Unsupervised Learning and Modeling of Knowledge and Intent for Spoken Dialogue Systems

Yun-Nung (Vivian) Chen

Ph.D. Thesis Proposal

Thesis Committee:

Dr. Alexander I. Rudnicky, Carnegie Mellon University
Dr. Anatole Gershman, Carnegie Mellon University
Dr. Alan W Black, Carnegie Mellon University
Dr. Dilek Hakkani-Tür, Microsoft Research

April 2015

Abstract

Spoken language interfaces are being incorporated into various devices (smart-phones, smart TVs, in-car navigating system, etc). The role of spoken language understanding is of great significance to a successful spoken dialogue system: in order to capture the language variation from dialogue participants, the spoken language understanding component must create a mapping between the natural language inputs and semantic representations that captures users' intentions.

The semantic representation must include "concept", "structure", etc, where concepts are the important semantics capturing the domain-specific topics, and the structure describes the relation between concepts and conveys high-level intention. However, spoken dialogue systems typically use manually predefined semantic elements to parse users' utterances into unified semantic representations. To define the knowledge and the structure, domain experts and professional annotators are often involved, and the cost of development can be expensive. Therefore, current technology usually limits conversational interactions to a few narrow predefined domains/topics. With the increasing conversational interactions, this dissertation focuses on improving generalization and scalability of building dialogue systems by automating the knowledge inference and structure learning from unlabelled conversations.

In order to achieve the goal, two questions need to be addressed: 1) Given unlabelled raw audio recordings, how can a system automatically induce and organize the domain-specific concepts? 2) With the automatically acquired knowledge, how can a system understand individual utterances and user intents? To tackle the above problems, we propose to acquire the domain knowledge that can be used in specific applications in order to capture human's semantics, intent, and behavior, and then build an SLU component based on the learned knowledge to offer better interactions in dialogues.

The dissertation mainly focuses on five important stages: ontology induction, structure learning, surface form derivation, semantic decoding, and behavior prediction. To solve the first problem, ontology induction automatically extracts the domain-specific concepts by leveraging available ontologies and distributional semantics. Then an unsupervised machine learning approach is proposed to learn the structure and then infer the meaningful organization for the dialogue system design. Surface form derivation learns natural languages that describe elements from the ontology and convey the domain knowledge to infer better understanding.

For the second problem, the learned knowledge can be utilized to decode users' semantics and predict the follow-up behaviors in order to provide better interactions.

In conclusion, the dissertation shows that it is feasible to build a dialogue learning system that is able to understand how particular domains work based on unlabelled conversations. As a result, the initial spoken dialogue system can be automatically built according to the learned knowledge, and its performance can be quickly improved by interacting with users for practical usage, presenting the potential of reducing human effort for spoken dialogue system development.

Keywords

Distributional Semantics
Knowledge Graph
Domain Ontology
Matrix Factorization (MF)
Random Walk
Spoken Language Understanding (SLU)
Spoken Dialogue System (SDS)
Word Embeddings

List of Abbreviations

AF Average F-Measure is an evaluation metric that measures the performance of a ranking list by averaging the F-measure over all positions in the ranking list.

AMR Abstract Meaning Representation is a simple and readable semantic representation in AMR Bank.

AP Average Precision is an evaluation metric that measures the performance of a ranking list by averaging the precision over all positions in the ranking list.

ASR Automatic Speech Recognition, also known as computer speech recognition, is the process of converting the speech signal into written text.

AUC Area Under the Precision-Recall Curve is an evaluation metric that measures the performance of a ranking list by averaging the precision over a set of evenly spaced recall levels in the ranking list.

BPR Bayesian Personalized Ranking is an approach to learn with implicit feedback for matrix factorization, especially used in item reommendation.

CBOW Continuous Bag-of-Words is an architecture for learning distributed word representations, which is similar to the feedforward neural net language model but uses continuous distributed representation of the context.

CMU Carnegie Mellon University is a private research university in Pittsburgh.

FE Frame Element is a descriptive vocabulary for the components of each frame.

ISCA International Speech Communication Association is a non-profit organization that aims to promote, in an international world-wide context, activities and exchanges in all fields related to speech communication science and technology.

LTI Language Technologies Institute is a research department in the School of Computer Science at Carnegie Mellon University.

LU Lexical Unit is a word with a sense.

MF Matrix Factorization is a decomposition of a matrix into a product of matrices in the discipline of linear algebra.

NLP Natural Language Processing is a field of artificial intelligence and linguistics that studies the problems intrinsic to the processing and manipulation of natural language.

