represented in the Interchange Format interlingua as sequences of meaningful segments called semantic dialogue units, our analyzer next identifies the semantic dialogue unit boundaries in an utterance. Finally, the analyzer assigns a domain action, which captures intent and focus, to each semantic dialogue unit in an utterance. Machine learning techniques are applied to the tasks of segmentation and domain action classification. Our analysis approach was developed primarily during the course of the NESPOLE! project, and an online version of the approach is fully incorporated in the NESPOLE! translation servers for English and German. In addition to the development of the analysis approach, we also conducted an experimental evaluation of our analysis approach using English and German data from the two NESPOLE! domains (Travel & Tourism and Medical Assistance).

We developed a memory-based (k-nearest-neighbor) segmentation classifier that provided strong performance on the task of identifying semantic dialogue unit boundaries. Our segmentation classifier made a binary decision about the presence or absence of a boundary at each potential boundary position using information about the words and interlingua arguments surrounding the potential boundary, the probabilities that those elements occurred around a boundary, and the length of the current semantic dialogue unit (in terms of words and argument parse trees). Our experiments showed that the classifier achieved high accuracy on transcribed utterances when the argument parser was not allowed to produce trees that spanned a true boundary. We also found that the performance of the segmentation classifier could be improved by including training examples created based on the partial information available at the

beginning and end of an utterance. Finally, we examined the performance of our semantic boundary detector in an online end-to-end evaluation using automatically recognized utterances and full speaker turns as input. Although performance dropped relative to the "clean" input experiments, the identification of true boundary positions was still quite strong, and our error analysis showed that many of the segmentation errors did not have a meaningful negative effect on translation quality.

We conducted extensive empirical evaluations of several aspects of automatic domain action classification and the related subtasks of speech act classification and concept sequence classification. Despite the difficulty of the classification tasks and data sparseness, the performance of our domain action classifiers, and especially our speech act classifiers, was quite good. Our best domain action classifiers achieved classification accuracies of 55-60%, and our best speech act classifiers achieved accuracies of 79-87%. We first compared the performance of several machine learning techniques (memory-based learning, decision trees, neural networks, and naïve Bayes classifiers) on the three tasks using features derived from argument parses. Our experiments showed that none of the learning approaches definitively outperformed the others across all tasks, domains, and languages. We also compared two different approaches to the domain action classification problem. In the first approach, a single classifier was used to identify the complete domain action for each semantic dialogue unit. In the second approach, separate classifiers were used to identify the speech act and concept sequence. We found that the dual classifier approach generally provided at least small improvements in domain action classification performance and increased flexibility with respect to the domain actions that could be produced by the analyzer. We also tested the effects of using different input feature sets on domain action classification performance. Our experiments demonstrated that information based on the words and argument parse for a semantic dialogue unit could be effectively combined to improve performance over either information source alone. Furthermore, we found that the word information played an important role in the identification of the speech act while argument information was particularly useful for classifying the concept sequence.

The combination of the raw output from argument parsing and domain action classification used in our analysis approach is not guaranteed to produce a domain action that licenses all of the arguments in an argument parse. Since valid interlingua representations must be produced for effective translation, we described and evaluated a fallback strategy that used the Interchange

Format specification to guarantee that the representations produced by our analyzer were legal. The results of our experiments clearly demonstrated the necessity for some form of fallback strategy for producing legal interlingua representations in our analysis approach. We found that the strategy we employed was very effective at finding alternative domain actions that licensed most of the parsed top-level arguments for each semantic dialogue unit. We also found that multiple alternative argument parses could be used during fallback processing to improve domain action classification accuracy relative to the use of only the single best argument parse.

One of the motivations for developing the analysis approach described in this dissertation was to improve the portability of the analyzer relative to a purely grammar-based approach that used only handwritten domain-action-level grammars. Our evaluations provided evidence that our analysis approach was both more portable and more robust than the domain action grammar approach. We found that annotation of data for training our classifiers could be performed quickly and that reasonable levels of classification performance could be achieved with relatively small amounts of training data. We also found that maintaining domain action grammars on an ongoing basis would require more effort than maintaining the argument grammars used in our approach. The set of arguments remains small and relatively fixed as new data is encountered whereas the set of domain actions continues to grow at a steady rate. The domain action grammar approach requires the development of new rules for each new domain action. Our analysis approach reduces the effort required for data annotation as well as grammar development and maintenance compared to the domain action grammar approach. We also evaluated the end-to-end performance of the NESPOLE! translation system for the Medical Assistance domain using both our approach and an approach using a domain action grammar. Although the translation quality using the domain action grammar approach appeared to be slightly better than when our approach was used, our analysis approach required less development time and generally outperformed the domain action grammar approach on identifying components of the interlingua representation. Our approach also generally suffered smaller performance degradations in the face of unseen and automatically recognized input.

6.2 Future Work

Our hybrid analyzer serves as the analysis module for English and German input in the NESPOLE! machine translation system. As demonstrated by the experiments described in this

dissertation, we have achieved our goals of improving the robustness and portability of the analyzer component relative to a purely grammar-based analysis approach. Furthermore, we have seen that our hybrid analyzer performs at a level that is generally sufficient to allow users of the NESPOLE! system to successfully complete dialogues in the intended domain of coverage. Nevertheless, there is still room for improving the performance of our analysis approach and correspondingly improving overall translation performance. The NESPOLE! translation system and Interchange Format database provide a useful platform for a continuing the investigation of our hybrid analysis approach. In this final section, we first describe several avenues of investigation that we would like to perform in the near future in order to develop a clearer understanding of the properties of our approach. We then discuss several directions for additional research that we feel might contribute to the further improvement of our approach.

6.2.1 Near Future

6.2.1.1 Assessment of Scalability

One issue which was not directly addressed in the work described in this dissertation was the scalability of our analysis approach in porting to larger domains. What should we expect in terms of necessary resources and development time if we need to port to a new domain that is *n* times larger than an existing domain? We have seen that our approach leads to improvements in portability and robustness compared to a fully grammar-based approach for translation in the limited domains of Travel & Tourism and Medical Assistance. We would expect that the approach using manually developed grammars would likely scale worse than linearly as the domain coverage grew due to increasing interactions among the domain actions and grammar rules. In addition to portability and robustness, we would like to explore how well our hybrid approach scales up as the domain of coverage expands. In an ideal situation, we would create a new larger domain and compare the development effort and data requirements for porting the analyzer to the new domain using both our approach and the domain action grammar approach. However, since such an effort would not be practical given the available resources, we would instead attempt to evaluate some indications of the scalability of our approach using existing resources.

The first step in this evaluation would be to define exactly how to measure the size of a domain. There are a variety of ways in which the size of a domain could be measured: number of

unique domain actions, number of arguments, size of the Interchange Format specification, vocabulary size, etc. For the purpose of our evaluation, we will define the size of a domain as the number of domain actions that the translation system is expected to cover in practice. As mentioned previously, even for the Travel & Tourism domain, the Interchange Format specification defines many thousands of domain actions that could be covered in principle. However, only a small subset of the possible legal domain actions is typically covered in practice. Thus, we will use the number of domain actions observed in the data for a domain to estimate the size of the domain. Although we have seen that new domain actions continue to appear as more data is collected, this should serve as a reasonable estimate of the practical size of a domain if the training data is representative of the domain.

We would like to conduct two additional experiments using the resources available for the existing NESPOLE! domains in order to investigate some aspects of the scalability of the hybrid analysis approach. First, we would conduct an experiment similar to the data ablation experiments described in Section 5.1 for the Travel & Tourism and Medical Assistance domains. We would create a new larger domain by combining the Travel and Medical domains into a single larger domain. We would again measure the changes in the performance of the domain action classifiers as size of the training data set was increased. By combining the two NESPOLE! domains, we would create a single domain that was larger than either of the individual domains in terms of the number of unique domain actions found in the data. By comparing the performance trends of the larger combined domain with those of the two smaller domains, we can get some idea of how the training data requirements change for the larger domain.

In the second experiment, we would conduct another variation of the data ablation experiment. In this variation, we would artificially create a "domain" of a certain size by randomly selecting semantic dialogue units from the data until a specified number of unique domain actions had been seen. For example, we might draw domain actions from the data until 500 different domain actions had been seen. We would assume that the domain actions seen in this random selection composed the entire set of domain actions defined in our artificial "domain". After eliminating the semantic dialogue units that were not tagged with one of these domain action from the data, we would conduct a data ablation style experiment to assess the performance of the hybrid analyzer for a domain of the specified size. We would repeat this process multiple times for each domain size and for a variety of artificial domain sizes. Based on

the annotation effort requirements for the Medical Assistance domain described in Section 5.3.2, we could estimate the annotation effort that would be required for each domain size and training set size. A comparison of the performance trends and annotation effort requirements across domain and training set sizes should provide useful insight into the scalability properties of our analysis approach.

