Thesis Proposal
Context Awareness and Personalization in Dialogue Planning

Robert W. H. Fisher

May 13, 2014

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Reid Simmons, Chair
Geoffrey Gordon
Carolyn Penstein Rosé
Dieter Fox, University of Washington

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

Copyright © 2014 Robert W. H. Fisher
Keywords: Dialogue Planning, Spectral Methods, Intelligent Tutoring Systems, Context Awareness
Thanks to all my friends and family for their support, particularly my wife Katherine, without whom none of this would be possible.
Abstract

In this thesis proposal, we present techniques to improve the effectiveness of a conversational intelligent agent by incorporating rich, contextual information into the dialogue planning process. First, we propose the usage of linguistic features derived from semantic and discourse parsing to better understand the structure of a dialogue with a user. Discourse parsing gives us the relational structure of a dialogue and can be used to identify inter-sentential references and connections in a conversation. We have developed a novel, spectral algorithm for discourse parsing that is statistically consistent and computationally efficient. We also suggest the incorporation of non-linguistic contextual factors, such as a user's physical engagement or the aural characteristics of the environment. Modeling the diverse array of social factors we wish to include requires a very expressive state space representation. To account for this, we propose dialogue planning using contextual information stored in a Contextual Knowledge Base (CKB) that can be combined with Predictive State Representations and imitation learning techniques. A conversational agent with access to a Contextual Knowledge Base will be able to respond to environmental and linguistic conditions in ways that a simple question answering agent is not capable of. Our work is motivated by the education domain, and we intend for our methods to be included in a conversational intelligent tutoring system. By incorporating information about the student, the environment, and the state of the dialogue into a planning system, we hope to reduce the collaborative effort required to teach new concepts and to improve the overall effectiveness of conversational tutoring systems.
Acknowledgments

I would chiefly like to acknowledge my advisor Reid Simmons, and the rest of my thesis committee, Geoff Gordon, Carolyn Rosé, and Dieter Fox. Thanks also to the many colleagues who have provided valuable contributions to this work. In particular Tom Kollar, Ankhur Parikh, Maxim Makatchev, Eleanor Avrunin, Max Tomlinson, Miriam Robinson, Jayant Krishnamurthy, and the entire staff of the Human Engineering Research Laboratory. Finally thanks to the National Science Foundation Graduate Fellowship program, the Quality of Life Technology Center, and the American Rewards for College Scientists program for their generous financial support.
# Contents

1 Introduction ........................................... 1  
   1.1 Educational Tutoring Application ................. 2  

2 Related Work ........................................... 5  

3 Overview of Approach ................................ 9  

4 Extracting Contextual Features from Text and Sensor Data ............ 13  
   4.1 Semantic Features .................................. 13  
      4.1.1 Syntactic Features ............................. 15  
      4.1.2 Lexical Features ............................... 16  
      4.1.3 Mapping Sentences to MRLs with Integer Linear Programming .... 16  
      4.1.4 Active Learning & Bootstrapping ............... 17  
   4.2 Semantic Parsing Results .......................... 20  
   4.3 Discourse Features .................................. 22  
      4.3.1 Problem Definition and Dataset ................. 23  
      4.3.2 Spectral Algorithm ............................ 24  
      4.3.3 Spectral Parsing Results ....................... 26  
   4.4 Non-Linguistic Contextual Features ............... 27  
      4.4.1 Posture Recognition ............................ 27  
      4.4.2 Voice Activity Detection ....................... 28  
      4.4.3 Smooth Constraint Learning .................... 28  
   4.5 Proposed Work in Linguistic Feature Extraction ............. 31  
      4.5.1 Weakly-labeled and unlabeled discourse parsing ......... 31  
      4.5.2 New Discourse Parsing Frameworks .............. 32  
      4.5.3 Dialogue Personalization ....................... 33  

5 Proposed Work in Dialogue Planning with Contextual Social Features ... 35  
   5.1 Contextual Knowledge Bases ....................... 35  
   5.2 Predictive State Representations .................. 37  
   5.3 Policy Learning ................................... 38  

6 Conclusions ............................................. 39
List of Figures

4.1 Parsing Pipeline ........................................ 14
4.2 Learning Regime ........................................ 15
4.3 Context-Tree Phrase Structure Tree .................. 15
4.4 Sequences of labels given to parser. Black bars indicate full logical forms, and whitespace represents binary feedback ........................................ 21
4.5 Overview of all relations occurring in the Penn Discourse Treebank .............. 23
4.6 An example of the latent variable discourse parsing model taken from the Penn Discourse Treebank Dataset. The relation here is an example of an implicit cause attribution relation. ........................................ 24
4.7 An example element of $\mathcal{G}$ with interval agreement ....................... 30
4.8 Speaker identification smoothing over three speakers. Clockwise from top left: un-smoothed semi-supervised without constraints (69.7), un-smoothed constrained semi-supervised (77.0), fixed constraint smoothing (91.0), learned constraint smoothing (94.4) ........................................ 31
5.1 The Average Action Cost of a CRF with Contextual Knowledge Base input ... 37
5.2 Example of Discourse Relations in Tutoring .......................... 38
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Integer Linear Program Formulation</td>
<td>18</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of different training methods for parser</td>
<td>21</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparison of accuracy compared to other work: CGCR10 [11], ZC07 [75], and KZGS11 [40]</td>
<td>21</td>
</tr>
<tr>
<td>4.4</td>
<td>Empirical Results for Implicit Relation Labeling</td>
<td>27</td>
</tr>
<tr>
<td>4.5</td>
<td>Voice activity detection features</td>
<td>29</td>
</tr>
<tr>
<td>4.6</td>
<td>Voice activity detection accuracy</td>
<td>29</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Natural language understanding in computing systems has shown great forward strides in recent years. Projects such as IBM’s Watson have demonstrated near-human levels of precision at language understanding tasks, while systems like Apple’s Siri have shown that these systems can be scaled to interact with millions of end users. However, while precise solutions have been discovered for language comprehension tasks, conversational agents continue to suffer from poor dialogue skills and social intelligence. Without greater understanding of contextual information and inter-sentential dialogue structure, intelligent agents will continue to struggle with the most basic conversational skills such as establishing mutual understanding. According to common ground theory [30], when an agent and a user are trying to achieve a common goal, such as conveying a complex concept from one party to the other, the system should strive to minimize the collaborative effort. This requires the conversational agent to accurately model the user’s knowledge and relevant contextual factors, and use that information when planning for an intended dialogue outcome.

Broadly speaking, we have identified two tasks that are central to improving the competence of conversational agents. The first task entails the discovery of relevant contextual factors. To establish linguistic context, we utilize traditional single sentence syntactic and semantic parsers. Additionally, dialogue planning requires knowledge of broader conversational structure. To address this, we propose a novel spectral discourse parsing algorithm. Discourse parsing models language structure by learning inter-sentential relations such as causality and contrast relations. Our algorithm uses spectral matrix decomposition with Markovian latent variables to learn a highly expressive model without suffering from local optima. We have also previously worked to reduce the need for annotated training data in semantic parsing using active, semi-supervised learning. Going forward, we plan to apply similar techniques to reduce the need for supervision while training a discourse parser, with the eventual goal of creating an algorithm to learn fully latent language structure without the need for annotated training data. We have also previously investigated non-linguistic contextual information, such as location and audio signals, and we believe these data modalities could be useful for a sensor equipped conversational agent. For example, the system may behave different in a private or public space, or it may respond to the physical engagement of the user. In many domains, a user’s personality, knowledge level, and preferences for certain types of interactions will also be critical contextual information required to carry on a successful dialogue. These personal factors are best discovered through repeated
interactions with a given user.

The second task is dialogue planning using contextual and personal information. We propose the use of *Contextual Knowledge Bases* (CKBs), which use logical relations to represent contextual information from multiple linguistic and non-linguistic data sources. While CKBs are very expressive, planning using a CKB is difficult due to the extremely high-dimensional and sparse state space. To address these issues, we identify several useful techniques related to spectral dynamical systems. In particular, *Predictive State Representations* (PSRs) [65] have been shown to be a powerful tool for dealing with large, unwieldy state spaces. By formulating the state space as a combination of known basis functions (known as tests and histories), we are able to formulate a tractable state representation. PSRs and their generalized forms (such as Transformed Predictive State Representations) have been successfully combined with several planning and dynamical systems algorithms, even when dealing with sparse high-dimensional state spaces [6].

Our work is motivated by the education domain, and we intend for the methods described in this thesis to be potential components of a conversational intelligent tutoring system. By incorporating information about the student, the environment, and the state of the dialogue into a planning system, we hope to reduce the collaborative effort required to teach a new concept, and improve the overall effectiveness of conversational tutoring systems.

### 1.1 Educational Tutoring Application

Nearly every measure of quality of life is positively correlated with education [60]. The well educated enjoy more economic resources, better access to healthcare, and less emotional and physical distress. However, reports from the U.S. Department of Education Office of Civil Rights indicate a large disparity of access to educational opportunity across America.¹ Students in poor and rural areas do not have access to the same advanced classes or college preparatory programs as students in wealthy regions.

Furthermore, it has been known for many years that personalized, one-on-one education is much more effective than a one-size-fits-all approach [3]. Many students suffer when a curriculum caters to the ‘average’ student. The more gifted students become disinterested and do not fulfill their potential, while students with cognitive disabilities or inadequate background knowledge will quickly fall behind the rest of the class. This problem is further exacerbated in massive online courses with thousands of students that may range in education level from graduate students to those without high school diplomas.

Intelligent Tutoring Systems (ITSs) have the potential to combine the effectiveness of personalized education with the scalability of online courses. By algorithmically modeling a student’s grasp of basic skills and his/her preferences for different learning styles, an ITS can provide individualized tutoring that has been shown to be much more effective than learning in a traditional classroom [29].