POS Part of Speech tag, also known as word class, lexical class or lexical class are traditional categories of words intended to reflect their functions within a sentence.

SDS Spoken Dialogue System is an intelligent agent that interacts with a user via natural spoken language in order to help the user obtain desired information or solve a problem more efficiently.

SGD Stochastic Gradient Descent is a gradient descent optimization method for miniming an objective function that is written as a sum of differentiable functions.

SLU Spoken Language Understanding is a component of a spoken dialogue system, which parses the natural languages into semantic forms that benefit the system's understanding.

SVM Support Vector Machine is a supervised learning method used for classification and regression based on the Structural Risk Minimization inductive principle.

WAP Weighted Average Precision is an evaluation metric that measures the performance of a ranking list by weighting the precision over all positions in the ranking list.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Spoken Dialogue System

A spoken dialogue system (SDS) is an intelligent agent that interacts with a user via natural spoken language in order to help the user obtain desired information or solve a problem more efficiently. As for current technologies, a dialogue system is one of many spoken language applications that operate on limited specific domains. For instance, the CMU Communicator system is a dialogue system for the air travel domain that provides information about flight, car, and hotel reservations [81]. Another example, the JUPITER system [104], is a dialogue system for a weather domain, which provides forecast information for the requested city. More recently, a number of efforts in industry (e.g. Google Now[1], Apple's Siri[2], Microsoft's Cortana[3], and Amazon's Echo[4]) and academia [1, 8, 34, 42, 61, 63, 73, 77, 78, 82, 97, 98] have focused on developing semantic understanding techniques for building better SDSs.

An SDS typically is composed of the following components: an automatic speech recognizer (ASR), a spoken language understanding (SLU) module, a dialogue manager, a natural language generation module, and a speech synthesizer. When developing a dialogue system in a new domain, we may be able to reuse some components that are designed independently of domain-specific information, for example, the speech recognizer and the speech synthesizer. However, the components that are integrated with domain-specific information have to be reconstructed for each new domain, and the cost of development is expensive. Usually participants engage in a conversation in order to achieve a specific goal such as accomplishing a task in their mind or getting the answers to the questions, for example, to obtain the list of restaurants in a specific location. Therefore in the context of this dissertation, domain-specific information refers to the knowledge specific to a task that an SDS has to support rather than the knowledge about general dialogue mechanisms. The dissertation mainly focuses on two parts:

- **Knowledge acquisition** is to learn the domain-specific knowledge that is used by a

[1] http://www.google.com/landing/now/
[2] http://www.apple.com/ios/siri/
[3] http://www.microsoft.com/en-us/mobile/campaign-cortana/
[4] http://www.amazon.com/oc/echo

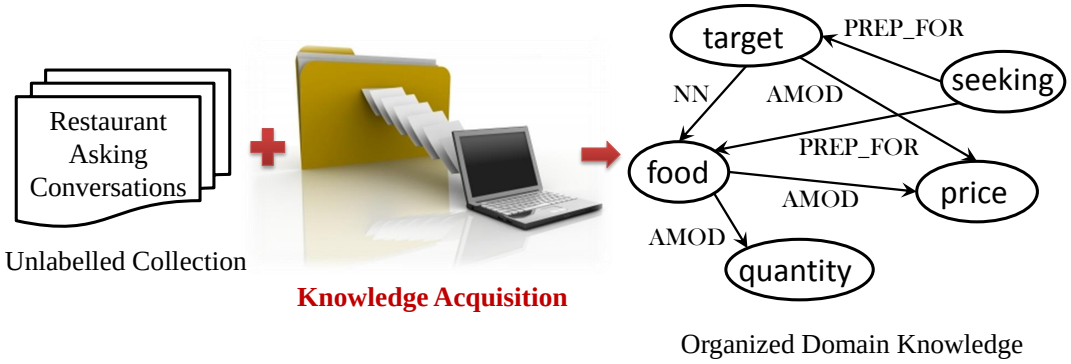


Figure 1.1: An example output of the proposed knowledge acquisition approach.