6.2.1.2 Assessment of Sensitivity to Argument Parse Quality

Another aspect of the performance of our hybrid analysis approach that was not directly addressed in this dissertation was the sensitivity of the segmentation and domain action classifiers to the coverage of the argument grammars and the accuracy of the argument parses produced. We believe that it would be insightful to conduct an additional experiment to examine how the performance of the classifiers degrades as the quality of the argument parses decreases due to poor grammar coverage. Analyzing the sensitivity of the domain action classifiers to the quality of the argument parse would provide additional information about the robustness of our approach as it will show how well the classifiers overcome errors in the argument parse to determine the correct domain action. In order to address this question, we need to compare the performance of several hybrid analyzers that are trained on the same training data but that use argument grammars with varying amounts of development effort.

We can make use of the argument grammars developed for the Medical Assistance domain portability experiments in order to perform this comparison. As described in Chapter 5, we already evaluated the performance of the hybrid analysis approach using argument grammars at two different stages of development. The *GraDevGra* grammars were developed based on only four dialogues from the Medical Assistance domain, and the *AllDevGra* grammars were developed based on all available data as well as the Interchange Format specification. Thus, we already have performance measures available for two grammar development points. Without any additional grammar development, we could get performance results for a third development point by using the final versions of the Travel & Tourism domain argument grammars that were used as a starting point for the Medical Assistance domain grammars. Using these grammars, we can produce performance results for grammars with no explicit development on the Medical Assistance domain. Considering this third performance point along with the two points already available will allow us to observe the behavior of our hybrid analysis approach as the quality of the argument parse degrades. In addition, the evaluation with the Travel & Tourism domain

grammars will allow us to evaluate the coverage of the argument grammars without any explicit development on the Medical Assistance domain.

6.2.1.3 Comparison of "Clean" versus "Real" Argument Parses

All of the experiments reported in this dissertation made use of "real" argument parses for training the domain action classifiers. In other words, the arguments used to train the classifiers were produced by parsing the training examples using the same grammars that would be used in the online hybrid analyzer. Of course, there was no guarantee that the argument parse produced for an example actually contained the true arguments present in the manual annotation of the example. On one hand, this may have been advantageous because the domain action classifiers were trained under the same conditions that would be present when testing on unseen input. On the other hand, errors introduced by the argument parser may have actually hurt classification performance by introducing too much noise into the training data. Thus, we feel that it would be informative to compare the performance of domain action classifiers trained using "real" argument parses with that of classifiers trained on "clean" argument parses (i.e. on the arguments from the manual annotation).

In order to evaluate the effects of training on "real" argument parses compared to training on "clean" parses, we can make use of the true arguments in the annotated training data. We will compare the performance of domain action classifiers trained under two conditions. In the first condition, the domain action classifiers will be trained on "real" argument parses as in the experiments reported in this dissertation. Under the second condition, the classifiers will be trained using the "clean" argument parses available in the annotated data. Because the annotated argument parses are in the Interchange Format rather than the output format produced by the argument parser, the classifiers will use the top-level Interchange Format arguments feature set (described in Section 2.5.2 and referred to as *IFArgs* in Table 53) to represent the arguments present in a semantic dialogue unit in both conditions. In order to make the classifiers most similar to those used in the portability experiments described in Chapter 5, we will train domain action classifiers using TiMBL with the pseudo-argument grammar labels (*PseudoGra*) and the top-level Interchange Format arguments (*IFArgs*) as input features. The concept sequence classifier will also use the speech act assigned to the semantic dialogue unit as a feature. For the classifiers trained on "clean" parses, the top-level Interchange Format argument produced by the argument parser will be replaced with the true arguments from the annotation. The argument

features for the test data will be produced using the top-level arguments extracted from the argument parser output since those are the only arguments that would be available in a true test situation. We will compare the speech act, concept sequence, and domain action classification performance of the classifiers trained under each condition using a 20-fold cross-validation setup as in the previously described experiments. It will not be possible to make a similar comparison for the segmentation classifiers. There is no "clean" data available for training the segmentation classifiers because the annotated arguments are not associated with specific spans of text, and the arguments are not guaranteed to appear in the same order in the annotation that they appear in the text. Since the segmentation classifier depends on the order of the arguments in the text, the annotated arguments would be insufficient for training.

6.2.2 Longer-Term Research Directions

The experiments described in the previous section are intended to provide us with additional information regarding several properties of our hybrid analysis approach. Those experiments can be conducted using available resources to develop a fuller understanding of our approach. In this section, we mention several additional lines of research that believe could lead to further improvements of one or more aspects of our approach.

6.2.2.1 Improvement and Automation of Argument Parsing

In the analysis approach described in this dissertation, we used handwritten grammars for the purpose of argument parsing and focused our attention on automating the identification of domain actions. Although our approach improved portability by reducing the effort required for data annotation and grammar development, grammar writing still consumed a large portion of the development time for the analyzer, as shown in Table 81. The reliance of our analyzer on handwritten grammars also means that expert human grammar writers must be involved in any efforts to expand or adapt the analyzer. Since grammar writing still represents one of the bottlenecks of developing an analyzer for a new domain, we would like to explore ways of automating argument parsing and/or argument grammar development in order to further improve the portability of the analyzer and reduce the requirements for human effort and expertise.

One seemingly promising possibility for automating grammar development would be to extract grammar rules from the interlingua specification. A complete extraction of all possible rules allowed by the specification would likely result in a grammar that was too large and too

ambiguous to be practical for argument parsing. Thus, the extraction of grammar rules from the specification would have to be restricted in some way, or the extracted grammar would have to be pruned in order to provide a practical grammar. Such restriction or pruning might be performed based on the arguments observed in annotated data. This could serve to eliminate most of the human effort required for grammar writing but would also certainly increase the effort required for annotation. Alternatively, a grammar writer could manually prune the grammar extracted from the specification rather than developing grammar rules from scratch.

As an alternative to automating grammar development, it might be possible to treat the argument parsing task as a form of chunk parsing. In this case, chunk parsers could be trained to identify Interchange Format arguments and values as well as useful phrases that are currently identified using handwritten grammars. Of course, training a chunk parser would minimally require the annotation of data with complete Interchange Format representations. Unlike in the current NESPOLE! database, the annotation of arguments would also have to include an indication of the words spanned by each argument, subargument, or value. Useful phrases would either have to be identified and marked during annotation or automatically extracted from the data. How the data and annotation requirements for training an effective chunk parser would compare with the data and annotation requirements for writing an effective argument grammar would have to be determined empirically.

6.2.2.2 Automate Data Annotation

Classification of domain actions rather than writing domain action grammar rules eliminates a substantial portion of the grammar-writing effort, but training the classifiers used in our approach still requires a corpus of utterances that have been divided into semantic dialogue units and annotated with domain actions. Although data annotation represents a substantially smaller portion of the human effort required for developing our hybrid analyzer, we still believe that it would be useful to examine the possibility of automating the annotation process. We saw that a relatively small amount of data was necessary in order to achieve the level of performance observed in our experiments. Automating annotation might allow us to further reduce the effort required to produce this data. Furthermore, machine learning approaches generally benefit from the availability of more data, and automating annotation might allow us to produce much more annotated data in the same amount of time required for manual annotation of a smaller data set.

One possible way to at least partially automate the annotation process would be to utilize the segmentation and/or domain action classifiers to perform a first-pass annotation of new data. Human annotators would then only be required to correct the annotations rather than producing them from scratch. Of course, it would also be possible to simply train new classifiers including the automatically annotated data without human verification, but there is certainly no guarantee that such an approach would improve performance. A compromise might be to accept automatic annotations that exceeded some confidence threshold. The attention and effort of human annotators could then be directed to those semantic dialogue units on which the existing classifiers exhibited low confidence. It is likely that the domain action classifiers would have low confidence mainly for those training examples that were relatively rare or previously unseen. This would serve to focus the attention of the human annotators on those utterances in the data which truly required human input and should allow for more efficient use of limited human resources. It would have to be determined empirically whether or not automating annotation in any of these ways would reduce the human annotation time required per semantic dialogue unit. Making the annotation process more efficient would also be especially important if argument parsing and or grammar development were automated since the annotation requirements would increase.

6.2.2.3 Alternative Fallback Processing Strategies

Production of legal Interchange Format representations is absolutely critical to the success of translation using the NESPOLE! system. When the tasks of argument parsing, speech act identification, and concept sequence identification are separated as they are in our analysis approach, some method for checking the validity of the interlingua representations produced is therefore necessary. We demonstrated that utilizing the Interchange Format specification during domain action classification actually improves the quality of the interlingua representations produced by the analyzer. Although testing the validity of the representations produced by the analyzer is a simple task, the strategy for correcting illegal representations is an area where further investigation may provide additional improvements in analyzer performance.

The fallback processing strategy used in our analyzer is only one possibility for extracting a legal interlingua representation from the outputs of the analyzer components. Our current fallback strategy looks for a domain action that licenses the largest set of parsed arguments and drops any arguments that are not licensed by the domain action. However, the results from the

evaluation of the fallback strategy described in Section 4.6 indicate that the correct domain action for a semantic dialogue unit was sometimes returned by the classifiers but changed during fallback in order to accommodate errors in the argument parses. Furthermore, the results from the portability study described in 5.3.4 suggest that even producing fragmentary analyses containing individual arguments is sometimes sufficient for producing acceptable translations.