To date, though, most Intelligent Tutoring Systems rely on structured input from students (typically multiple choice or fill-in-the-blank). In contrast, much of effective human-based education involves a conversation between student and teacher, where the discourse enables the

¹[http://www2.ed.gov/about/offices/list/ocr/reports-resources.html](http://www2.ed.gov/about/offices/list/ocr/reports-resources.html)
student to ask questions and the teacher to better understand the student’s thought processes by observing his/her reasoning structure. In addition, the teacher is able to tailor her/his feedback to the student, based on the cues provided by the students questions and detailed responses to exercises. Modern conversational intelligent systems have become quite adept at question answering tasks, but are much less effective when faced with complex, structured dialogue and debate. Emulating conversational social intelligence is a difficult problem that requires the system to demonstrate extensive natural language capabilities, as well as domain knowledge and contextual awareness. In particular, when trying to coach a user to improve his or her deliberative skills, the system must be able to assess those skills and initiate interactions to test and improve the user’s capabilities.
Chapter 2

Related Work

Interest in Intelligent Tutoring Systems has been growing in recent years. One of the most popular is the Carnegie Learning Cognitive Tutor [57]. This tutor is used by thousands of students around the country, and most students demonstrate an improvement of approximately one standard deviation (1 sigma) compared to students taught without the use of an ITS. Intelligent Tutoring Systems like this one conduct a process of knowledge tracing, wherein a student’s aptitude is tracked according to a set of atomic knowledge components that are required for solving problems. As students answer evaluation questions, the model of the student is updated accordingly. While existing ITSs, such as the Cognitive Tutor, show a great deal of promise, they nearly always rely on highly structured interactions with students and have limited or non-existent natural language understanding.

There are also a handful of conversational ITSs that have some natural language functionality, such as AutoTutor [29], Why2 [71], and DeepTutor [61]. The learning improvements gained with these tutors have been reported to be as high as 1.64 sigma, indicating a significant improvement over ITSs that use only structured input [71]. However, many of these systems rely on myopic semantic parsers that ignore discourse structure [29]. DeepTutor is an example of an emerging conversational ITS that identifies discourse structure, but it does so using inefficient branch and bound optimization, and the discourse structure is not used when assessing student reasoning—student answers are simply marked as correct or incorrect according to the semantic similarity distance from a canonical correct answer.

Automated essay grading software has been available for some time, and takes advantage of natural language features of varying levels of sophistication—early examples used very simple features like length and size of vocabulary, but recent examples can take advantage of more-sophisticated text representations [47]. We emphasize that we are not proposing to do automated essay grading. Instead we are looking for more-subtle, more-structured output from our learners: discourse structure instead of simply “B+.”

The fields of semantics and syntactic understanding in natural language have enjoyed widespread attention from researchers in many different disciplines. In our work, we are particularly interested in efforts to reduce the need for annotated training data when building a textual parser. New approaches to parsing have tried to lessen the burden of labeling data by using Markov Logic Networks [51], support vector machines [36], or binary feedback mechanisms [11]. In general, the semi-supervised parsers have lower classification accuracy than their fully super-
vised cousins when given a fixed amount of data. Semi-supervised approaches have been shown to be especially effective when large amounts of unlabeled data are available [39].

While active learning is a very mature and well studied topic [63], it has been applied to text parsing only in limited ways [68, 69]. Because the cost of annotation is high when building semantic parsers, it is thought that active learning can help to make the most of a limited amount of labeled data, but further study in this field is required [64].

There has been quite a bit of work concerning the use of spectral methods for supervised models [5, 35] and latent variable models with known structure [1, 66]. However, there has been much less work with regards to learning latent structure using spectral methods. One prominent example of spectral learning of latent structure was an algorithm for learning latent syntactic parse trees [34]. In this approach, the parameters of all possible latent topologies are expressed in a compounded mixing matrix, and spectral decomposition is then applied to this matrix to recover the parameters of the latent structures. While this approach demonstrates the viability of learning latent structure using spectral methods, it entails a spectral decomposition of a matrix proportional in size to the number of possible latent topologies—which becomes intractable for any non-trivial space of latent structures. More recently, researchers have demonstrated an unsupervised, spectral constituent parsing algorithm that utilizes observable additive tree metrics to estimate latent constituent parse topologies [50]. It is possible that similar techniques could be applied to perform other types of parsing tasks without supervision.

Existing algorithms for learning discourse structure rely on slow or inexact optimization procedures and do not model global dependencies in the discourse structure [72]. Many discourse parsing algorithms construct parse trees iteratively, by adding and labeling the edges of the parse tree [20] one at a time, while ignoring the broader characteristics of the parse tree. Latent variable, spectral methods have successfully been applied to dependency parsing [16, 45] and Probabilistic Context Free Grammars [12]. There has also been work training a latent variable model for dependency parsing using EM [49], but this approach was slightly less accurate and orders of magnitude slower than the spectral equivalent. Less attention has been paid to discourse parsing, with much of the recent work utilizing comparatively simple models such as linear support vector machines and maximum entropy classifiers [20, 43]. In contrast, we propose a much more sophisticated parsing model, with fast, statistically consistent optimization using spectral methods.

There has also been some effort to build context awareness in non-linguistic settings. One example we address is interruptibility, in which we wish to determine if a user would be amiable to interruption from their current activity given sensor signals, such as GPS, audio, and smartphone accelerometer data. Researchers have been studying interruptibility in the field of human-computer interaction for many years. This research was initially in the domain of desktop computers, when multi-tasking applications began introducing irritating interruptions to users while they worked. Early research focused on using only information about the state of the user’s software to determine interruptibility [62], but more recent work has instead been using sensors to perceive the user’s environment in order to achieve context-aware interruptibility [27, 33]. Some systems have introduced a decision-theoretic component that allows the system to ask questions only when the cost of interruption is low [59].

Recently there has been growing interest in natural language generation and dialogue planning using machine learning techniques. Given a reinforcement signal, dialogue planning can be
addressed using reinforcement learning techniques [56]. However, such a reinforcement signal is not always easily observable, as such, it may be preferable to use inverse reinforcement learning (also called imitation learning) to train a dialogue system using human demonstration [10]. There has been a fair amount of work in using linguistic context in dialogue systems [42], but much less work focuses on dialogue generation using environmental context, personal user information, or discourse relations. Some of the most successful efforts towards dialogue personalization have utilized techniques from common ground theory and social psychology, which has yielded models of evaluating speech style accommodation and predictive models for group dialogue settings [31, 48].
Chapter 3

Overview of Approach

Most modern dialogue systems operate using a question-response based approach, wherein a given linguistic input will always illicit a single response, or a response randomly chosen from a set. This approach is adequate for many question answering applications, in which an answer does not vary across contexts. However, linguistic and environmental context play a tremendous role in everyday human conversation. Likewise, a dialogue agent engaging in a more complex task, such as establishing shared understanding or accomplishing a cooperative task, must similarly react to changing conditions. As an example of this, consider the following excerpt of a conversation between a tutor and a student:

Student: “What is the Pythagorean theorem?”

Tutor: “The Pythagorean theorem states that the square of the hypotenuse of a right triangle is equal to the sum of the squares of the other two sides.”

Student: “I don’t understand, what does the Pythagorean theorem mean?”

A question-response driven agent would semantically parse the student’s second statement as a request for the definition of the pythagorean theorem and provide the same response that the tutor had just given. However, if we look at the student’s second utterance within the context of the conversation, we see that the standard definition of the theorem is not sufficient to convey this information to the student. The system should either try to convey the meaning of the theorem using a different approach, for instance using visual aids, or the system could try to determine if the student understands the concepts related to the theorem, such as the notion of a triangle’s hypotenuse. There are many other examples of important linguistic context, such as ambiguous intersentential subject references. For instance, if the student had said “what does the theorem mean” without specifying which theorem was being referenced, the tutor would need to consider the student’s previous statement to identify the ambiguous referent of this question. In these sorts of back and forth conversations traditional semantic parsing, which only gives us the semantic representation of a single sentence, is insufficient.

Another factor we would like our dialogue agent to respond to is environmental context. Establishing environmental context requires external sensors, such as microphones, cameras, GPS units, or 3D range finders. Much of the contextual information we may be interested in requires sensor processing algorithms, for instance if we wish to determine the posture of a student given an image and depth map of the environment. Within the education domain, an example of po-
tentially useful contextual data is the physical engagement of the student. If a student appears to be disengaged with the tutor, instead using a mobile device or reading a magazine, we may infer that the student may be bored with the material, confused, or dissatisfied with the system’s teaching method. Similarly, the agent’s dialogue when interacting with multiple users simultaneously should differ from instances in which there is only a single user. Simpler contextual information could also be valuable, for instance the time of day. A tutor may wish to use a different strategy at 3:00 in the morning, compared to 3:00 in the afternoon.

The third type of contextual information we consider is a student’s personal preferences and relative knowledge level. No two students are the same; users may have very different learning styles and aptitude for various tasks. Knowledge Tracing is the task of estimating a student’s grasp of atomic skills known as knowledge components. In mathematics, examples of knowledge components may include the ability to add or subtract two single digit numbers. A more complex task, such as algebraically determining the value of an unknown variable, will require the student to demonstrate several knowledge components to successfully solve the problem. Knowledge tracing is a well studied problem, and we can utilize the knowledge component estimates of a student as important contextual information in dialogue planning. Additionally, information discovered in a conversation with a student may help us refine the knowledge component estimates. For instance in the example dialogue given above, we can infer that the student is lacking aptitude for a knowledge component related to Pythagorean’s theorem.

While there are many methods for estimate a student’s level of knowledge, what is less well studied is the problem of tailoring a tutor’s behavior to a given student’s learning style. Some users may prefer visual teaching methods, while others may respond better to an interactive dialogue with the tutor. Just as an exam question consists of multiple required knowledge components, different teaching methods may also consist of several style components, such as visual, auditory, or interactive. A lecture delivered by instructor with slides, for instance, would contain the visual and auditory style components, but not the interactive component. As the tutor spends more time with an individual user, we can refine a model of the user’s learning preferences towards different style components by identifying the style components that lead to the largest gains in quantitative student performance. Similarly, we could model the student’s learning habits. If we see that a student is studying well into the night on a weekend, when the student normally only studies for an hour at a time on weekdays, we may infer that the student is cramming information for an upcoming exam, and a different approach to teaching may be required.

As we have seen, there are many contextual factors that we wish for our dialogue agents to response to, which requires a very expressive state space representation. We have previously explored the notion of Contextual Knowledge Bases (CKB), which represent contextual information in logical form using combinations items taken from an ontology including logical concepts, predicates, and relations [25]. The logical form will also be accompanied by a timestamp indicating when the information was added to the CKB. A logical form representation is convenient because it allows us to place the output of semantic or discourse parsing algorithms directly into the knowledge base. We see that with a large enough ontology, a state space represented as a contextual knowledge base is extremely expressive, allowing us to encode a potentially infinite number of logical forms. However, this expressiveness comes at a cost, as a CKB state space will be very sparse and high-dimensional. Fortunately, the state space is highly structured, and
it is likely that a relatively small set of contextual variables will be most informative during the
dialogue forming process. Semantic and lexical similarity measures also give us an intuitive
measure of distance between CKB entries, which could allow us to leverage logical forms that
we have never seen before.