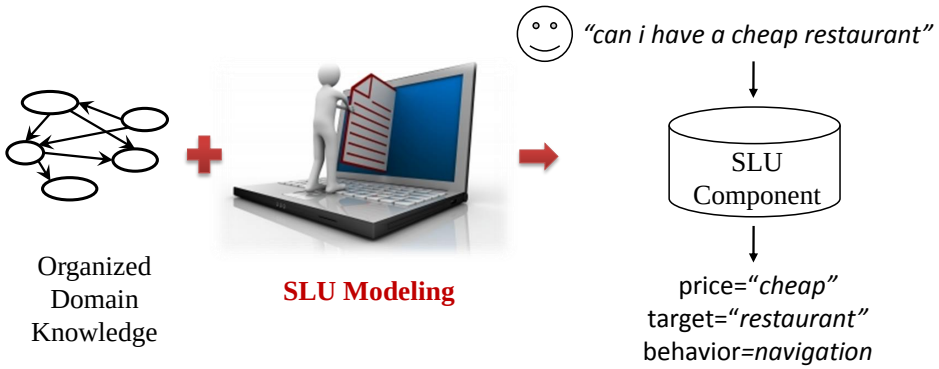


Figure 1.2: An example output of the proposed SLU modeling approach.

spoken language understanding (SLU) component. The domain-specific information used by an SLU component includes the organized ontology that a dialogue system has to support in order to successfully understand the actual meaning. An example of the necessary domain knowledge about restaurant recommendation is shown in Figure 1.1, where the learned domain knowledge contains the semantic slots and their relations[5].

- **SLU modeling** is to build an SLU module that is able to understand the actual meaning of domain-specific utterances based on the domain-specific knowledge and then further provide better responses. An example of the corresponding understanding procedure in the restaurant domain is shown in Figure 1.2, where the decoded output by the SLU component is the semantic form of the input utterance.

Conventionally the domain-specific knowledge is defined manually by domain experts or developers, who are familiar with the specific domains. For common domains like a weather domain or a bus domain, system developers are usually able to identify the such information.

[5]The slot is defined as a semantic unit usually used in dialogue systems.

However, some domains, such as a military domain [7], require the domain experts, which makes the knowledge engineering process more difficult. Furthermore, the experts' decision may be subjective and may not cover all possible real-world users' cases [99]. Therefore, poor generalization results in the limited predefined information and even biases the subsequent data collection and annotation. Another issue is about the efficiency: the manual definition and annotation process for domain-specific tasks can be very time-consuming, and have high financial costs. Finally, the maintenance cost is also non-trivial: when new conversational data comes in, developers, domain experts, and annotators have to manually analyze the audios or the transcriptions for updating and expanding the ontologies. With more available conversational data, to acquire the domain knowledge, recent approaches are data-driven in terms of generalization and scalability.

In the past decade, the computational linguistics community has focused on developing language processing algorithms that can leverage the vast quantities of available raw data. Chotimongkol proposed a machine learning technique to acquire domain-specific knowledge, showing the potential of reducing human effort in the SDS development [24]. However, the work mainly focused on the low-level semantic units like word-level concepts. With increasing high-level knowledge resources, such as knowledge bases, this dissertation moves forward to investigate the possibility of developing a high-level semantic conversation analyzer for a certain domain using an unsupervised machine learning approach. The human's semantics, intent, and behavior can be captured from a collection of unlabelled raw conversational data, and then be modeled for building a good SDS.

Considering the practical usage, the acquired knowledge may be revised manually to improve the system performance. Even though some revision might be required, the cost of revision is significantly lower than the cost of analysis. Also, the automatically learned information may consider the real-world users' cases and avoid biasing the subsequent annotation. This thesis focuses on the highlighted parts, inducing acquiring domain knowledge from the dialogues using the available ontology, and modeling the SLU module using the automatically learned information. The proposed approach combining both data-driven and knowledge-driven perspectives shows the potential of improving *generalization*, *maintenance*, *efficiency*, and *scalability* of dialogue system development.

1.2 Thesis Statement

The main purpose of this work is to automatically develop SLU components for SDSs by using the automatically learned domain knowledge in an unsupervised fashion. This dissertation mainly focuses on acquiring the domain knowledge that is useful for better understanding and designing the system framework and further modeling the semantic meaning of the spoken

language. For knowledge acquisition, there are two important stages – **ontology induction** and **structure learning**. After applying them, an organized domain knowledge is inferred from the unlabeled conversations. For SLU modeling, there are two aspects – **semantic decoding** and **behavior prediction**. Based on the learned information, semantic decoding analyzes the meaning in each individual utterance and behavior prediction model user intents and predicts possible follow-up behaviors. In conclusion, the thesis demonstrates the feasibility of building a dialogue learning system that is able to automatically learn the important knowledge and understand how the domains work based on unlabelled raw audio. With the domain knowledge, then the initial dialogue system can be constructed and improved quickly by interacting with users. The main contribution of the dissertation is presenting the potential of reducing human work and showing the feasibility of improving efficiency for dialogue system development by automating the knowledge learning process.