Based on these observations, we believe that an exploration of alternative fallback strategies would be useful. Even if an argument were parsed incorrectly, it could still contain important information from the original utterance. One example of a simple alternative to the fallback strategy employed in our analyzer would be to trust the best (legal) domain action produced by the classifiers and remove any unlicensed arguments from the semantic dialogue unit. Rather than dropping arguments however, unlicensed arguments could be placed in a separate semantic dialogue unit with either a default domain action or a newly classified domain action. Another possibility might be to split the semantic dialogue unit around the unlicensed argument, creating multiple semantic dialogue units containing fewer arguments. The domain action classifiers could then be used to assign domain actions to the new semantic dialogue units. The relative effectiveness of any alternative fallback strategies would have to be evaluated empirically. An empirical investigation of the possibility of using the specification to identify mandatory segmentation points in the argument parse might also be useful.

6.2.2.4 Additional Sources of Classifier Features

Finally, the classifiers in our analysis approach only use features that can be extracted directly or automatically from the text of a speaker turn or semantic dialogue unit. This was partly due to the design of the NESPOLE! system architecture in which the only input the analyzer receives from the rest of the system is the text (either the best recognizer output or typed input) of the current speaker turn. However, additional features extracted from sources other than the text of a semantic dialogue unit may prove to be useful for improving the performance of the classifiers. For example, some of the systems described in the Section 1.6 used acoustic properties of the current turn to assist in the identification of segment boundaries and speech acts. The addition of prosodic features from the speech recognizer could be very useful for the identification of semantic dialogue unit boundaries and for the identification of some domain actions. In addition to acoustic features, contextual features describing the domain actions that occurred in previous turns from both sides of the dialogue may also be useful for improving domain action

classification. For example, knowing that the domain action just produced by the other speaker was included the *request-information* speech act might establish a higher expectation that the speaker would produce a *give-information* speech act. These are only a few of the most obvious sources from which additional features may be drawn, and they are sources which could in principle be available in systems other than NESPOLE!. We would like to investigate the possibility of using these or other untapped sources of features to further improve the performance of out hybrid analysis approach.

Appendix A Travel & Tourism Domain Evaluation Sets

A.1 Transcribed English Evaluation Utterances

hello
yes can you hear me
okay i'm interested in a your summer packages preferably for campsites for
good morning
okay
okay
i went on the website a little bit i don't know too much about it but we were
looking to go camping preferably like near a lake or something somewhere
where they have activities for children do you have an
okay and do they have we were also interested in taking maybe surfing lessons
maybe mountain bikes visiting castles just to keep the children busy
okay
okay that is great and do you know of any you would you know how to get the
best way to reach buonconsiglio castle we were interested in maybe visiting
that castle
okay
okay
okay
yes yes
okay
yes uh huh
okay and is that easily acceptable by a camper van like an rv or do you would
yes to for traveling and stuff is that possible
okay that is great
i think we just wanted to know exactly how to reach it and like maybe get in
contact for tours or something like that
mhm
okay that is great
yes i have it right here
thank you
mhm
okay
okay
okay
okay
all right that sounds good
yes oh and i also want
i also wanted to ask about specific activities for children like in this
campsite area or any or in the whereabouts what kind of activities there are
for specifically for children
mhm
oh okay that sounds good
okay and these packages are available all summer long or are there specific
dates
okay 'cause we were looking to
yes yes i can hear you
okay all right
yes i can hear yes
yes i
i think july like around july first
around jul
okay that sounds good oh and also at the campsite what is in

yes yes it will thank you
just a second yes i have the map
yes
okay i see it
yes
yes that is very good
okay oh and in this campsite what is included with your with your fee like are there any meals or bathroom
okay
okay okay so it is all one big thing all right let me think
okay
okay and that is another that is not another campsite that is just included
okay okay
mhm
no i think that is pretty good i think i covered everything thank you very much for your time and
okay have a good day thank you
okay bye bye
hello can you hear me
excellent can you hear me okay right now
yes i'm looking for a package this summer i'm wanting to travel in trentino
oh i wanna be right next to a lake i would like to be next t
a lake would be great a lake would be great
wow
that sounds great yeah
first week in A the first week the first week in august
uh huh
okay
great

okay there it is i got it
oh that is lovely

well i'm traveling by camper van so i'm gonna need some sort of campsite in the area
okay tha
okay i it will
okay good 'cause it is it is gonna be me and another adult and two children so
perfect
'kay could you tell me a little bit about the campsite
oh great
uh huh
oh good
okay
yup
okay yup i see it
i see it great
is there any way to get surfing lessons by the lake as well
hello yes i see it i see it
yes
uh huh
yup i see it
okay
okay i see that are they any are there any loc
are there any local castles in the area
yes i see that

yeah
yes i see it
yeah
okay
okay thank you i think i have all the information i need to plan my trip now i'll gonna have to think about some of this but you've been most helpful
one mor one more thing would be helpful just one more could you tell me how to reach my campsite when i'm in my driving my camper how to get there
yes
she has got one more thing
yes i see it i see it
i see it
yes i see that yup
okay
okay o excellent well thank you very much
all right bye

A.2 Automatically Recognized English Evaluation Utterances

okay
can you hear me
okay if any any usually package in Cavalese or can I well
good morning
okay
okay
I on the the web page a little bit hello to much about that but we were looking to go camping confirming like nearly something had where they have activities for children do you have any it
okay and do they have we're also interested in taking maybe surfing lesson in ah me mountain bike visiting castle of picky the children to the
okay
okay that great and and do you have any you would you know how to get the best way for to reach Buonconsiglio castle we're interested in maybe visiting that castle
okay
okay
okay
yes
okay
yes
okay that if in easily a couple by and I camper van the lake and are the the park you would have
I yes to for traveling in is that possible
okay that is great and
and just one that you know exactly how to reach at and like maybe get and contact for tourist on or something like that
no
okay that great if okay
yes I have that weekend
thank you
and
okay
okay
okay
okay
I can
yes the if and helpful and

it to ask about the to take activities for children like in this camping am or any or anywhere about all kind of activity you there are for specifically for children
no
okay that sounds good
okay on in these packages there available summer long or a there specific a
okay can you in looking okay
yes yeah I can hear you
okay great
yes I can hear you
yes I think
I think July in the lake around July first
around it
okay that sounds good I don't know what the camping what is the
yes yeah that will be thank you
and I can yes I have the map
yes
okay I see it
yes
yes that very good
okay I when is the campsite what is included with here what here C-E-O like are there any me over the that too
okay
I okay okay hotel one they can are they at and and
okay
okay and that another that nine other camping that just included
okay okay

and I think that pretty good think haven't everything so thank you very much for your time in
okay have a good thank thank you
okay bye bye
I will can you hear me
actually I'm can you hear me okay right now
I guess I'm looking for a package either some I'm on to travel in Trentino
all right one of the right next to a lake in the
the lake would be great lake introductory
what
that sounds great yes
I'm the first week about the first week in the first week in August and
uh-huh
okay
great

okay that is like got it
and locally

well I'm traveling by camper van some anytime sort of campsites in the area
okay that
okay it will
okay good is it gonna send me me another dealt and two children so
perfect
okay could you tell me a little bit about the cancelled
how great
uh-huh
okay
okay

yes
okay yes by here
the
is there any way to get surfing lessons other lake as well
yes I did here
yes
uh-huh
yes I see it
okay
okay you that are there any festivals
available castles in the area
yes I see it
yes
yes I can hear you
yes
okay
okay I don't here have what are the information I need to Trento like trip
and I'm okay good that can that but you can most helpful
I really want that would be helpful just one more could you tell me how do
you reach my camp like what in the Trento by camper how to get there
yes

yes I did I see it
I here
yes I see that yes
okay
okay actually we're thank you very much
bye bye

A.3 Transcribed German Evaluation Utterances

hallo
hal
ja 's geht
ja
geht
okay
hallo und zwar geht's darum ich wuerd' gern einen Winterurlaub plan hallo
verstehen Sie mich nicht
doch
okay
ich wuerd' gern einen Winterurlaub planen fuer zwei Personen in Val-di-Fiemme
mhm
mehr Apartment
mhm
mhm
mhm
mhm
Skipass mhm
mhm
mhm
mhm
nein keine Kinder
mit Rodelschlitten und Schlittschuh laufen kann man da auch oder auch
ja Schlittschuh laufen kann man da auch
mhm
mhm

aha
ja die seh' ich
mhm
ah ja
mhm
mhm
ja
mhm
ach das da oben mhm
mhm nee aber das sieht ja ganz gut aus
in dem oberen Ort in dem Predazzo oder wa was gibt's da alles
mhm ja
mhm
aha
okay
mhm
mhm
ja
ja
fuenf null
zweimal die fuenf alles klar hab' ich
mhm
ich seh' sie mhm
ja
Zwei-Bett mhm
Doppelbett oder sind es zwei Einzelbetten
ist es ein Doppelbett oder sind es zwei Einzelbetten
beide Moeglichkeiten
okay
mhm
aha
ah ja
mhm
mit dem Auto ja
mhm
mhm
ah ja
oh koennen Sie mir dann spaeter auch sagen wie ich dahin komme mit dem Auto
ja
mhm
mhm
ich seh' die Karte ja
mhm
mhm
ah ja mhm
ich seh' es
okay
mhm
ah ja gut
okay
ja gibt's da irgendwelche Museen oder irgendwelche Veranstaltungen die da
dann sind im Winter oder
mhm
mhm
mhm
ah ja
mhm