One promising approach for dealing with the sparse, high-dimensional state space is the use
of Predictive State Representations (PSRs). Rather than using the full, uncompressed observation
history, Predictive State Representations represent state using a set of observation predictions
known as tests. It can be shown that for any equivalent Partially Observable Markov Decision
Process (POMDP) there is a basis set of tests equal or smaller in size to the number of POMDP
hidden states, such that the dynamics of the system can be represented completely using the basis
of the PSR. In practice, these tests can be used for subspace identification, because we are able
to use spectral methods to identify a small set of tests that are sufficient to predict the outcome
of any sequence of future actions. This is particularly useful when combined with a large, sparse
semantic state space because we can potentially identify a small set of predictive contextual
features given a large CKB. A predictive state space can then be used in conjunction with many
dynamical system modeling and planning algorithms. Within the education tutoring application,
we believe that an imitation learning framework trained using data taken from human tutors may
be effective.

A predictive state representation allows us to build our state space using a large number of
contextual features from a wide variety of modalities. Our linguistic features can be driven by our
previous work in semantic and discourse parsing. Semantic parsing is the process of converting
a natural language utterance into a semantically equivalent representation in a machine readable
language, called a Meaning Representation Language (MRL). This output generally consists of
logical forms generated from an ontology, similar to the entries in our contextual knowledge
base. Our semantic parsing framework uses syntactic and lexical features, trained using struc-
tured Support Vector Machines, and MRLs that are created using Integer Linear Programming.
Our framework is unique in our use of active, semi-supervised training, which dramatically re-
duces the need for annotated data, while maximizing the impact of a small number of annotated
examples.

A semantic parsing framework like this is essential for understanding the utterances of users,
but does not give us any sense or linguistic context necessary to improve dialogue planning.
To better understand the dialogue structure, we also incorporate features derived from discourse
parsing. Discourse parsing is the process of labeling inter sentential relations of text in a docu-
ment or utterances in a dialogue. Examples of discourse relations may include causal, rebuttal,
and paraphrase relations. Our discourse parsing technique leverages a latent variable model cou-
pled with spectral optimization to learn an expressive parsing model quickly and with statistical
consistency. Similar to our work with semantic parsing, we also plan to leverage unsupervised,
semi-supervised, and active learning to minimize the need for annotated data.

Finally, sensor processing algorithms will also be necessary to understand environmental
context. We have previously worked on decision theoretic algorithms for use with contextual
data. We have identified several algorithms for extracting useful information from audio, video,
and location data, and we also developed a constraint learning technique that uses structure learn-
ing in graphical models to enforce known structure topologies in sequences of labeled data. This
is particularly useful in smoothing sequences of labels to better fit the structures known to occur

in sensor processing tasks. For a dialogue agent, these techniques could be useful in discovering non-linguistic contextual cues that could potentially improve the effectiveness of the dialogue policy.

Going forward, we will broadly break the context-aware dialogue planning task into two subtasks: 1) discovering contextual information and 2) creating a dialogue policy based on a state space populated with contextual features. In chapter 4, we will discuss our previous and ongoing work in discovering both linguistic and non-linguistic contextual features, chiefly those derived from semantic parsing, discourse parsing, and sensor processing. In chapter 5, we will discuss a Contextual Knowledge Base representation of the state space, and dynamical systems trained with a Predictive State Representation.

We have previously developed several techniques for extracting contextual information. Our central contributions proposed for future work consist of minimizing the need for annotation in parsing tasks using active learning, semi-supervised learning, and latent variable spectral modeling. We also propose to develop a new planning framework for generating a policy in the sparse high dimensional space required to model the complexities of human social interaction.
Chapter 4

Extracting Contextual Features from Text and Sensor Data

In this chapter, we introduce methods for discovering environmental, linguistic, and personal contextual features. This information will be used to populate a contextual state space induced by a CKB, which will subsequently be used in the dialogue planning task. We begin by describing semantic features derived from a semi-supervised semantic parsing algorithm. We then describe our spectral, latent-variable discourse parsing algorithm, which can be used to uncover the relational structure of a dialogue. Finally we discuss some of our efforts to improve sensor processing algorithms used to identify environmental features and propose methods that may be used to generate a model of a specific user’s preferences and knowledge.

4.1 Semantic Features

One critical component in dialogue systems is semantic understanding of unstructured user input. Semantic parsing is the processing of natural language phrases, with the goal of producing a semantically equivalent representation in a machine-readable Meaning Representation Language (MRL). Interest in semantic parsing has grown significantly in the past few years, as state-of-the-art systems have begun to exceed human performance in certain non-trivial domains, such as Jeopardy! competition [21].

Although the accuracy of semantic parsers has risen dramatically, many existing parsers suffer from some key drawbacks. To begin with, the process of providing the algorithm with labeled training data is often costly and time-consuming. Many MRLs have a functional language structure, similar to Prolog. Annotating a sentence requires familiarity with the semantic ontology, and even for annotators familiar with the language, having a human parse a sentence to produce an MRL is slow and imprecise. Much work has been done recently to minimize the need for supervision in the training of semantic parsers [15, 36, 51].

Another disadvantage of most semantic parsers is that they do not leverage user interaction post-deployment to continue training. With a fully supervised parser, it would be unrealistic to imagine that a user would be willing or able to provide a correct MRL annotation of a sentence. However, as new methods of supervision have arisen in the literature, it is becoming possible
We begin with a framework proposed by Clarke, et al [11]. This approach has the distinct advantage of relying on binary feedback, rather than full supervision. When an annotator provides a sentence to the parser, the sentence is parsed, and the resulting MRL is fed into a response generator. The user then simply tells the system if this response was appropriate for the input or not. In this way, the MRL representation of the sentence is completely obfuscated from the annotator. This eliminates the need for the user to be familiar with the semantic language, and allows many sentence to be annotated much more quickly.

We seek to extend this approach by including bootstrapping and active learning in the framework. In this way, a corpus of unlabeled data can be used to improve the accuracy of the classifier without requiring annotation, and annotations will be requested only for the most informative sentences. Furthermore, the binary feedback mechanism could allow the system to continue to learn post deployment by evaluating user responses. For example, when a user says “thank you” after issuing a query to the semantic parser, we can infer that the query was parsed correctly and add it to the training corpus.

The following sections describe our framework used for the semantic parsing system [22]. We combine syntactic features given by a dependency tree [38] with lexical features [17]. Weights over these features provide a likelihood that a given MRL is correct. Previous work has shown that Integer Linear Programming (ILP), while NP-Hard in general, is very effective when used for the sparse problem of producing a maximum-weighted MRL given a set of features and weights [53]. This framework for parsing a given sentence was proposed in [11], and a graphical representation of the parsing pipeline is shown in figure 4.1.

The MRL that is output by the parser after the pipeline is executed depends on the vector of feature weights, $W$. Previous work has shown that these weights can be learned using only binary user feedback (i.e. the MRL produces a correct or an incorrect response). In our work, we extend this to include several new modalities of training. The training steps are shown in figure 4.2. The process begins by providing the parser with a small set of $\{\text{Sentence}, \text{MRL}\}$ pairs with which a working baseline parser is constructed. At this point, queries are made to the human annotator according to the active learning framework described in section 4.1.4. After a label is received, and the weight vector is updated, the sentences in the unlabeled dataset are parsed, and
the algorithm’s confidence in these parsings is recorded. Any unlabeled sentences with sufficient confidence are given back to the parser with the predicted MRL used as a ground truth label.

4.1.1 Syntactic Features

The Stanford Parser Probabilistic Context-Free Grammar (PCFG) suite is used to extract syntactic information from the input sentence[38]. Some of this syntactic information is computing the lexical features of the sentence, described in section 4.1.2, while some information from the syntactic information is used for separate syntactic features. Figure 4.3 shows a phrase structure tree generated by the syntactic parser when applied to the sentence ”How many people live in New York City?” This tree contains part of speech tags on each leaf of the tree, and also whole noun phrases, such as “New York City”.

In addition to the phrase tree, a typed dependency tree of the sentence is also computed. This tree is used to predict the semantic relatedness of words in the sentence. For example, in the sentence “the man with the dog likes to run”, we would like to know if the predicate $\text{likes\_to\_run()}$ applies to the man or the dog. To help understand the relatedness of two phrases in a sentence, we compute the normalized distance in the dependency tree between the head words of these two phrases. In this way, we could infer that $\text{likes\_to\_run}(\text{man})$ is a more probable MRL than $\text{likes\_to\_run}(\text{dog})$, even though they are both valid semantic representations.
4.1.2 Lexical Features

Lexical similarity can be a valuable tool for mapping words in an input sentence to ontology items. For instance, if our ontology contains a concept *dog*, the concept could be instantiated in a sentence with many different words, such as “pooch”, “hound”, “mutt”, or “puppy”. Lexical similarity features give us a tool for mapping diverse sentence constituents to known concepts, predicates, and relations in the ontology.

Our system uses WordNet[67] to compute the lexical similarity of two natural language phrases. In WordNet, words are organized into synonym sets (synsets), for example the words “humans”, “mankind”, and “humanity” would share a common synset. The synsets are then organized into a tree, with parenthood representing hypernym-hyponym relationships. As such, the “humans” synset will be a descendent of the “mammals” synset, but not the “invertebrate” synset. Nouns, verbs, and adjectives all lie in separate hierarchies.

To compute the lexical similarity of two words using the WordNet, we utilize the distance of these two words to their least common subsumer (lcs) in the WordNet tree[38]. All words in a given hierarchy descend from some common general concept. We denote \( l_1 \) the distance to the lcs from word 1, similarly for \( l_2 \). We further denote the depth of the least common subsumer as \( d(lcs) \). The WordNet similarity is described by the following function.

\[
WN(w_1, w_2) = \begin{cases} 
0.3^{l_1+l_2} & \text{if } l_1 + l_2 \leq 3 \\
0.3^3 & \text{if } l_1 + l_2 \leq 3 \cdot d(lcs) \
0 & \text{Otherwise}
\end{cases}
\]

We also use some syntactic information to inform our computation of lexical similarity. We first use the phrase structure tree to create a set of *constituents* from the input sentence, which includes both the literal words in the sentence (e.g. people) as well as chunked noun phrases (e.g. new_york_city). At this stage we also remove determiners, conjunctions, and the like from the set of constituents. Furthermore, we use part of speech tags to map the word to the correct synset tree. For instance, only when used as a verb will the constituent “dog” have lexical similarity with the concept “harass”.