1.3 Thesis Structure

The thesis proposal is organized as below.

- **Chapter 2 - Background and Related Work**
  This chapter reviews some background knowledge and summarizes related works. The chapter also discusses current challenges of the task, describes several structured knowledge resources and presents distributional semantics that may benefit understanding problems.

- **Chapter 3 - Ontology Induction for Knowledge Acquisition**
  This chapter focuses on inducing the ontology that are useful for developing SLU modules of SDSs based on the available structured knowledge resources in an unsupervised way. Some of the contributions were published [20, 22]

  – Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky, "Unsupervised Induction and Filling of Semantic Slots for Spoken Dialogue Systems Using Frame-Semantic Parsing," in *Proceedings of 2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU'13)*, Olomouc, Czech Republic, 2013.
  (Student Best Paper Award)

  – Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky, "Leveraging Frame Semantics and Distributional Semantics for Unsupervised Semantic Slot Induction for Spoken Dialogue Systems," in *Proceedings of 2014 IEEE Workshop on of Spoken Language Technology (SLT'14)*, South Lake Tahoe, Nevada, USA, 2014.

- **Chapter 4 - Structure Learning for Knowledge Acquisition**
  This chapter focuses on learning the structures, such as the inter-slot relations, for helping SLU development. Some of the contributions were published [23]:
  - Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky, "Jointly Modeling Inter-Slot Relations by Random Walk on Knowledge Graphs for Unsupervised Spoken Language Understanding," in *Proceeding of The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*, Denver, Colorado, USA, 2015.
- **Chapter 5 - Surface Form Derivation for Knowledge Acquisition**
  This chapter focuses on deriving the surface forms conveying semantics for the entities from the ontology, where the derived information helps predict the probability of semantics given the observation more accurately. Some of the contributions were published [21]:
  - Yun-Nung Chen, Dilek Hakkani-Tür, and Gokhan Tur, "Deriving Local Relational Surface Forms from Dependency-Based Entity Embeddings for Unsupervised Spoken Language Understanding," in *Proceedings of 2014 IEEE Workshop of Spoken Language Technology (SLT'14)*, South Lake Tahoe, Nevada, USA, 2014.
- **Chapter 6 - Semantic Decoding in SLU Modeling**
  This chapter focuses on decoding the users' spoken languages into corresponding semantic forms, which is the task of SLU models. Some of the contributions were under review:
  - Yun-Nung Chen, William Yang Wang, Anatole Gershman, and Alexander I. Rudnicky, "Matrix Factorization with Knowledge Graph Propagation for Unsupervised Spoken Language Understanding," under submission.
- **Chapter 7 - Behavior Prediction in SLU Modeling**
  This chapter focuses on modeling the behaviors in the SLU component, so that the SDS is able to predict the users' behaviors and further provide better interactions. Some of the contributions were published [18]:
  - Yun-Nung Chen and Alexander I. Rudnicky, "Dynamically Supporting Unexplored Domains in Conversational Interactions by Enriching Semantics with Neural Word Embeddings," in *Proceedings of 2014 IEEE Workshop of Spoken Language Technology (SLT'14)*, South Lake Tahoe, Nevada, USA, 2014.
- **Chapter 7 - Conclusions and Future Work**
  This chapter makes the conclusions and presents the proposed work and the timeline.

2 Background and Related Work

2.1 Semantic Representation

Considering to understand natural language for machines, a semantic representation is introduced. A semantic representation for an utterance carries its core content so that the actual meaning behind the utterance can be inferred only through the representation. For example, an utterance "*show me action movies directed by james cameron*" can be represented as target="*movie*", genre="*action*", director=*james cameron*. Another utterance, "*find a cheap taiwanese restaurant in oakland*" can be formed as target="*restaurant*", price="*cheap*", type="*taiwanese*", location="*oakland*". The semantic representations are able to convey the whole meaning of the utterances, which can be more easily processed by machines. The semantic representation is not unique, and there are several forms for representing the meaning. Below we describe two types of semantic forms:

- Slot-Based Semantic Representation
  The slot-based representation includes flat semantic concepts, which are usually used in simpler tasks. Above examples belong to slot-based semantic representation, where semantic concepts are target, location, price, etc.