mhm
mhm
die Seite seh' ich ja
mhm
ah ja
mhm
und koennen Sie mir noch mehr fuer das zu dem Ap Apartment sagen das Sie mir vorhin gezeigt haben das Zwei-Bett-Apartment
nicht
mhm
mhm
okay
ja
ja
ja
ja
mhm
ah ja
mhm
und koennten Sie mir noch 'n Hotel sagen fuer den Fall dass das Apartment fuer mich nicht in Frage kommt
ja mhm
mhm
mhm
mhm
ja
ja
ja
nee es geht
mhm
drei Sterne
mhm
ja
mhm
ja
ja
zwei drei
okay und also ich hab' vor da vom ersten ersten zweitausend drei bis zum achten ersten hinzufahren sind da irgendwelche Veranstaltungen schon geplant fuer den Zeitraum oder
mhm
ja
kommt nichts
mhm
mhm
mhm
ja
ja
alles klar dann bedank' ich mich bei Ihnen
okay danke tschuess
ja hallo
ja ich hoer' Sie wunderbar
ja ich moecht' einen Sommerurlaub buchen fuer zwei Personen
ja
noch nicht richtig nein
mhm
mhm ja also wichtig ist mir dass ich Mountainbike fahren kann

hallo
hallo
hallo koennen Sie mich hoeren
hallo
hallo
ja ich hoer' sie nicht mehr und so
also ich hoer' mich selber und ich hoer's rauschen
jetzt ist kein Stereo mehr jetzt hast am Stecker was verdreht
ja hallo jetzt hoer' ich Sie wieder
ja ich hoer' Sie
sind wir wieder da
gut
ja mhm
mhm ja
so mhm ja jetzt seh' ich die Karte schoen
ja seh' ich mhm
mhm also wichtig ist mir dass ich Mountainbike fahren kann dann sind Berge sind nicht schlecht
mhm
mhm
mhm
mhm
also ich komme mit einem Wohnmobil und moechte dann auf einen Campingplatz
mhm
ja
mhm
ja mhm
ja Sommer mhm
ja ich habe ges gesagt zwei Personen
mhm
und ich wae
also was interessant waer' waer' ein vielleicht ein all All-Inclusive-Paket oder also fuer eine Woche dann
mhm
mhm
mhm
mhm
ja das waer' sehr schoen ja danke
ja jetzt hab' ich den Plan
mhm
mhm
wo ist dann der Camping
im Moment noch nichts
ah jetzt okay ja da unten
rechts unten ja ich seh's mhm
mhm ist das Naturpark oder
mhm
mhm
mhm
mhm
mhm
mhm
mhm sehr schoen mhm
dann was mich noch interessiert
ja ich das ist schoen mit dem Naturpark aber ich waer' jetzt wenn es vielleicht schlechtes Wetter hat oder so gibt es auch irgendwelche Museen oder Sachen die man sich anschauen koennte

ja das waer' sehr schoen
mhm
mhm
ich seh' die Seite jetzt ja
mhm ja mit den Anreiseinformationen
mhm
mhm
mhm
mhm
mhm
ja das waer' interessant
mhm ja das Castello-di-Arco
Di-Arco ja
mhm
mhm
mhm
mhm
mhm
ja mhm gut mhm das was auch noch interessant waer' wenn wir gerade bei Schloessern sind ist das Schloss Buonconsiglio oder wie das gesprochen wird in Trient
kann ich das von meinem Urlaubsort leicht erreichen oder
mhm
im Whiteboard ja mhm
mhm
ja
mhm
mhm
mhm
mhm
mhm
mhm okay montags zu mhm
dann ja dann
dann wuerd' ich gern noch mal mit dem Zelt was
ich haett' noch gern Informationen ueber den Zeltplatz
ja genau also
ja oder s koennen Sie m mir nicht ueber den t noch was ueber den t Auskunft geben wie teuer oder so das ist oder soll ich dort selber anrufen
mhm eine Woche dann
mhm
mhm
mhm a
mhm
mhm
ach so ja dann ruf' ich dort an wenn Sie mir noch die Telefonnummer geben
mhm
mhm
mhm
gut
ja
ja das waer' auch gut
Riva-Aura
mhm
mhm
ich wiederhol' gerade die Telefonnummer
drei neun vier sechs vier fuenf sechs acht acht neun neun
gut okay dann vielen Dank fuer die Auskunft

und auf Wiederhoeren
ja zwei auf einen Besuch

A.4 Automatically Recognized German Evaluation Utterances

hallo

ja es geht
ja ja
ja
okay
hallo und zwar geht es leider nicht bis Veneto oder klar hallo
welchen Termin
ja
okay
ich w~urde gerne mit Sauna Plan festhalten so in Val-di-Fiemme
ja
mit mir Apartment

ja
hm
ja
schlittschuhlaufen
hm
ja
aha
nein keine Kinder
ja ist okay und schlittschuhlaufen kann man da auf oder auch
ja schlittschuhlaufen kann man auch
hm
ja
aha
ja leserlich
ja
ah ja
hm
aha
ja
ja
ach Restaurant bin
aha ja das w~are ganz gut aus
dann in Veneto Varena angenehmen vielleicht davor oder mal was gibt es da
alles
hm ja ja
hm
aha
okay
hm
genau
ja
ja
f~unften oder
zweimal wieviel alles klar habe ich
hm
ich sehe k~onnen
ja
okay dann

Doppelbett da das m~u~ste vereinzelt werden
ist das ein Doppelbett oder vielmehr zwei Einzelzimmer nehmen
beide M~oglichkeiten
okay
hm
aha
ah ja
ja
mit dem Auto ja
ja
ja
ah ja
oh dann k~onnten wir dann sp~ater auch sagen ich da hinkommen mit dem Auto
ja

hm
ich sehe grade ja
hm
aha
ah ja
ich sehe
okay
ja
ah ja gut
okay
ja gibt es da irgendwelche Museen oder wie wir Veranstaltungen in Via-
Segantini nehmen da oder
hm

ja
ah ja
ja
in
ja
geht es eigentlich ja
hm
ah ja
dann
und k~onnten Sie mir noch ein erstes f~ur den Abend der Apartment machen ich
mir eingezeichnet zweite dankbar Ball
nicht
hm
ja
okay
ja
ja
ja
ja
ja
ah ja
ja
und wir k~onnten Sie mir einen noch ein Hotel sagen sinnvoller f~ur das
Apartment f~ur mich nicht in Frage kommen
ja k~onnen
hm

hm

ja
ja
ja
ja es geht
ja
alles klar
ja
ja
ja
ja
ja
nein planen
okay und also ich habe vor da vom ersten ersten zweitausend frei bis zum achten ersten zu fahren w~are finde ewige Veranstaltungen Stundenplan f~ur den Zeitraum oder

ja
gut nix
ja
ja
aha
ja
ja
ja dann bedanke ich mich bei Ihnen
okay danke tsch~u~s
ja hallo
ja ich h~ore Sie wunderbar
ja ich m~ochte einen Sommerurlaub buchen f~ur zwei Personen
ja
haben Sie sich nein
hm
ja also wichtiges mir da~s ich Mountainbike fahren kann
hallo
hallo
hallo da bin ich wieder
hallo
hallo
ja ich habe Sie nicht mehr und so
ab M~unchen di-Fiemme und ich w~urde vorschlagen
da h~atte ich nur jetzt ist Castello-Molina Fassung schlecht aus April
ja hallo bef~urchte wir
ja ich habe den
ja wunderbar
gut
ja
ja
ach ja ich sehe grade sch~on
das sehe ich
also wichtig ist mir was ich morgen wollte fahren fahren dann sind wir bestimmt nicht schlecht

nein
ja
also ich komme mit einem wohl okay und m~ochte und auch ein gern fliegen dort
hm
ja

ja dann
ja Sommerabend
ja ich glaube wir haben gesagt zwei Personen

und ich werde
also was interessant w~are w~are ein vielleicht eineinhalb unbedingt sowieso okay
oder also f~ur eine Woche Turn

ja das w~are sehr sch~on ja danke
ja haben Sie fahren

Montag der Termin
im Moment noch nicht
ah ja okay also unten
recht und ja surfen
das Naturpark oder

ja

ja sch~on
dann aber vielleicht noch interessiert
ja ich bin das ist sch~on in Hamburg aber ich w~are es wenn wir vielleicht schlecht da~s wir da hat oder so gibt es auch irgendwelche Museen oder verhandeln und anschauen k~onnte
ja das w~are sehr sch~on

ich finde soll wird ja
ja mit anreisen Promotionen

ja
ja

ja sollen wir fahren
ja das habe ich notiert ausgucken
ja Frau ja

genau gut das was auch immer von daher Mitarbeiter Schl~osser und Seen ist das skil~auft und kann viele oder das gesprochen wird interessieren
also von meinem Wohnort aus weil ich da reichen oder

wir wollten uns ja
Wiedersehen

ja

ja
okay Montag so um
dann ja dann
dann w~urde ich gern noch mal den fest was
ich h~atte noch kl~aren Information ~uber den Zeltplatz
ja genau dazu
ja oder k~onnen Sie mir nicht ~ubernimmt noch was ~ubersehen aufgef~uhrt wird
vorher oder so das ist oder soll ich f~ur zwei Wochen wovon
eine Woche fahren

guten Tag

obwohl irgendwo vielleicht dauern w~urde mir auch den Telefonnummer geben

und
okay
ja
ja das w~are auch gut
lieber Hannover

ich wiederhole Ball wir Telefonnummer
reisen wollen wir fertig ja f~unf sechs acht acht neun neun
gut okay dann vielen Dank f~ur die Auskunft
und auf Wiederh~oren
ja drei auf einen Besuch