4.1.3 Mapping Sentences to MRLs with Integer Linear Programming

Given a sentence, a set of weights, \( w \), and feature functions \( \Phi \), we wish to find the MRL representation that best represents the semantics of the input. To accomplish this, we represent the mapping of the sentence into an MRL with binary variables and use an Integer Linear Program to assign values to these variables.

We denote our ontology as \( D \) and our input as \( X \). \( X \) contains the constituents of the sentence. *First-order* variables indicate if text phrase \( c \) is mapped to ontological item \( s \)—we denote this variable \( \alpha_{c,s} \). Items in the ontology may represent concepts (such as New York City) or functions (such as \( \text{population}[x] \)). Every concept has a type (city, river, etc), and every function has a list of valid arguments (\( \text{population}[x] \) requires that \( x \) is of type city or state). To represent a valid MRL, we must consider how these ontological items are composed functionally. To represent this, we use *second-order* variables. For two items from the input, \( c, d \in X \), and two items from
the ontology, $s, t \in D$ the variable $\beta_{cs,dt}$ represents that these items are composed in the form $s(t)$.

The formulation of the integer linear program is given in Table 4.1. We wish to select the MRL that maximizes the linear weights multiplied over the lexical features $\Phi_1$ and the syntactic features $\Phi_2$. We use several ILP constraints to ensure that the values assigned to these variables after optimization corresponds to a valid MRL. The first constraint restrains all $\alpha$ and $\beta$ variables to be binary, because those variables denote which ontological items are present and how they are composed. The second constraint requires that every constituent from the input text corresponds to exactly one item in the ontology (which may be the \textit{NULL} concept). The third constraint requires that second-order variable $\beta_{cs,dt}$ can only be active if first-order variables $\alpha_{cs}$ and $\alpha_{dt}$ are active. Because the $\beta$ coefficients are included in the objective function, any time this constraint is satisfied, we will have $\beta_{cs,dt} = 1$ in an optimal solution.

Integer Linear Programming is NP-Hard, and the problem as we have formulated it will produce an exact solution. If an input sentence has $|X| = n$ constituents, and the ontology is of size $|D| = m$, there are $O(mn)$ optimization variables for the ILP to consider. This fast becomes intractable, even with a modestly sized ontology. Fortunately, the number of variables that could possibly be active is much lower. We leverage two properties of the problem structure to regain tractability while maintaining the optimality of our final solution. These two properties are lexical feature sparseness and typed dependencies.

We first use lexical feature sparseness to reduce the number of first-order features, which in turn reduces the number of second-order features. The average synonym set in WordNet contains 7.82 entries\cite{73}. Two words will usually have non-zero similarity according to the WordNet metric only if the larger distance to the least common subsumer is no more than three. This means that most words will have non-zero lexical similarity with only around 100 words, out of the 250,000 common words in the English dictionary. By automatically setting $\alpha_{cs} = 0$ when the lexical similarity between $c$ and $s$ is 0, we can reduce the number of first order variables considerably. We also need to consider a second-order variable only if both of its first-order variables could be active, thus the decrease in second order variables is even more dramatic.

Secondly, we can use the typing of items in the ontology to restrict the number of second-order variables. Second order variable $\beta_{cs,dt}$ can be active only if the composition $s(t)$ is valid, meaning that $s$ is a function and $t$ has a valid-type to be an argument to $s$. We explicitly set $\beta_{cs,dt} = 0$ whenever these requirements are not met.

Using these two methods for reducing the dimensionality of the optimization problem, we are typically left with an ILP over 200-400 variables, rather than the one million variables we would need if no pre-processing was done. This makes the exact inference tractable to run in an interactive system, where promptness is essential.

\section*{4.1.4 Active Learning & Bootstrapping}

The need for annotated data is a serious bottleneck in parsing applications, particularly when porting a parser to a new textual domain. However, unlabeled text data is abundant for most domains, and this data should be leveraged during the training process. Fully unsupervised methods are effective in domains with a vast amount of unlabeled data, but they are inefficient during the early phases of training because most unlabeled inputs will contain little useful information. In-
Maximize:
\[ \sum_{c \in X} \sum_{s \in D} \alpha_{c,s} \cdot w^T \Phi_1(X, c, s) \]
\[ + \sum_{c,d \in X} \sum_{s,t \in D} \beta_{c,s,d,t} \cdot w^T \Phi_2(X, c, d, s, t) \]

**Objective Function**

Subject to:
\[ \forall (c, s) \quad \alpha_{c,s} \in \{0, 1\} \]
\[ \forall (c, d, s, t) \quad \beta_{c,s,d,t} \in \{0, 1\} \]
(All variables are binary)

**Constraint I**
\[ \forall c \quad \sum_{s \in D} \alpha_{c,s} = 1 \]
(Every constituent mapped to exactly one item in Ontology)

**Constraint II**
\[ \forall (c, d, s, t) \quad \frac{\alpha_{c,s}}{2} + \frac{\alpha_{d,t}}{2} \geq \beta_{c,s,d,t} \]
(An ontological combination can only be active if both its constituents are active.)

**Constraint III**

Table 4.1: Integer Linear Program Formulation
Instead, we propose strategic use of a human annotator, particularly early in training, which will maximize the impact of the unlabeled data later on.

In this section, we present three candidate metrics for selecting a sentence to be approved by a human annotator. Given a corpus of unlabeled sentences, these metrics are computed for each sentence. An MRL is constructed for the sentence that is selected, and a system response for the MRL is generated. At this point, the original sentence and the response are given to the user, who indicates to the system whether this pair is correct. An example pair that could be given to the user is <“What is the capital of New York?”’, “Albany”>.

**Maximum weighted sampling**

$$w(X) = \sum_{\{\alpha_{c,s}=1\}} w^T \Phi_1(X, c, s) + \sum_{\{\beta_{c,s,d,t}=1\}} w^T \Phi_2(X, c, d, s, t) \sum w_i$$

With this metric, we are leveraging the notion that samples that are labeled as correct by the user are much more informative than samples that are labeled as incorrect. This is due to the fact that a correct parsing implicitly gives the algorithm a good MRL for the sentence, whereas an incorrect parsing only states that the given MRL was incorrect. In the equation above \(\{\alpha_{c,s} = 1\}\) denotes the set of first-order variables that are active after the ILP has been run. \(\{\beta_{c,s,d,t} = 1\}\) is defined similarly.

This approach to active learning will supply the user with sentences that we already believe are correctly parsed. While this will give the algorithm more “correct” labels, which are more informative than “incorrect” labels, we will not be improving the performance of the parser on labels that are likely incorrect to begin with.

**Density sampling**

$$density(X) = \sum_{c \in X} \sum_{Y_i} \frac{1}{|Y_i|} \sum_{d \in Y_i} WN(c, d)$$

With density sampling, we wish to select sentences that are representative of many other sentences in the unlabeled dataset. For given sentence X, we compare the similarity of X to all other sentences in the corpus, Y. We use normalized lexical similarity between words to compute the similarity of these sentences. We assume that X and Y_i have already been syntactically parsed, the constituents have been chunked (for instance, “The United States of America” is one chunked item), and determiners and conjunctions have been removed.

**Weighted density sampling**

$$WD(X) = w(X) \cdot density(X)$$

For our final metric, we combine the two approaches described above. This will result in an active learning scheme that will sample from the denser regions of the unlabeled dataset, while preferring parsings that we believe are correct.
Bootstrapping

To conduct bootstrapping between active learning queries, we leverage the objective function described in the Integer Linear Program, which we denote $w(X)$. We will use this objective function as a proxy for parsing confidence. Using a validation set of unlabeled sentences, we compute $\hat{z}_{95}$, the 95th percentile of weights for the current model. Any sentences that have weight larger than this threshold are labeled as positive and added to the training set.

$$\hat{z}_{95} \leq w(X)$$

4.2 Semantic Parsing Results

GeoQuery is a database containing facts about American geography, and it has been used to evaluate many semantic parsers [74]. The data contains natural language sentences (e.g. “What is the population of California?”) with equivalent MRL representations (e.g. `query(population(state('California')))`). The data contains 1,130 such pairs, broken into one set of size 880 and one set of size 250. In our work, we set aside 150 samples from the larger set for the purpose of parameter validation, and the smaller dataset was used as a testing set.

The ontology present in the GeoQuery dataset includes many functions, such as `population(city)`, `elevation(state)`, and `length(river)`, as well as concepts for all the major US cities, rivers, mountains, and states. When the items of this ontology are combined, it allows users to ask questions such as “what is the largest state that the Mississippi river runs through?”.

We conducted experiments to examine three methods of selecting a sequence of training samples to provide to the semantic parser. In all three experiments, the parser was allowed to select 250 sentences out of the 730 possible training sentences.

The first method, random sampling, randomly selects an unlabeled sentence with uniform probability. The sentence is given to the algorithm with a complete MRL with probability $\frac{1}{\delta}$, where $\delta$ is the cost of a full annotation (20 in our experiments). With probability $1 - \frac{1}{\delta}$ the parser receives only binary feedback for the sentence.

In the second method, the algorithm used the active learning metrics described in section 3.4 to iteratively select which sentence and type of feedback it would be given. In this experiment, the algorithm was allowed to select 250 training samples without bootstrapping.

Finally, the algorithm was run using both bootstrapping and active learning. In this case, any sentence in the unlabeled corpus with sufficiently high confidence was treated as positive binary feedback training samples using the current predicted logical form.

Because this semantic parser is capable of producing a logical form for an given input sentence, parsing accuracy was used to evaluate performance rather than precision and recall.

The results of the GeoQuery evaluation of the semantic parser are shown in Table 4.2. The random sampling method selected only 14 sentences to fully label. With full supervision, 250 labeled examples were randomly selected from the full set of 730. The inclusion of bootstrapping and active learning allows the learner to select the most informative labeled examples from the full set of 730, and use the rest of the examples as unlabeled data. Therefore we see an improvement in the performance of the parser compared with simply using random sampling to select data to be labeled. Figure 4.4 shows the sequences of labels that were chosen by the three
training methods. In this figure, the black bars indicate training samples in which full logical forms were chosen. We see in Figures 4.4(b) and 4.4(c) that the algorithm was more likely to ask for full logical forms towards the beginning of the training sequence. This is due to the fact that the model had higher uncertainty before seeing significant training data, and thus the binary feedback was likely to be less informative. In total, the active learning scheme asked for full MRLs in only 54 out of 250 instances. When bootstrapping is added, the number drops to only 38 fully labeled instances.

Table 4.3 shows the performance of our approach compared to the most successful algorithms that have been demonstrated on the GeoQuery dataset. In particular, we see that only 38 carefully chosen logical forms yields a 12 percentage point increase in accuracy compared to using only binary feedback as in Clarke [11]. We also see that our approach is quite competitive with fully supervised, CCG based parsers, despite only being given a fraction of the logical forms.

Table 4.3: Comparison of accuracy compared to other work: CGCR10 [11], ZC07 [75], and KZGS11 [40]
4.3 Discourse Features

In addition to sentence level parsing, we believe that dialogue agents must have a greater understanding of global dialogue structure. Discourse parsing is a fundamental task in natural language processing that entails the discovery of the latent relational structure in a multi-sentence piece of text. Unlike semantic and syntactic parsing, which are used for single sentence parsing, discourse parsing is used to discover inter-sentential relations in longer pieces of text. This is especially important in dialogue systems in which inter-sentential references are critical components in the conversational structure.

It has recently been shown that using latent variables to refine the category labels in a labeled syntactic parse tree can lead to higher final parsing accuracy—even if we just measure accuracy on the original, unrefined category labels [46]. For example, if the labels just distinguish the general syntactic category “noun phrase,” we might see that the latent-variable learner decides to distinguish more-specific categories like mass vs. count nouns; doing so would improve accuracy elsewhere in the tree by allowing the parser to distinguish between the ways these two types of noun phrases are used. Based on this result, interest in latent variable parsing models has grown recently [12, 70].

Once we introduce latent variables in this way, spectral methods become highly attractive: not only do they give us the opportunity to improve the accuracy and running time of a discourse parser that uses labeled examples, but they can leverage latent variables without the performance penalty associated with local optima. For that reason, a spectral Probabilistic Context Free Grammar (PCFG) parser that incorporates a latent state with each non-terminal tree node has been shown to achieve state-of-the-art performance in syntactic and constituent parsing, both in accuracy and in computation speed [12, 13].

In more detail, a parse tree with supervised labels but without hidden state values is denoted a skeletal tree, while a parse tree with accompanying hidden states is called a full tree. A parsing task requires the construction of only a skeletal tree, but modeling a full tree and computing the probability of the skeletal tree by marginalizing out the hidden variables yields a more accurate likelihood estimate for the skeletal tree. Finding the highest scoring tree in this framework would require computing marginal probabilities $p(i_m, j, k)$ for relation $i$ with hidden state $m$ and discourse elements $j$ and $k$. Given these marginals, the most likely tree can then be discovered using dynamic programming [28]. These marginals can be computed based on a labeled dataset using dynamic programming or expectation maximization, but recent work has resulted in a tensor formulation which allows the problem to be solved quickly and easily using spectral decomposition [12, 35].

This section presents our spectral, latent variable model [26] for the sequential relation labeling task for discourse parsing in the Penn Discourse Treebank (PDTB) [52]. The methods shown in this section can also be extended to operate on more complex discourse structures, such as those seen in Rhetorical Structure Theory (RST) [9]. The proposed spectral framework has many theoretically desirable properties, and we also demonstrate the practical advantages of the model with an empirical evaluation on the PDTB dataset, which reveals an $F_1$ score of 0.46. This accuracy shows a 3-6 percentage point improvement over approaches that do not model the joint structure dependence, and a 26 percentage point improvement over the naive baseline model that always selects the relation label with the highest occurrence in the dataset.
The total of Explicit, Implicit and AltLex tokens is shown of the specified "CLASS" tag and all its types and subtypes. Each "CLASS" count includes all the annotations in the corpus. Table 4 shows the distribution of "CLASS" level tags in the corpus. Disagreement at lower levels was resolved by proposing different subtypes, e.g., "expectation" vs. "contrast". Cases when one annotator picked a class level tag, e.g., "COMPARISON", and the other picked a type level tag of the same class, e.g., "Contrast", did not count as disagreement. At the subtype level, disagreement was noted when the two annotators picked different subtypes, e.g., "expectation" vs. "contradiction".

Class level disagreement was adjudicated by a team of three experts. Disagreement was resolved by applying the higher level tag "Concession" and the other, with the type "Contrast", to determine with confidence whether the relation and its arguments were attributed to the writer (e.g., attribution to the writer in 2965 tokens). The connectives that may or may not coincide with sentence boundaries.

Figure 1 shows the hierarchy of sense tags. The connectives, which typically relate non-simultaneous situations, are ambiguous between the "Contrast" and "Concession" types and their subtypes of "COMPARISON" but rarely between different classes. The connectives, however, are ambiguous between "although", "since", "meanwhile", and "while". The connectives, e.g., "after", "before", "e.g.", "i.e.", "thus", "therefore", and "thereby", which may or may not coincide with sentence boundaries.

Recent work (Wiebe et al., 2005; Prasad et al., 2005) has shown the importance of attributing beliefs and assertions to the writer(s) holding or making them. From latent variable models. We will use the ground truth parse structures provided by the PDTB dataset, so as to isolate the error introduced by relation labeling in our results, but in practice a greedy structure learning algorithm can be used if the parse structures are not known a priori. Previous work that learned the skeletal structure using a maximum entropy classifier indicates a roughly 10 percentage point decrease in the $F_1$ score of relation labeling with error propagation compared to using the ground truth skeletal structure [44].

Figure 4.5: Overview of all relations occurring in the Penn Discourse Treebank

4.3.1 Problem Definition and Dataset

In this section, we define the discourse parsing problem, and discuss the characteristics of the Penn Discourse Parsing Treebank (PDTB), which consists of annotated articles from the Wall Street Journal, and is used in our empirical evaluations.

Discourse parsing can be reduced to three separate tasks. First, the text is decomposed into elementary discourse units, or EDUs, which may or may not coincide with sentence boundaries. The EDUs are often independent clauses that may be connected with conjunctions. After the text has been partitioned into EDUs, the discourse structure is identified. This requires us to identify all pairs of EDUs that will be connected with some discourse relation. These relational links induce the skeletal structure of the discourse parse tree. Finally, each connection identified in the previous step is labeled using a known set of relations. Examples of these discourse relations include concession, causal, and instantiation relations. In the PDTB dataset, only adjacent discourse units are connected with a discourse relation, so with this dataset we are considering parse sequences rather than parse trees. The model we propose for this dataset could be easily modified to produce full parse trees, such as those we would see in a dataset based on RST.

Here we focus on the relation labeling task, as fairly simple methods perform quite well at the other two tasks [72], though initial results indicate that these other two tasks may also benefit from latent variable models. We will use the ground truth parse structures provided by the PDTB dataset, so as to isolate the error introduced by relation labeling in our results, but in practice a greedy structure learning algorithm can be used if the parse structures are not known a priori.
Some of the relations in the dataset are induced by specific connective words in the text. For example, a contrast relation may be explicitly revealed by the conjunction but. Sentence (1) demonstrates an explicit relation with two EDUs, connected by a coordinating connective.

(1) “The rapid expansion of the Bitcoin market initially led to huge gains, but recent electronic attacks of prolific banking services have resulted in a catastrophic free-fall of the digital currency’s value”.

Simple classifiers using only the text of the discourse connective with POS tags can find explicit relations with high accuracy [44]. For comparison, sentence (2) shows an example of the more difficult implicit relation. In this sentence, two EDUs are connected with an explanatory relation, shown in bold, although the connective word does not occur in the text.

(2) “But a few funds have taken other defensive steps. Some have raised their cash positions to record levels. [BECAUSE] High cash positions help buffer a fund when the market falls.”

We focus on the more difficult implicit relations that are not induced by coordinating connectives in the text. The implicit relations have been shown to require more sophisticated feature sets including syntactic and lexical information [43]. The PDTB dataset includes 16,053 examples of implicit relations.

A full list of the PDTB relations is shown in figure 4.5. In particular, we see that the relations are organized hierarchically into top level, types, and sub-types. Our experiments focus on learning only up to level 2, as the level 3 (sub-type) relations are too specific and have only 80% inter-annotator agreement [52].

### 4.3.2 Spectral Algorithm

Our discourse parsing framework utilizes Markovian latent states to compactly capture global information about a parse sequence, with one latent variable for each relation in the discourse parsing sequence. Most discourse parsing frameworks will label relations independently of the
rest of the accompanying parse sequence, but this model allows for information about the global 
structure of the discourse parse to be used when labeling a relation. A graphical representation 
of one link in the parsing model is shown in Figure 4.6.

Specifically, each potential relation \( r_{ij} \) between elementary discourse units \( e_i \) and \( e_j \) is 
accompanied by a corresponding latent variable as \( h_{ij} \). According to the model assumptions, the 
following equality holds:

\[
P(r_{ij} = r | r_{1,2}, r_{2,3}, \ldots r_{n,n+1}) = P(r_{ij} = r | h_{ij})
\]

To maintain notational consistency with other latent variable models, we will denote these 
relation variables as \( x_1 \ldots x_n \), keeping in mind that there is one possible relation for each adjacent 
pair of elementary discourse units.