Appendix B Medical Assistance Domain Evaluation Sets

B.1 Transcribed English Evaluation Utterances

yes i i'm trying to reach someone to talk to about some problems that i've been having lately
well i've been really feeling pretty badly for the past couple of days had a real bad headache and stomach problems with vomiting and it is just it has been a horrible time for me really
i'm fifty eight

well i have had asthma and rheumatism in the past
i'm taking motrin anti inflammatories and bronchial dialating medicine
something for the asthma that they prescribed for me
i'm doing the puff
well i've had a really bad headache kind of in the back of my head and like i said before i've been vomiting and really there is nothing more to vomit i've just basically into the dry heaves and the diarrhea and i've also had a cough for a couple of days too
yes i'm allergic to pollen and dust
just what i told you before the anti inflammatory and the the the puff the asthma medication
mhm
mhm

no not really i haven't been real hungry lately though
yes i have had diarrhea
about six or eight times a day
yes
yes
i'm trying to but you know nothing really tastes good and whenever i drink anything i feel like i could throw it back up
okay that sounds good
hello i'm having some pain in my chest today
it is it is pretty painful
it is about seven
it doesn't last it started this afternoon it only lasts about two to five minutes when it happens
yes i'm still here it it doesn't seem to matter if i'm exercising or not
yes i am
it is mostly right about in here but it also goes over like this
sometimes i have pins and needles in my arm
all the way down here
yes i am
yes yes i get a sweat also
i'm forty five years old
well i have sugar diabetes
no i i get shots of insulin
yes i do
i smoke about one and a half packs a day
since i was in my early twenties
i'm forty five years old
i like to drink a bit probably more than i should
okay i think i can do that thank you very much

B.2 Automatically Recognized English Evaluation Utterances

SHE NINE TEETH AND TRY TO REACH SOMEONE TO TALK TO THEM SOME PROBLEMS THAN IN PASSING LATELY
SEE I DIDN'T REALLY FEELING PRETTY BADLY FOR THE PAST COUPLE OF PAINS ABOUT THAT HATE CAN SEND THE PROBLEM IS I MEAN THAT'S JUST A HORRIBLE TIME FOR ME REALLY
AND FIFTEENTH

WELL I HAVE HAD ASTHMA AND RHEUMATISM IN THE PAST
HI I AM TAKING MOTRIN I TAKE FROM IT SURE USE A HAND UP FRONT DON'T FEEL BADLY IN MEDICINE SOMETHING FOR THE ASTHMA ACHES CRACK FOR ME TO THE
I AM DOING THE PULSE
WELL I I FELT LIKE THAT ANY KIND OF IN THE BACK OF MY HAND AND I SAID BEFORE I CAN VOMITING AND REALLY THERE'S NOTHING MORE TO PALM THAT JUST BASICALLY INTO THAT DRY HE'D SEND THE DIARRHEA AND ALSO TO COUGH A COUPLE OF DAYS TOO
YES I LOOKED POLLEN AND TEST
I JUST THOUGHT I COULD YOU BE FOR IT BE INTENSE AND TRYING TO THE TOP OF THAT MEDICATION
MHM
MHM

NO NOT REALLY AT I HAPPEN TO HUNGER IN THE EVENING
YES I HAVE KEPT AREA
AND THAT IS SIX THREE TIMES A DAY
YES
YES
I AM TRYING TO BET ENOUGH CAN REALLY TEACH THAT AND NEVER DRINK ANYTHING I FEEL LIKE I COULD THROW IT BACK
OKAY THAT SOUNDS GOOD
HELLO I AM HAVING SOME PAIN IN MY CHEST
IS THAT IS PRETTY PAINFUL
IT IS ABOUT SEVEN
IT DOESN'T LAST IT STARTED THIS AFTERNOON IS ONLY LASTS ABOUT TWO TO FIVE MINUTES WHEN IT HAPPENS
YES I AM STILL HERE IT IS DOESN'T SEEM TO MATTER IF I AM EXERCISING NOW
YES I AM
IT IS MOSTLY RIGHT ABOUT IN THE YEAR BUT IT ALSO GOES OVER LIKE THIS
SOMETIMES I HAVE PINS AND NEEDLES IN MY ARM
ALL THE WAY DOWN HERE
YES I AM
YES YES I DO TO SWEAT ALSO
I AM FORTY FIVE YEARS OLD
WELL I HAVE SUGAR DIABETES
NO I I GET SHOTS OF INSULIN
YES I DO
I SMOKE ABOUT ONE AND A HALF PACKS A DAY
NO SINCE I WAS IN MY EARLY TWENTIES
I AM FORTY FIVE YEARS OLD
HI EVEN LIKE TO DRINK IT PROBABLY MORE THAN I SHOULD
OKAY I THINK I CAN DO THAT THANK YOU VERY MUCH

B.3 Transcribed English Grammar Development Utterances

yeah hi doctor i'm elizabeth i'm just having some headaches and some fever today wondering if you could give me some advice tell me what i should do

the headaches are happening across both sides of my head behind my eyes it has been going on for the past two days and along with a fever
it was a hundred and two this morning i haven't taken it since then
i'm feeling achy all over my muscles just hurt all over my arms my legs and i just feel real bad
oh it is all all started with the headaches and with the fever about the same time
no no no chills
no i my children have been vomiting and diarrhea 'cause they're they're having a cough and they're having some fevers as well but but i'm not having any vomiting or diarrhea
yes they they got sick about one day later than me
no nothing like that
how do i direct it
oh okay

yeah i see it it is man that is real sensitive isn't it okay it is real sensitive it is up here above my eye and above my ear on both sides that is real sensitive
this is about a five right now
no i'm not taking any medications at all
no i no i'm in good health otherwise
no
i usually don't go to the doctor i haven't had much much reason to
no i don't smoke
just occasionally socially
no thank you doctor
yes hi this is mister rick jones and is this the emergency room
yes i'm having some problem today my chest really hurts and i'm wondering if i should come in to see a doctor
oh i think i'm about maybe sixty five or sixty six i don't i don't count birthdays anymore you know
i drink my milk every day
intoxicated what does that mean
oh no no no no no my church don't believe in that
yes i do smoke
oh gosh probably twenty twenty five years
no sir no other medical problems
my chest pain it hurts in the middle of my chest and i had it this morning when i got out of bed for the first time i'll try to draw you a picture
yes i do but that man is much smaller than i am
i'm sorry radiate what does that mean
oh yes actually it does it went up into my neck this morning and i think it went into my arm 'cause my arm went kind of numb this afternoon
yes
i took some well i tried to take some tylenol but i couldn't find it no no medicine
uh huh
no it feels pretty sharp it hurts like you know it stings and burns a lot
wow that is a good question doc probably it was about an eight yeah but it doesn't feel like that now now it is more like a ten
no when i'm sitting down my feet up i'm okay
i did get a little short of breath just couldn't quite breathe like i wanted to and that is what made me really call you guys
nine one one that is
oh
yes yes i sure do

yes
okay thank you very much
okay
yeah hi i've been having some problems and my buddy convinced me that i should try to call someone and find out what is going on i just i really don't know something is happening and it sometimes it kinda scares me
well i've been having a really a strong pain in my chest and it changes sometimes it gets better sometimes it gets worse and it just it comes at times that you know i'm not expecting it and sometimes it hurts a lot and i'm i'm just afraid that something is gonna happen to me when i when i'm least expecting it
i'm sixty five
i'm a truck driver
as far as i know no

not that i know of not that i'm aware of i haven't really talked about it 'cause i i just really didn't even think i had any heart problems
no
i'm five foot six inches
approximately two hundred twenty pounds
unfortunately yes i do smoke i i've been trying to quit but i just can't
let us see about two packs a day
no as far as i
why you know i don't think i've had it checked i've been a healthy person so i really haven't gone to the doctor
yes
yes i'm here
right
yes
yes
yes sometimes i just can't catch my breath when i'm walking
mhm
mhm
sometimes it gets better after you know i'll stop for a little bit and then sometimes it gets a little bit better when i'm resting
not really it is mostly during the day that it kinda comes and goes
right well it is lot of times when i'm walking and doing things you know a little bit active then i then i also get stomach pains with it really which is kind of different
no i don't think so it hasn't appeared with with my meals
well sometimes it goes up to my neck and around my jaw area and in my upper arms
i think so it is hard it is it is just hard to remember really
no not really i just kinda feel it up there sometimes and sometimes it does get it is a lot stronger than others too
yes
yes it it is a really strong pressure right in the middle
right right in the middle of my chest

no not really
mhm
okay thank you very much appreciate your help
hi this is misses jones i'm calling because i'm having a lot of pain
oh i got them in my muscles i got a slight fever too
i had the fever i've had for the last two days and i have a headache a tension headache in the back of my head
yes i did it was a hundred degrees

have a dry cough a cold and runny nose vomiting diarrhea i don't have much of an appetite i have throat pain my throat hurts when i swallow

no it is just what i've been eating my sister has similar symptoms but she doesn't have vomiting or diarrhea
no
i have asthma and rheumatism
i take an anti inflammatory and a bronchial dilating medication for the asthma
not to drugs but to pollen and dust
okay thank you

Bibliography

[Alexandersson *et al.*, 1998] J. Alexandersson, B. Buschbeck-Wolf, T. Fujinami, M. Kipp, S. Koch, E. Maier, N. Reithinger, B. Schmitz, and M. Siegel. Dialogue Acts in VERBMOBIL-2 Second Edition. *Verbmobil Report 226*, DFKI Saarbrücken, Universität Stuttgart, TU Berlin, and Universität des Saarlandes, 1998.