Our specified model requires learning three parameters: the initial state distribution \( \pi \), the 
state transition matrix \( T \), and the observation matrix, \( O_{ij} \), for observation \( i \) and latent state \( j \). 
These parameter matrices define the model completely, but are difficult to learn directly. Instead, 
the spectral formulation of the model has us learning the \textit{observable operators} of the model in a 
reduced dimensionality subspace. If we define the matrix \( A_x \) as

\[
A_x = T \text{diag}(O_{x,1} \ldots O_{x,m})
\]

Then the following equality holds:

\[
Pr[x_1 \ldots x_t] = \hat{\pi}^T m A_x t \ldots A_x 1 \pi
\]

Using this formulation of the model, we need to learn the matrices \( A_x \) and \( T \), as well as the vector 
\( \hat{\pi} \). To compute these parameters, we use the unigram, bigram, and trigram probability matrices. 
We denote the unigram matrix as \( P_1 \), the bigram matrix as \( P_2,1 \), and the trigram matrix as \( P_{3,x,1} \), 
with one trigram matrix for each value of \( x \). Define these matrices as follows:

\[
[P_1]_i = Pr[x_1 = i]
\]

\[
[P_{2,1}]_{ij} = Pr[x_2 = i, x_1 = j]
\]

\[
[P_{3,x,1}]_{ij} = Pr[x_3 = i, x_2 = x, x_1 = j] \forall x
\]

Spectral latent variable models utilize subspace identification to learn the model dynamics 
in a reduced dimensionality space. If we assume that the dimensionality of this subspace is no 
less than the rank of the parameter matrices \( O \) and \( T \), then the model learned in the subspace is 
equivalent to the model from the original feature space. There are different methods of projecting 
the data into this subspace, but one convenient and computationally tractable approach is to take 
the left Singular Values of the bigram matrix \( P_{2,1} \). We denote this subspace transformation matrix 
as \( U \in \mathbb{R}^{n \times m} \). If we denote the subspace model parameters as \( \hat{\pi}_U \), \( \hat{T}_U \), and \( A_U \), these parameters 
can be easily learned using factorization over the fully observable matrices defined above. Proofs 
of all equalities given in this section are available in [35].

\[
\hat{\pi}_U = U^T P_1
\]

\[
\hat{T}_U = (P_{2,1}^T U) + P_1
\]
For our original feature space, we will use the rich lexical discourse parsing features defined in [20], which includes syntactic and linguistic features taken from dependency parsing, POS tagging, and semantic similarity measures (similar to those presented in section 4.1.2) that allow us to determine the lexical similarity of words in the input sentence to discourse relation labels. We also include contextual features, which denote the relations immediately to the left and to the right of the current relation, but these contextual features are less important in our model due to the inclusion of the latent variables.

### 4.3.3 Spectral Parsing Results

For our empirical results, we learned the parameters of the latent variable model using labeled training data from the PDTB. To evaluate our approach, we used the generative latent variable model to rerank a list of candidate discourse parses output from a discriminative discourse parser over the testing set. For our baseline discriminative parser, we used a linear support vector machine based on the framework presented in [20]. In this work, a one-versus-all linear classifier is trained for each possible relation. To generate a list of candidate parses, we stochastically label the relations of a discourse proportional to the distance from the decision boundary of the relation (rather than deterministically selecting the most likely relation as was done in [20]).

To score a candidate tree, we used the learned latent variable model to factor the conditional probabilities of the relation labels based on the parameterized latent state distribution. Specifically, the weight of a tree $Y$ consisting of relation labels $\hat{x}_1...\hat{x}_n$ is defined as:

$$W(Y) = \prod_{i=1}^{n} \left( \sum_{j=1}^{m} P(x_i = \hat{x}_i | h_i = j) P(h_i = j) \right)$$

In this equation, $P(h_i = j)$ represents the marginal probability of the hidden state after accounting for the transition probabilities from the adjacent nodes in the sequence, or the initial probability of the first relation in the sequence. We then use the weights of a set candidate trees, $Y_1...Y_N$, to label the relations of the discourse parse. Define the indicator function $Y_i(\hat{x}_j)$ to return 1 if relation $x_j$ has label $\hat{x}_j$ in tree $Y_i$, with this function returning 0 otherwise. To label a relation $x_j$, we select the relation $\hat{x}_j$ that maximizes the following quantity:

$$w(\hat{x}_j) = \sum_{i=1}^{N} W(Y_i)Y_i(\hat{x}_j)$$

The empirical results of our experiments are shown in Table 4.4. The quantities in this table reflect the precision, recall, and accuracy of the relation labeling task when using sections 01-21 of the PDTB dataset for training and section 23 for testing. The dimensionality, $m$, of the subspace is selected using section 22 as a parameter validation set.

The results from using the linear support vector machine [20] without latent variable reranking are shown, as are the reported results from recent work that utilized a maximum entropy classifier with the PDTB dataset. The comparison experiments were also conducted without error propagation from structure learning and without explicit relations in the testing set. The naive
classifier that always selects the most common relation label achieves an $F_1$ score of 0.20 under these conditions. When explicit, as well as implicit, relations are included in the testing set, the $F_1$ score of the latent variable model increases to 0.69.

### 4.4 Non-Linguistic Contextual Features

We can imagine many ways in which a conversational agent could use environmental information, for instance it could be used to understand ambiguous referents in text, predict the user’s desired behavior for the agent, estimate the user’s level of engagement, or determine the most appropriate method of engagement in the current situation, e.g. textual, visual, or auditory engagement. We have previously worked on building non-linguistic, context aware systems that employ decision making algorithms based on features derived from sensor data. In particular, the InContext smartphone app uses an iPhone’s sensors coupled with user data to predict if a user is likely to prefer to have the phone’s ringer on or off in a given context [23]. In this application, we achieved an average accuracy of roughly 95% across all users. We also investigated automatic detection of smartphone loss and theft using the entropy of the phone GPS signal compared to normal usage [76]. We subsequently extended upon these ideas to build a system that predicts if power wheelchair users would be willing to perform a pressure relief exercise in the current context given sensor data collected on the wheelchair [24]. Wheelchair sensor data included GPS, audio, accelerometer, light, temperature, and wheelchair joint/wheel encoders. For the wheelchair application, we leverage a spectral algorithm with Markovian latent states similar to that presented in section 4.3.2, which allowed us to conduct sequential prediction without the need to represent the full history of the state space.

In this section, we will present some relevant algorithms and techniques we used to extract contextual features from sensor data. In particular, we will present algorithms for classifying accelerometer and audio data, as well as a constraint learning algorithm that can be used to enforce a known structure on a sequence of sensor data labels. These algorithms were designed to be deployed using smartphone sensors. If our conversational agent is smartphone based, like Apple’s Siri, these algorithms could be deployed as is. Otherwise, the basic ideas could be carried over to a different sensor equipped platform.

#### 4.4.1 Posture Recognition

Previous work has shown that having knowledge of the user’s physical activities can be used to help establish social context [32], which may affect a user’s desired behavior for a conversational agent. However, accurately classifying a user’s activity generally requires one or more

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Recall</th>
<th>$F_1$ Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latent Variable Model</td>
<td>0.51</td>
<td>0.42</td>
<td>0.46</td>
</tr>
<tr>
<td>SVM [20]</td>
<td>0.48</td>
<td>0.39</td>
<td>0.43</td>
</tr>
<tr>
<td>Max Entropy [44]</td>
<td>0.40</td>
<td>0.40</td>
<td>0.40</td>
</tr>
</tbody>
</table>

Table 4.4: Empirical Results for Implicit Relation Labeling
accelerometers placed at known locations around a user’s body. With a mobile phone, a user may carry the phone in their pocket, purse, or on their belt, so we do not have a known reference point from which to conduct activity recognition. Instead, we simplify the problem to trying to estimate the current physical posture of the device itself, which can also give us some insight into the user’s current activity. In this task, we wish to determine if the phone is resting on a flat surface, if it is being actively held by the user, or if it is placed in a pocket, purse, backpack, etc. To address this problem, we collected labeled data from these three classes, using the 3-axis accelerometer and the proximity sensor of the phone. The data was divided into overlapping half-second frames, with the sample mean and variance of the accelerometer axes recorded for each frame. The number of times that the proximity sensor was triggered over the half-second was also recorded. A linear support vector machine was then trained to differentiate these classes, attaining 91.4% accuracy over 89 test samples.

4.4.2 Voice Activity Detection

Another potentially usefully piece of contextual information is the presence or absence of human voice activity in the environment other than the user’s, which could help us determine if the user is in a public or private space each of which entails different requirements for conversational interactions. Audio data was collected from the smartphone devices at a sample rate of 8192Hz. Ten seconds of audio was recorded, and this signal was broken into 20 half-second samples. For each of these samples, a Fast Fourier Transform is used to extract 16 features, presented in Table 4.5. We empirically compared multiple classifiers for use in the voice activity detection task. A support vector machine with a linear kernel and a Gaussian mixture model were both trained on labeled audio samples to differentiate audio samples containing human speech from samples that do not contain speech. Although previous work has shown this approach to be effective at the voice activity detection task [37, 55], there is one complication that arises in a mobile devices application: the device may be in a user’s pocket or handbag when the sample is collected, resulting in a significantly dampened signal and many false-negative predictions by the classifier. Because we are able to detect when the phone is in a pocket using the accelerometers and proximity sensor, we train a second speech detection classifier for this scenario. A linear support vector machine and Gaussian mixture model were also trained in this instance, with a new set of trained weights to account for the dampened signal.

The performance of the classifiers with the phone in and out of a pocket is shown in Table 4.6. The testing set included many noisy audio samples without voice activity, such as music and sounds of car traffic. Based on these results, the linear support vector machine appears to be the most accurate audio classifier.

4.4.3 Smooth Constraint Learning

Now we consider the problem of trying to smooth the sensor data predictions, to better fit the structure of the ground truth labels. This task can be thought of as a constraint learning problem, wherein we constrain data over an interval to share a label, and we must discover the endpoints of those constraint intervals. There are two ways that these learned smoothing constraints can improve performance: applying constraints during training can improve parameter estimates,
<table>
<thead>
<tr>
<th>#</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fourier mean</td>
<td>The sample mean of the magnitudes of all Fourier coefficients in the sample.</td>
</tr>
<tr>
<td>2</td>
<td>Fourier variance</td>
<td>The sample variance of the magnitudes of all the Fourier coefficients.</td>
</tr>
<tr>
<td>3</td>
<td>Total signal power</td>
<td>The sum of the squared magnitudes of all the Fourier coefficients.</td>
</tr>
<tr>
<td>4</td>
<td>Mid-range power</td>
<td>The sum of the squared magnitudes of the Fourier coefficients in the 250-600Hz range of the spectrum.</td>
</tr>
<tr>
<td>5</td>
<td>Ratio</td>
<td>The ratio of the mid-range power over the total signal power.</td>
</tr>
<tr>
<td>6</td>
<td>Zero crossings</td>
<td>The number of zero crossings in the Linear PCM encoding of the audio signal</td>
</tr>
<tr>
<td>7-16</td>
<td>Band power</td>
<td>9 features representing the signal power in 100Hz bands from 1 to 1000Hz. The bands are 1-100Hz, 101-200Hz...901-1000Hz.</td>
</tr>
</tbody>
</table>

Table 4.5: Voice activity detection features

<table>
<thead>
<tr>
<th></th>
<th>GMM</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>In pocket</td>
<td>86.7%</td>
<td>91.3%</td>
</tr>
<tr>
<td>Out of pocket</td>
<td>90.5%</td>
<td>95.1%</td>
</tr>
</tbody>
</table>

Table 4.6: Voice activity detection accuracy
and applying constraints to the predicted labels during classification can improve accuracy. This approach can be generalized to other constraint learning tasks, for instance if we are trying to label discourse relations in a discourse parsing task, we may believe that some relations obey a mutual exclusion constraint that could be discovered autonomously, e.g. an elementary unit of text cannot simultaneously participate in a rebuttal and an agreement relation.