[Alshawi, 1992] H. Alshawi (Ed.). *The Core Language Engine*, MIT Press, 1992.

[Amengual *et al.*, 2000] J. C. Amengual, J. M. Benedí, F. Casacuberta, A. Castaño, A. Castellanos, V. M. Jiménez, D. Llorens, A. Marzal, M. Pastor, F. Prat, E. Vidal, and J. M. Vilar. The EUTRANS-I Speech Translation System. *Machine Translation*, Vol. 15, Issue 1-2, pp. 75-103, June 2000.

[Black *et al.*, 2001] A. Black, P. Taylor, and R. Caley. The Festival Speech Synthesis System: System Documentation Edition 1.4, for Festival Version 1.4.2. Available from http://festvox.org/festival/ or http://www.cstr.ed.ac.uk/projects/festival/.

[Brill, 1995] E. Brill. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics*, 21(4), pp. 543-566, 1995.

[Brugnara and Federico, 1997] F. Brugnara and M. Federico. Dynamic Language Models for Interactive Speech Applications. In *Proceedings of the 5th European Conference on Speech Communication and Technology*, Rhodes, Greece, 1997.

[Bub *et al.*, 1997] T. Bub, W. Wahlster, and A. Waibel. Verbmobil: The Combination of Deep and Shallow Processing for Spontaneous Speech Translation. In *Proceedings of the IEEE 1997 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-97)*, Munich, Germany, April 1997.

[Bub and Schwinn, 1996] T. Bub and J. Schwinn. VERBMOBIL: The Evolution of a Complex Large Speech-to-Speech Translation System. In *Proceedings of The Fourth International Conference on Spoken Language Processing (ICSLP-96)*, Philadelphia, PA, October 1996.

[Burger *et al.*, 2003] S. Burger, V. MacLaren, N. Mana, and K. Peterson. Annotated Data for the Second Showcase, NESPOLE! Project Deliverable D14 (Internal Document), 2003.

[Burger *et al.*, 2001] S. Burger, L. Besacier, P. Coletti, F. Metze, and C. Morel. The NESPOLE! VoIP Dialogue Database. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech-2001)*, Aalborg, Denmark, September 2001.

[Casacuberta *et al.*, 2001] F. Casacuberta, D. Llorens, C. Martínez, S. Molau, F. Nevado, H. Ney, M. Pastor, D. Picó, A. Sanchis, E. Vidal, and J. M. Vilar. Speech-to-Speech Translation Based on Finite-State Transducers. In *Proceedings of the IEEE 2001 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-01)*, Salt Lake City, UT, May 2001.

[Cattoni *et al.*, 2001] R. Cattoni, M. Federico, and A. Lavie. Robust Analysis of Spoken Input Combining Statistical and Knowledge-Based Information Sources. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop (ASRU-2001)*, Trento, Italy, December 2001.

[Costantini *et al.*, 2002] E. Costantini, F. Pianesi, and S. Burger. The Added Value of Multimodality in the NESPOLE! Speech-to-Speech Translation System: an Experimental Study. In *Proceedings of the International Conference on Multimodal Interfaces (ICMI-2002)*, Pittsburgh, PA, October 2002.

[C-STAR] C-STAR Consortium Website. http://www.c-star.org.

[Daelemans *et al.*, 2002] W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. TiMBL: Tilburg Memory Based Learner, version 4.3, Reference Guide. *ILK Technical Report 02-10*, 2002. Available from http://ilk.uvt.nl/downloads/pub/papers/ilk.0210.ps.

[Daelemans *et al.*, 2000] W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. TiMBL: Tilburg Memory Based Learner, version 3.0, Reference Guide. *ILK Technical Report 00-01*, 2000. Available from http://ilk.kub.nl/~ilk/papers/ilk0001.ps.gz.

[Dorr *et al.*, 1999] B. J. Dorr, P. W. Jordan, and J. W. Benoit. A Survey of Current Paradigms in Machine Translation. In *Advances in Computers*, Vol. 49, M. Zelkowitz (Ed.), Academic Press, London, pp. 1-68, 1999.

[Elman, 1990] J. L. Elman. Finding Structure in Time. *Cognitive Science*, 14, pp. 179-211, 1990.

[Fügen *et al.*, 2001] C. Fügen, M. Westphal, M. Schneider, T. Schultz, and A. Waibel. LingWear: A Mobile Tourist Information System. In *Proceedings of the 1st International Human Language Technology Conference (HLT-2001)*, San Diego, CA, March 2001.

[Fukada *et al.*, 1998] T. Fukada, D. Koll, A. Waibel, and K. Tanigaki. Probabilistic Dialogue Act Extraction for Concept Based Multilingual Translation Systems. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*, Sydney, Australia, 1998.

[Gavaldà, 2000] M. Gavaldà. SOUP: A Parser for Real-World Spontaneous Speech. In *Proceedings of the 6th International Workshop on Parsing Technologies (IWPT-2000)*, Trento, Italy, February 2000.

[Godfrey *et al*., 1992] J. J. Godfrey, E. C. Holliman, and J. McDaniel. SWITCHBOARD: Telephone Speech Corpus for Research and Development. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP-92)*, San Francisco, CA, March 1992.

[Gotoh and Renals, 2000] Y. Gotoh and S. Renals. Sentence Boundary Detection in Broadcast Speech Transcripts. In *Proceedings of the International Speech Communication Association (ICSA) Workshop: Automatic Speech Recognition: Challenges for the New Millennium (ASR-2000)*, Paris, France, September 2000.

[Guerzoni *et al*., 2003] F. Guerzoni, G. Lazzari, and F. Pianesi. NESPOLE! Final Report (Public Document), 2003.

[Han and Lavie, 2003] B. Han and A. Lavie. UKernel: A Unification Kernel. *Technical Report CMU-LTI-03-177*, Language Technologies Institute, Carnegie Mellon University, 2003.

[Hutchins and Somers, 1992] W. J. Hutchins and H. L. Somers. *An Introduction to Machine Translation*. Academic Press, 1992.

[Jekat *et al*., 1995] S. Jekat, A. Klein, E. Maier, I. Maleck, M. Mast, and J. J. Quantz. Dialogue Acts in VERBMOBIL. *Verbmobil Report 65*, Universität Hamburg, DFKI Saarbrücken, Universität Erlangen, and TU Berlin, 1995.

[Jurafsky *et al*., 1997] D. Jurafsky, E. Shriberg, and D. Biasca. Switchboard SWBD-DAMSL Shallow-Discourse-Function Annotation Coders Manual. Available from http://www.colorado.edu/ling/jurafsky/manual.august1.html.

[Kipp, 1998] M. Kipp. The Neural Path to Dialogue Acts. In *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, Brighton, United Kingdom, August 1998.

[Kuhn and De Mori, 1995] R. Kuhn and R. De Mori. The Application of Semantic Classification Trees to Natural Language Understanding. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 5, May 1995.

[Lavie *et al*., 2002] A. Lavie, F. Metze, F. Pianesi, S. Burger, D. Gates, L. Levin, C. Langley, K. Peterson, T. Schultz, A. Waibel, D. Wallace, J. McDonough, H. Soltau, R. Cattoni, G. Lazzari, N. Mana, E. Pianta, E. Costantini, L. Besacier, H. Blanchon, D. Vaufreydaz, and L. Taddei. Enhancing the Usability and Performance of NESPOLE! - a Real-World Speech-to-Speech Translation System. In *Proceedings of HLT-2002 Human Language Technology Conference*, San Diego, CA, March 2002.

[Lavie *et al*., 2001] A. Lavie, C. Langley, A. Waibel, F. Pianesi, G. Lazzari, P. Coletti, L. Taddei, and F. Balducci. Architecture and Design Considerations in NESPOLE!: a Speech Translation System for E-commerce Applications. In *Proceedings of the 1st International Human Language Technology Conference (HLT-2001)*, San Diego, CA, March 2001.

[Lavie *et al.*, 1997a] A. Lavie, L. Levin, P. Zhan, M. Taboada, D. Gates, M. Lapata, C. Clark, M. Broadhead, and A. Waibel. Expanding the Domain of a Multi-lingual Speech-to-Speech Translation System. In *Proceedings of Workshop on Spoken Language Translation, Joint Meeting of ACL/EACL-97*, Madrid, Spain, July 1997.

[Lavie *et al.*, 1997b] A. Lavie, D. Gates, N. Coccaro, and L. Levin. Input Segmentation of Spontaneous Speech in JANUS: a Speech-to-speech Translation System. In *Dialogue Processing in Spoken Language Systems: Revised Papers from ECAI-96 Workshop*, E. Maier, M. Mast, and S. Luperfoy (Eds.), LNCS series, Springer Verlag, June 1997.

[Lavie *et al.*, 1997c] A. Lavie, A. Waibel, L. Levin, M. Finke, D. Gates, M. Gavaldà, T. Zeppenfeld, and P. Zhan. JANUS-III: Speech-to-Speech Translation in Multiple Languages. In *Proceedings of the IEEE 1997 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-97)*, Munich, Germany, April 1997.

[Lavie *et al.*, 1996] A. Lavie, A. Waibel, L. Levin, D. Gates, M. Gavaldà, T. Zeppenfeld, P. Zhan, and O. Glickman. Translation of Conversational Speech with JANUS-II. In *Proceedings of The Fourth International Conference on Spoken Language Processing (ICSLP-96)*, Philadelphia, PA, October 1996.

[Lavie, 1996a] A. Lavie. GLR*: A Robust Parser for Spontaneously Spoken Language. In *Proceedings of ESSLLI-96 Workshop on Robust Parsing*, Prague, Czech Republic, August 1996.

[Lavie, 1996b] A. Lavie. GLR*: A Robust Grammar-Focused Parser for Spontaneously Spoken Language. PhD dissertation, *Technical Report CMU-CS-96-126*, Carnegie Mellon University, Pittsburgh, PA, May 1996.

[Lendvai *et al.*, 2003] P. Lendvai, A. van den Bosch, and E. Krahmer. Machine Learning for Shallow Interpretation of User Utterances in Spoken Dialogue Systems. In *Proceedings of the EACL-03 Workshop on Dialogue Systems: Interaction, Adaptation and Styles of Management*. Budapest, Hungary, April 2003.

[Levin *et al.*, 2003a] L. Levin, D. Gates, D. Wallace, K. Peterson, E. Pianta, N. Mana. The NESPOLE! Interchange Format. NESPOLE! Project Deliverable D13 (Public Document), 2003. http://nespole.itc.it/public/devilerables/D13_final.doc.

[Levin *et al.*, 2003b] L. Levin, C. Langley, A. Lavie, D. Gates, D. Wallace, and K. Peterson. Domain Specific Speech Acts for Spoken Language Translation. In *Proceedings of 4th SIGdial Workshop on Discourse and Dialogue*, Sapporo, Japan, July 2003.

[Levin *et al.*, 2002] L. Levin, D. Gates, D. Wallace, K. Peterson, A. Lavie, F. Pianesi, E. Pianta, R. Cattoni, N. Mana. Balancing Expressiveness and Simplicity in an Interlingua for Task Based Dialogue. In *Proceedings of Workshop on Speech-to-Speech Translation: Algorithms and Systems at the 40th Annual Meeting of the Association of Computational Linguistics (ACL-02)*, Philadelphia, PA, July 2002.

[Levin *et al*., 2000a] L. Levin, A. Lavie, M. Woszczyna, D. Gates, M. Gavaldà, D. Koll, and A. Waibel. The Janus-III Translation System. *Machine Translation*, Vol. 15, Issue 1-2, pp. 3-25, June 2000.

[Levin *et al*., 2000b] L. Levin, D. Gates, A. Lavie, F. Pianesi, D. Wallace, T. Watanabe, and M. Woszczyna. Evaluation of a Practical Interlingua for Task-Oriented Dialogue. In *Workshop on Applied Interlinguas: Practical Applications of Interlingual Approaches to NLP*, Seattle, 2000.

[Levin *et al*., 1998] L. Levin, D. Gates, A. Lavie, and A. Waibel. An Interlingua Based on Domain Actions for Machine Translation of Task-Oriented Dialogues. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*, Sydney, Australia, 1998.

[Macon *et al*., 1998] M. W. Macon, A. Kain, A. E. Cronk, H. Meyer, K. Mueller, B. Saeuberlich, and A. W. Black. Rapid Prototyping of a German TTS System. *Technical Report CSE-98-015*, Department of Computer Science, Oregon Graduate Institute of Science and Technology, Portland, OR, September 1998.

[Macon *et al*., 1997] M. W. Macon, A. E. Cronk, J. Wouters, and A. Kain. OGIresLPC: Diphone synthesizer using residual-excited linear prediction. *Technical Report CSE-97-007*, Department of Computer Science, Oregon Graduate Institute of Science and Technology, Portland, OR, September 1997.

[Mast *et al*., 1996] M. Mast, R. Kompe, S. Harbeck, A. Kießling, H. Niemann, E. Nöth, E. G. Schukat-Talamazzini, and V. Warnke. Dialog Act Classification with the Help of Prosody. In *Proceedings of the Fourth International Conference on Spoken Language Processing (ICSLP-96)*, Philadelphia, PA, October 1996.

[Mast *et al*., 1995] M. Mast, H. Niemann, E. Nöth, and E. G. Schukat-Talamazzini. Automatic Classification of Dialog with Semantic Classification Trees and Polygrams. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-95), Workshop on New Approaches to Learning for Natural Language Processing*, Montreal, Quebec, Canada, August 1995.

[Mayfield *et al*., 1995a] L. Mayfield, M. Gavaldà, Y-H. Seo, B. Suhm, W. Ward, and A. Waibel. Parsing Real Input in JANUS: A Concept-Based Approach to Spoken Language Translation. In *Proceedings of The Sixth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-95)*, Leuven, Belgium, July 1995.

[Mayfield *et al*., 1995b] L. Mayfield, M. Gavaldà, W. Ward, and A. Waibel. Concept-Based Speech Translation. In *Proceedings of the IEEE 1995 International Conference on Acoustics, Speech and Signal Processing (ICASSP 95)*, Detroit, Michigan, May 1995.

[Metze *et al*., 2002] F. Metze, J. McDonough, H. Soltau, A. Waibel, A. Lavie, S. Burger, C. Langley, L. Levin, T. Schultz, F. Pianesi, R. Cattoni, G. Lazzari, N. Mana, and E. Pianta. The NESPOLE! Speech-to-Speech Translation System. In *Proceedings of HLT-2002 Human Language Technology Conference*, San Diego, CA, March 2002.

[Mikheev, 2000] A. Mikheev. Tagging Sentence Boundaries. In *Proceedings of the First Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2000)*, Seattle, WA, 2000.

[Mikheev, 1997] A. Mikheev. LT POS – the LTG part of speech tagger. Language Technology Group, University of Edinburgh. http://www.ltg.ed.ac.uk/software/pos.

[Munk, 1999] M. Munk. Shallow Statistical Parsing for Machine Translation. Diploma Thesis, Karlsruhe University, May 1999.

[NESPOLE!] NESPOLE! Project Website. http://nespole.itc.it.

[NESPOLE!-IF-Spec] NESPOLE! Interchange Format Specification. Available from http://www.is.cs.cmu.edu/nespole/db/specification.html.

[NESPOLE!-D8, 2001] ITC-irst, UKA, CMU, UJF, AETHRA. First Showcase Documentation. NESPOLE! Project Deliverable D8 (Public Document), 2001. Available from http://nespole.itc.it/public/deliverables/D8-a.zip.

[NESPOLE!-D18, 2003] CMU, ITC-irst, UKA, UJF, AETHRA. Evaluation of the NESPOLE! Showcase-2a System. NESPOLE! Project Deliverable D18 (Public Document), 2003. Available from http://nespole.itc.it/public/deliverables/D18_final.doc.

[Tomita and Nyberg, 1988] M. Tomita and E. Nyberg. Generation Kit and Transformation Kit, Version 3.2: User's Manual. *Technical Report CMU-CMT-88-MEMO*, Carnegie Mellon University, Pittsburgh, PA, 1988.

[Osterholtz *et al.*, 1992] L. Osterholtz, C. Augustine, A. McNair, H. Saito, T. Sloboda, J. Tebelskis, A. Waibel, and M. Woszczyna. Testing Generality in JANUS: A Multi-Lingual Speech Translation System. In *Proceedings of the IEEE 1992 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-92)*, San Francisco, CA, 1992.

[Palmer and Hearst, 1997] D. D. Palmer and M. A. Hearst. Adaptive Multilingual Sentence Boundary Disambiguation. *Computational Linguistics*, 23(2), June 1997.

[Pastor *et al.*, 2001] M. Pastor, A. Sanchis, F. Casacuberta, and E. Vidal. EUTRANS: a Speech-to-Speech Translator Prototype. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech-2001)*, Aalborg, Denmark, September 2001.

[Qu *et al.*, 1996a] Y. Qu, C. P. Rose, B. DiEugenio. Using Discourse Predictions for Ambiguity Resolution. In *Proceedings of 16th International Conference on Computational Linguistics (COLING-96)*, Copenhagen, Denmark, August 1996.