We propose a restricted structure learning algorithm that learns constraints by searching over a class of possible constraints supplied by the user. The input to this algorithm is a class $G$ of graphical model structures that contains possible expressions of constraints; the algorithm searches only over graphical model structures within this class. Exactly which structures are members of $G$ depends on which constraints the user wishes to encode.

As an example of a class $G$, we may believe that our data is arranged in a sequence, with labels adopting the same value over intervals (such as $[11100001111]$). This situation arises naturally in problems with temporal structure, such as speaker identification. In this case, we would like to enforce an agreement constraint over intervals of the sequence. For this problem, we define the class $G$ such that (1) every label $Y^t$ in the sequence is associated with exactly one true interval, represented by a constraint variable $Z^i$, (2) all labels within a given interval have the same value, and (3) each interval must span a minimum number of label variables. In this class, every label $Y^t$ is the parent of exactly one constraint node, and each constraint node $Z^i$ has a continuous set of parent variables $Y^t, ..., Y^{t+k}$. An example element of $G$ is shown in Figure 4.7. As shown in Figure 4.8 this class $G$ increases prediction accuracy and quality of parameter estimation significantly when applied to a speaker identification dataset. Note that $G$ forces every label variable to participate in a constraint; without this requirement, we would trivially select a model with no constraint variables.

The object of structure learning is to select the element of $G$ that maximizes the data likelihood. Say the structure $g \in G$ contains constraint variables $Z_g$. We can reparameterize the discriminative objective to include the graph structure:

$$\ell_D(\theta, g) = \sum_{i=1}^{n} \log P(Z_g = 1, Y = y_i | X = x^i; \theta) + \sum_{i=n+1}^{n+m} \log P(Z_g = 1 | X = x^i; \theta)$$

Note that the objective is now a function of both the graph structure $g$ and the model parameters $\theta$. We propose to optimize this objective by alternately estimating each set of parameters. On the $t$th iteration, we compute:

$$\theta^{(t+1)} \leftarrow \arg \max_{\theta \in \Theta} \ell_D(\theta, g^{(t)})$$

$$g^{(t+1)} \leftarrow \arg \max_{g \in G} \ell_D(\theta^{(t+1)}, g)$$
Figure 4.8: Speaker identification smoothing over three speakers. Clockwise from top left: un-smoothed semi-supervised without constraints (69.7), un-smoothed constrained semi-supervised (77.0), fixed constraint smoothing (91.0), learned constraint smoothing (94.4)

The above optimization over $g$ can be performed efficiently for many classes $G$, including the class of interval agreement constraints, which occurs often when labeling sequential sensor data.

### 4.5 Proposed Work in Linguistic Feature Extraction

#### 4.5.1 Weakly-labeled and unlabeled discourse parsing

One key limiting factor in porting a discourse parser to a new domain is the current reliance on annotated data. Training an accurate discourse parser using only annotated data requires tens of thousands of labeled examples, which is not tractable if we need to retrain the parser to use it for each new topic the tutor instructs. As such, it is critical that we leverage the ample unlabeled text data available through online education programs, as well as online tutoring support forums which contain examples of multi-agent, conversational discourse.

Depending on the resolution of the labels in our skeletal discourse structure, the discourse annotation task can be more or less strongly labeled. At the coarsest resolution, only the structure is labeled, with no information beyond the existence of some discourse relationship that covers a particular range of the text. The importance of latent-variable learning, and therefore the importance of spectral methods, increases as the strength of the labeling decreases.

Going even further, we can imagine that parts of the discourse structure itself are elided: e.g., we label just the fact that some argument is being made in a particular portion of the text, without
specifying the type of argument or its internal structure. In the limit of this process, we get the fully-unlabeled discourse structure learning task; before we reach this limit, we say we are in the weakly-labeled case.

Our end goal, though, is to scale to a dataset in which many of the examples are completely unlabeled. In this direction, some preliminary progress has been made by other researchers: two spectral algorithms have been proposed for producing a generative model of latent tree topologies for dependency and constituency parsing [2, 34]. The core idea behind these approaches is the use of spectral decomposition to “unmix” a matrix of factored latent topology parameters in the case of syntactic parsing. Specifically, under each topology, the matrix of observed moments can be factored according to the latent parameters of that topology; then the topologies are combined by multiplying by a mixing matrix [34]. Unfortunately, the complexity of this spectral decomposition will be proportional to the number of possible latent topologies, which is super-exponential in the number of leaves. However, by using dynamic-programming-like techniques to try to unmix groups of related topologies at the same time, it can be reduced to a polynomial-sized (although still large) optimization [2].

One hurdle to fully unsupervised parsing is the absence of a clear observable distance metric between parse trees that would allow us to leverage existing work in unsupervised learning. Recently, additive tree metrics have been proposed as a distance formalism that would allow us to group similar sub-structures into clusters [18]. A distance metric over trees is considered an additive tree metric if the sum of distances of all subtrees between two trees is equal to the total distance between those trees. Some additive tree metrics, such as the empirical covariance of word counts in a block of text, are fully observable and do not require human annotation of latent structure. Additive tree metrics have recently been used to create an unsupervised, spectral constituency parsing algorithm[50]. By developing new additive tree metrics for the discourse parsing framework, we may be able to leverage unlabeled data much more extensively than was previously possible.

We can also simulate weakly-labeled data by erasing parts of the information in a more-strongly-labeled dataset such as the Penn Discourse Treebank. By doing so, we miss one of the advantages of learning from unlabeled data, namely the ability to take advantage of large, cheap unlabeled datasets; however, we gain the advantage of being able to test and develop algorithms on a range of levels of supervision. This advantage is important, since learning fully-latent dialog structure is very much an unsolved problem: we can test ideas that solve parts of this problem without needing to solve the whole problem at once.

There has also been some work on creating branch-and-bound-like techniques for dependency parsing [54] and discourse parsing [61]. If large sections of the topology space could be ruled out using these methods, the spectral decomposition could be made more viable.

4.5.2 New Discourse Parsing Frameworks

For the Penn Discourse Treebank Dataset, the discourse parses behave like sequence of random variables representing the relations, which allows us to use an HMM-like latent variable model based on the framework presented in [35]. While the Penn Discourse Treebank is one of the larger discourse parsing datasets currently available, there are many representational advantages to using a full tree structure, as in the Rhetorical Structure Theory framework. The sequential la-
tent variable model presented in this work should be extended to parse full trees. If the discourse parses were instead trees, such as those seen in Rhetorical Structure Theory (RST) datasets, we can modify the standard model to include separate parameters for left and right children, as demonstrated in [16].

We will also define dialogue relations specific to the education setting that are not seen in RST or the PDTB. For instance, one important relation may be a student expressing uncertainty with regards to a statement made by the tutor. Labeled datasets would not be available for any newly created discourse framework for education, which compounds the need for new unsupervised methods of parser training. Some labeled data would be required for evaluation purposes, but we will not have access to a Treebank sized labeled dataset with which we could conduct fully supervised training.

4.5.3 Dialogue Personalization

Another potentially crucial factor in dialogue planning are the personal factors that are unique to an individual user. In a tutoring application, for instance, different students will have different levels of knowledge and learning styles. The goal is to teach students new concepts using a teaching style that is effective for the individual, and only do so when the students has attained the necessary prerequisite knowledge. Quite a bit of work has gone into modeling a student’s grasp over atomic knowledge components given a qualitative evaluation of a students skill. One such knowledge tracing algorithm uses spectral methods similar to the discourse parsing framework we have presented [19]. These knowledge component estimates could be a critical component in generating an effective teaching plan, and they should be included in the tutor’s Contextual Knowledge Base.

Similarly, preference for teaching styles can be accounted for when trying to efficiently convey a concept to a student. Just as problems a student must solve entail a mixture of knowledge components, different methods of conveying information could consist of a mixture of style components, such as visual, auditory, interactive, with intrinsic pacing or student driven pacing. A lecture oriented approach with visual aids would include visual, auditory, and intrinsic pacing style components, but not the interactive or student-driven pacing style components. A problem set, on the other hand, would be interactive and student driven, but not auditory. Quantitative measures of student progress give us the best signal with which to estimate a student’s preferences. After being given instruction with a method consisting of a known style components, the estimates for the student’s preferences would be modified proportional to the increase in the student’s quantitative performance. Similarly, students could explicitly specify their preferences through natural language or by evaluating the effectiveness of different teaching methods. By estimating a student’s predilection towards certain style components, the tutor could select a teaching style that would most effectively convey the desired information, with the tutoring system providing assistive dialogue when appropriate.
Chapter 5

Proposed Work in Dialogue Planning with Contextual Social Features

We now turn to the task of storing and utilizing the information presented in the previous chapter in the dialogue planning task. The task of dialogue planning is to select the ITS's dialogue choices, and other accompanying actions such as assigning quizzes and playing videos, in order to improve a student’s quantitative skill at some task. Many conversational ITSs use a greedy approach to selecting actions based on the most recent interaction with a student [61]. To see an example where such an approach will fail, imagine the tutor has attempted to convey a concept to a student, but the student has failed to understand the concept. A greedy approach would likely attempt to communicate the concept again, verbatim, despite the fact that the previous attempt was unsuccessful. A more intelligent tutor would recognize that the previous tactic was unsuccessful, and try to communicate the concept in a different way, or step back and verify the student’s understanding of a previously learned idea required to grasp the new concept.

Our state space must be large and highly expressive to encode the diverse array of social factors that could determine the success of a dialogue. However, such a large and sparse state space induces a challenging learning problem. In this chapter we discuss the Contextual Knowledge Base representation that allows us to build a powerful state space using information gained from many linguistic and non-linguistic sources. We then propose the use of predictive state representations (PSR), to transform this state space into a more tractable lower dimensional subspace. Finally, we describe imitation learning frameworks, that we propose to use in conjunction with a PSR to create a dialogue policy based on human expert demonstration.

5.1 Contextual Knowledge Bases

A Contextual Knowledge Base is a set of facts about a user and the environment [25]. In dialogue systems, examples of entries in a CKB may include the semantic representation of the user’s most recent utterances, the current time of day, the user’s current physical engagement, the noise of the environment, the user’s age and gender, or an estimate of the user’s aptitude for various deliberative tasks.