[Qu *et al.*, 1996b] Y. Qu, B. DiEugenio, A. Lavie, L. Levin, and C. P. Rose. Minimizing Cumulative Error in Discourse Context. In *Proceedings of ECAI-96 Workshop on Dialogue Processing in Spoken Language Systems*, Budapest, Hungary, August 1996.

[Quinlan, 1993] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[Rayner *et al*., 2000] M. Rayner, D. Carter, P. Bouillon, V. Digalakis, and M. Wirén. *The Spoken Language Translator*. Cambridge University Press, 2000.

[Rayner and Carter, 1997] M. Rayner and D. Carter. Hybrid Language Processing in the Spoken Language Translator. In *Proceedings of the IEEE 1997 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-97)*, Munich, Germany, April 1997.

[Rayner and Carter, 1996] M. Rayner and D. Carter. Fast Parsing Using Pruning and Grammar Specialization. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, Santa Cruz, CA, June 1996.

[Reithinger and Engel, 2000] N. Reithinger and R. Engel. Robust Content Extraction for Translation and Dialog Processing. In W. Wahlster (Ed.), *Verbmobil: Foundations of Speech-to-Speech Translation*, Springer, pp. 428-437, 2000.

[Reithinger and Klesen, 1997] N. Reithinger and M. Klesen. Dialogue Act Classification Using Language Models. In *Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech-97)*, Rhodes, Greece, September 1997.

[Reynar and Ratnaparkhi, 1997] J. C. Reynar and A. Ratnaparkhi. A Maximum Entropy Approach to Identifying Sentence Boundaries. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, Washington, D.C., April 1997.

[Samuel *et al*., 1998a] K. Samuel, S. Carberry, and K. Vijay-Shanker. Dialogue Act Tagging with Transformation-Based Learning. In *Proceedings of the 17th International Conference on Computational Linguistics and the 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL-98)*. Montreal, Quebec, Canada, August 1998.

[Samuel *et al*., 1998b] K. Samuel, S. Carberry, and K. Vijay-Shanker. Computing Dialogue Acts from Features with Transformation-Based Learning. In *Applying Machine Learning to Discourse Processing: Papers from the 1998 American Association for Artificial Intelligence Spring Symposium*. Stanford, CA, March 1998.

[Shriberg *et al*., 1998] E. Shriberg, R. Bates, A. Stolcke, P. Taylor, D. Jurafsky, K. Ries, N. Coccaro, R. Martin, M. Meteer, and C. Van Ess-Dykema. Can Prosody Aid the Automatic Classification of Dialog Acts in Conversational Speech?. *Language and Speech*, 41(3-4), pp. 439-487, 1998.

[Soltau *et al*., 2001a] H. Soltau, F. Metze, C. Fügen, and A. Waibel. A One-Pass Decoder Based on Polymorphic Linguistic Context Assignment. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop (ASRU-2001)*, Trento, Italy, December 2001.

[Soltau *et al*., 2001b] The ISL Evaluation System for Verbmobil-II. In *Proceedings of the IEEE 2001 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-01)*, Salt Lake City, UT, May 2001.

[Stolcke *et al*., 2000] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. Van Ess-Dykema, and M. Meteer. Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, 26(3), pp. 339-373, 2000.

[Suhm *et al*., 1994] B. Suhm, L. Levin, N. Coccaro, J. Carbonell, K. Horiguchi, R. Isotani, A. Lavie, L. Mayfield, C. P. Rose, C. Van Ess-Dykema, and A. Waibel. Speech-Language Integration in a Multi-Lingual Speech Translation System. In *Proceedings of AAAI-94 Workshop on Integration of Natural Language and Speech Processing*, Seattle, WA, August 1994.

[Suhm *et al*., 1995] B. Suhm, P. Geutner, T. Kemp, A. Lavie, L.J. Mayfield, A.E. McNair, I. Rogina, T. Sloboda, W. Ward, M. Woszczyna, and A. Waibel. JANUS: Towards Multilingual Spoken Language Translation. In *Proceedings of ARPA Workshop on Spoken Language Technology*, 1995.

[Stamatatos *et al*., 1999] E. Stamatatos, M. Fakotakis, and G. Kokkinakis. Automatic Extraction of Rules for Sentence Boundary Disambiguation. In *Proceedings of the Workshop in Machine Learning in Human Language Technology, Advance Course on Artificial Intelligence (ACAI-99)*, Chania, Greece, July 1999.

[Stevenson and Gaizauskas, 2000] M. Stevenson and R. Gaizauskas. Experiments on Sentence Boundary Detection. In *Proceedings of the Sixth Conference on Applied Natural Language Processing and First Conference of the North American Chapter of the Association for Computational Linguistics (ANLP-NAACL-2000)*, Seattle, WA, 2000.

[Taddei *et al*., 2003] L. Taddei, L. Besacier, R. Cattoni, E. Costantini, A. Lavie, N. Mana, and E. Pianta. Second Showcase Documentation. NESPOLE! Project Deliverable D17 (Public Document), 2003. http://nespole.itc.it/public/deliverables/D17-final.zip.

[Taddei *et al*., 2002] L. Taddei, E. Costantini, and A. Lavie. The NESPOLE! Multimodal Interface for Cross-lingual Communication - Experience and Lessons Learned. In *Proceedings of ICMI 2002 International Conference on Multimodal Interfaces*, Pittsburgh, USA, October 2002.

[van Rijsbergen, 1979] C. J. van Rijsbergen. *Information Retrieval*. 2nd Edition, London: Butterworths, 1979.

[Waibel *et al*., 1997] A. Waibel, A. Lavie, and L. Levin. JANUS: A System for Translation of Conversational Speech. *Kuenstliche Intelligenz (KI)*, 97(4), 1997.

[Waibel *et al*., 1992] A. Waibel, A. N. Jain, A. McNair, J. Tebelskis, L. Osterholtz, H. Saito, O. Schmidbauer, T. Sloboda, and M. Woszczyna. JANUS: Speech-to-Speech Translation Using Connectionist and Non-Connectionist Techniques. *Advances in Neural Information Processing Systems*, 4, 1992.

[Wahlster, 2000] W. Wahlster (Ed.). *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, 2000.

[Walker *et al*., 2001] D. J. Walker, D. E. Clements, M. Darwin, and J. W. Amtrup. Sentence Boundary Detection: A Comparison of Paradigms for Improving MT Quality. In *Proceedings of MT Summit VIII*, Santiago de Compostela, Spain, September 2001.

[Warnke *et al*., 1997] V. Warnke, R. Kompe, H. Niemann, and E. Nöth. Integrated Dialog Act Segmentation and Classification Using Prosodic Features and Language Models. In *Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech-97)*, Rhodes, Greece, September 1997.

[Wermter and Weber, 1997] S. Wermter and V. Weber. SCREEN: Learning a Flat Syntactic and Semantic Spoken Language Analysis Using Artificial Neural Networks. *Journal of Artificial Intelligence Research*, Vol. 6, pp. 35-85, January 1997.

[Wermter and Löchel, 1996] S. Wermter and M. Löchel. Learning Dialog Act Processing. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, August 1996.

[Woszczyna *et al*., 1998] M. Woszczyna, M. Broadhead, D. Gates, M. Gavaldà, A. Lavie, L. Levin, and A. Waibel. A Modular Approach to Spoken Language Translation for Large Domains. In *Proceedings of the 3rd Conference of the Association for Machine Translation in the Americas (AMTA-98)*, Langhorne, PA, 1998.

[Woszczyna *et al*., 1994] M. Woszczyna, N. Aoki-Waibel, F. D. Buø, N. Coccaro, K. Horiguchi, T. Kemp, A. Lavie, A. McNair, T. Polzin, I. Rogina, C. P. Rose, T. Schultz, B. Suhm, M. Tomita and A. Waibel. "JANUS 93: Towards Spontaneous Speech Translation". In *Proceedings of the IEEE 1994 International Conference on Acoustics, Speech, and Signal Processing (ICASSP-94)*, Adelaide, Australia, April 1994.

[Woszczyna *et al*., 1993] M. Woszczyna, N. Coccaro, A. Eisele, A. Lavie, A. McNair, T. Polzin, I. Rogina, C.P. Rose, T. Sloboda, M. Tomita, J. Tsutsumi, N. Aoki-Waibel, A. Waibel, and W. Ward. Recent Advances in JANUS: A Speech Translation System. In *Proceedings of the 3rd European Conference on Speech Communication and Technology (Eurospeech-93)*, Berlin, Germany, September 1993.

[Zavrel, 1997] J. Zavrel. An Empirical Re-Examination of Weighted Voting for k-NN. In *Proceedings of the 7th Belgian-Dutch Conference on Machine Learning*, W. Daelemans, P. Flach, and A. van den Bosch (Eds.), Tilburg, 1997.

[Zell *et al*.] A. Zell, G. Mamier, M. Vogt, N. Mache, R. Hübner, S. Döring, K. Herrman, T. Soyez, M. Schmalzl, T. Sommer, A. Hatzigeorgiou, D. Posselt, T. Schreiner, B. Kett, G. Clemente, J. Wieland, J. Gatter. SNNS Stuttgart Neural Network Simulator User Manual, Version 4.2.