The information is stored as logical forms created by combining concepts, relations and pred-
icates. This method of representing information allows us to incorporate the output of semantic and discourse parsing algorithms into the CKB very easily. Information from other sources, such as cameras, microphones, or GPS units, can also be incorporated into the knowledge base. For instance, we could represent the knowledge that a student has previously asked for the meaning of the pythagorean theorem in logical form using the following expression:

\[
\text{student\_request(definition[pythagorean\_theorem])}
\]

This logical form is composed of a concept, \textit{pythagorean\_theorem}, coupled with two predicates, \textit{definition} and \textit{student\_request}.

By allowing the agent to use the CKB when planning dialogue interactions, we can respond to critical information that would be ignored by a traditional conversational agent, such as the student’s knowledge level, the time of day, the state of the dialogue, the student’s physical engagement, and the student’s predilection towards different teaching styles. If a measurable outcome metric is defined, the system can be trained using a traditional dynamical system such as a Conditional Random Field (CRF) \cite{41}.

Figure 5.1 shows preliminary results training a CRF with simulated CKB inputs. The simulation consisted of several simulated users, with user \(i\) employing a hidden function \(F_i(s, a) \rightarrow c\), which defined the cost \(c\) for the tutoring agent to take action \(a\) when the user is in state \(s\). The teaching agent began with a partially populated CKB defined by the original state space. Some actions resulted in new elements being added to the CKB, which allowed the agent to refine its estimate of the current state. Each experiment represented the agent trying to learn the cost function of one specific user, denoted user \(k\) with cost function \(F_k\). 90\% of the training data was taken from users other than user \(k\), and the remaining data was taken from user \(k\). Each data point was accompanied by the originating user, which allowed the agent to weight some users more highly if their behavior was more similar to user \(k\). The evaluation then required the agent to select a series of actions to minimize \(F_k\). The simulated data was of lower dimensionality than a practical real world CKB, regardless, we see a sharp decrease in the average cost of the agents actions after only a small sample of interactions with the simulation.

Moving forward, we foresee some challenges that will need to be addressed to deploy a CKB in a real world dialogue planning application. One such challenge will be the need to automatically curate the knowledge base to remove old and irrelevant entries in order to prevent the state space from growing unnecessarily large. A naive solution would be to store all entries indefinitely, or to only store information for a fixed period of time. However, we believe that some entries in the CKB will remain helpful for long periods of time, such as a user’s explicitly stated preferences, while other entries will only be useful temporarily, such as a user’s current physical engagement. As such, we will investigate more sophisticated techniques for curating the contents of the knowledge base over time. We also plan to incorporate subspace identification methods to make the state space more tractable, and the extent of curation that is necessary will depend in part of the effectiveness of the state space compression.
Figure 5.1: The Average Action Cost of a CRF with Contextual Knowledge Base input

5.2 Predictive State Representations

As we have alluded to previously, a state space that is rich enough to express all of the relevant contextual factors we wish to consider will necessarily be sparse and high dimensional. We will need to incorporate a state space transformation method to make planning more tractable. One promising avenue of research for training a dynamical system, particularly in high dimensional space, is the use of Predictive State Representations (PSRs) [65]. A central idea in this approach is to embed the state of the dynamical system in a lower dimensional predictive state representation discovered through a subspace identification process. To accomplish this task, we take the history of a sequence of observations in a matrix $Y_P$ and the future observations that we wish to predict in a matrix $Y_F$—for simplicity we assume that these matrices both have a dimensionality of $n \times n$. We can recover a set of predictive coefficients, $W$, using matrix inversion, this can also be done using the More-Penrose pseudoinverse if the matrix is non-invertable:

$$W = Y_F \cdot Y_P^{-1}$$

Once we compute the predictive coefficients, we can factor these coefficients using Singular Value Decomposition (SVD):

$$W = U \cdot V$$

We now have a factorization of the predictive matrix, where $|U| = n \times m$ and $|V| = m \times n$. We can then embed the original observation history, $Y_P$, into the lower dimensional predictive state representation by simply computing $Y_P \cdot U$.

The quality of the predictive subspace will depend a great deal on the original state space. Intuitively, we expect a larger original state space to be more likely to contain a useful Predictive State Representation, but identifying this subspace becomes more difficult as the original state space grows. Our challenge will be to create a representation of contextual features that produces
the most informative PSR. This task is related to the knowledge base curation problem identified in the previous section. We plan to develop methods for automatic state space generation using CKB entries that will result in a Predictive State Representation that is conducive to dialogue planning.

5.3 Policy Learning

The predictive state representation, or the more general transformed predictive state representation [58], can be combined with reinforcement learning algorithms, such as least squared temporal differencing, to learn a new policy [5]. The most readily available datasets in the education domain consist of expert demonstrations of near optimal policies, specifically human tutors interacting with human students. Therefore, we propose an alternate, novel approach of coupling the PSR embedding with an imitation learning algorithm, such as those based on the principle of maximum causal entropy [77]. In an imitation learning task, we are given the state/action sequential pairs of an expert performing some task. The goal is then to recover the utility or cost function of the expert.

Given a log of human tutors interacting with students, we can generate a Contextual Knowledge Base populated by linguistic context using the semantic and discourse parsing algorithms described in chapter 4. Additionally, if the logs are accompanied by supplementary information, such as student evaluations, logs describing the student’s studying habits, or video or audio logs recording the tutoring interactions, we will use this additional information to populate the training knowledge base with additional contextual features as well. Imitation learning can then allow us to identify the latent policy employed by the tutors, for instance, we may discover that when faced with a direct question from a student, tutors often respond with a direct answer followed by a more detailed elaboration (as seen in Figure 5.2).

It is possible the the factors that human tutors are using to make their teaching decisions will not be observable in the tutoring datasets that are most readily available, which could make imitation learning difficult or impossible. Latent variable models may help to mitigate this problem, but if the policies learned through human tutor demonstration are not sufficient, we will need to pursue a more traditional reinforcement learning framework for dialogue planning. Initial evaluation of such a framework could be conducted using a simulated testing environment, and supplementary evaluation can be conducted later with human volunteers.
Chapter 6

Conclusions

Contextual linguistic and environmental information are critical components in human conversation, and they must be addressed to improve the effectiveness of conversational systems. We have developed several techniques for identifying relevant features using the dialogue as well as environmental sensors. One of the chief challenges for incorporating these techniques into an ITS is the current need for annotated training data, particularly with discourse and semantic parsing. Going forward, we will be investigating active, semi-supervised training to reduce the need for human supervision [22], which will allow us to leverage unlabeled data, which is readily available, and only request annotations for the most informative data points. If we are also able to identify fully latent structures and labels using spectral methods, we can further improve the effectiveness of using unlabeled data.

Modeling the myriad of factors entailed in human social interaction necessarily entails the use of a sparse and complex state space representation. We have identified a suitably expressive state space representation with Contextual Knowledge Bases, which allows us to conveniently incorporate contextual information from multiple linguistic and environmental inputs. However, generating a policy for interacting with users in this state space is a difficult and unsolved problem. However, we have identified some useful ideas that we believe will make this problem more tractable, such as Transformed Predictive State Representations and Maximum Entropy Reinforcement Learning.

Building intelligent agents with improved social and environmental awareness requires us to close the loop on context modeling and interaction planning. In this thesis proposal, we have identified what we believe are some of the most promising next steps towards building more competent social agents that go beyond the simple question answering systems seen on the market today. By applying this research towards education, one of the foundational elements of a productive and healthy society, we hope to build systems that can improve the learning experience for any student with access to a personal computer.
Chapter 7

Thesis Schedule

What follows is a tentative schedule for completing the remainder of the thesis.

- **June-August 2014: Dataset generation for parsing algorithms and initial parsing tests within education domain**
  In this period, we will define the semantic parsing ontology and the discourse parsing relations that we believe will be necessary within the tutoring application. We will utilize a summer undergraduate intern working on the project to produce small, labeled datasets for both problems that can be used for evaluation and also as a template for future annotations if additional labeled training data is required.

- **June-December 2014: Development of unsupervised or semi-supervised parsing algorithms**
  This time will be spent investigating spectral methods for learning latent structure, particularly for semantic and discourse parsing. We have already developed frameworks for parsing with semi-supervision and active learning, and these frameworks can be used in conjunction with or in lieu of a fully unsupervised approach if our results using unsupervised methods are unsatisfactory.

- **January-May 2015: Development of CKB pipeline**
  In this stage, we will integrate all of the algorithms we will use to generate our contextual state space. This is also the phase in which we will consider the need for knowledge base curation and integrate a Predictive State Representation with our Contextual Knowledge Base.

- **May 2015-August 2015: Integration and development of planning systems**
  Given a compressed contextual state space, we will begin exploring options for dialogue planning. We will begin with imitation learning algorithms trained using human demonstration and then explore alternative reinforcement learning frameworks as necessary. If we determine that human volunteers will be required to test the system, the Institutional Review Board documents will be drafted and submitted to the administration at Carnegie Mellon in preparation for testing with student volunteers.

- **Fall 2015: Final evaluations and thesis presentation**
  Evaluations of the effectiveness of the overall system will be conducted using simulation, trials with human volunteers, or some combination of the two. These evaluations will
form the basis for the final thesis document, with the thesis defense planned for the Fall semester, 2015.
Bibliography


[2] Raphaël Bailly, Xavier Carreras, Franco M Luque, and Ariadna Quattoni. Unsupervised spectral learning of wcfg as low-rank matrix completion. 4.5.1


[6] Byron Boots, Arunkumar Byravan, and Dieter Fox. Learning predictive models of a depth camera & manipulator from raw execution traces. 1


[19] Mohammad H Falakmasir, Zachary A Pardos, Geoffrey J Gordon, and Peter Brusilovsky. A spectral learning approach to knowledge tracing. 4.5.3


[44] Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. A pdtb-styled end-to-end discourse parser. *Natural Language Engineering*, pages 1–34, 2012. 4.3.1, 4.3.1, 4.3.3


[52] Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. The penn discourse treebank 2.0. In LREC. Citeseer, 2008. 4.3, 4.3.1


[54] Xian Qian and Yang Liu. Branch and bound algorithm for dependency parsing with non-local features. TACL, 1:37–48, 2013. 4.5.1


[64] B. Settles and X. Zhu. Behavioral factors in interactive training of text classifiers. 2


[77] Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. Modeling interaction via the principle of maximum causal entropy. 2010. 5.3