- Relation-Based Semantic Representation
  The relation-based representation includes structured concepts, which are usually used in tasks that have more complicate dependency relations. For instance, "*show me action movies directed by james cameron*" can be represented as movie.directed_by, movie.genre, director.name="*james cameron*", genre.name="*action*". A semantic slot in the slot-based representation is formed as relations, which can be either two concepts or the concept's name.

2.2 Spoken Language Understanding (SLU) Component

The main purpose of an SLU component is to convert the natural language into semantic forms, which is also called as semantic decoding or semantic parsing. Building the state-of-the-art semantic parsing system needs the large training data with annotations. For example,

Berant *et al.* proposed SEMPRE[1], which used the web-scaled knowledge bases to train the semantic parser [6]. Das *et al.* proposed SEMAFOR[2], which utilized a lexicon developed based on a linguistic theory – Frame Semantics to train the semantic parser [29].

The SLU module includes following challenges:

- How to define the semantic elements from the unlabelled data?
- How to understand the structure between defined elements?
- How to detect the semantics for the testing data?
- How to use the learned information to predict user behaviors for improving the system performance?

2.3 Ontology and Knowledge Base

There are two main types of knowledge resources available, generic concept and entity-based, both of which may benefit SLU modules for SDSs. The generic concept knowledge bases cover the concepts that are more common, such as a food domain and a weather domain. The entity-based knowledge bases usually contain a lot of named entities that are specific for certain domains, for example, a movie domain and a music domain. The following describes several knowledge resources, which contain the rich semantics and may benefit the understanding task.

2.3.1 Generic Concept Knowledge

There are two semantic knowledge resources, FrameNet and Abstract Meaning Representation (AMR).

- **FrameNet**[3] is a linguistically semantic resource that offers annotations of predicate-argument semantics, and associated lexical units for English [3]. FrameNet is developed based on semantic theory, Frame Semantics [38]. The theory holds that the meaning of most words can be expressed on the basis of semantic frames, which encompass three major components: frame (F), frame elements (FE), and lexical units (LU). For example, the frame "food" contains words referring to items of food. A descriptor frame element within the food frame indicates the characteristic of the food. For example, the phrase

[1] `http://www-nlp.stanford.edu/software/sempre/`
[2] `http://www.ark.cs.cmu.edu/SEMAFOR/`
[3] `http://framenet.icsi.berkeley.edu`

Frame: Revenge	
Noun	*revenge, vengeance, reprisal, retaliation*
Verb	*avenge, revenge, retaliate (against), get back (at), get even (with), pay back*
Adjective	*vengeful, vindictive*
FE	avenger, offender, injury, injured_party, punishment

Table 2.1: The frame example defined in FrameNet.

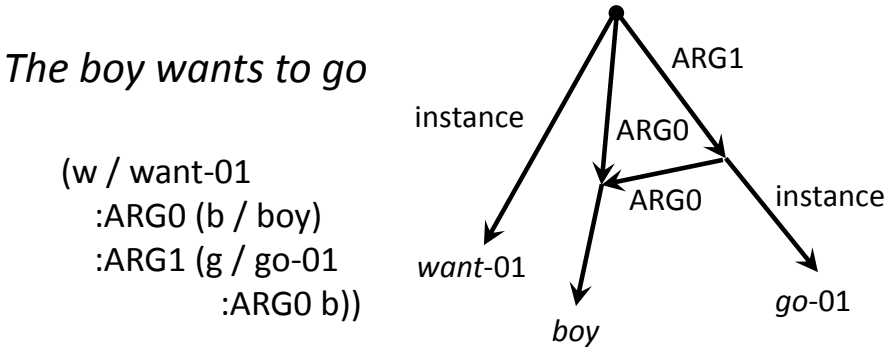


Figure 2.1: A sentence example in AMR Bank.

"*low fat milk*" should be analyzed with "*milk*" evoking the food frame, where "*low fat*" fills the descriptor FE of that frame and the word "*milk*" is the actual LU. A defined frame example is shown in Table 2.1.

- **Abstract Meaning Representation (AMR)** is a semantic representation language including the meanings of thousands of English sentences. Each AMR is a single rooted, directed graph. AMRs include PropBank semantic roles, within-sentence coreference, named entities and types, modality, negation, questions, quantities, etc [4]. The AMR feature structure graph of an example sentence is illustrated in Figure 2.1, where the "*boy*" appears twice, once as the ARG0 of *want-01*, and once as the ARG0 of *go-01*.

2.3.2 Entity-Based Knowledge

- **Semantic Knowledge Graph** is a knowledge base that provides structured and detailed information about the topic with a lists of related links. Three different knowledge graph examples, Google's knowledge graph[4], Microsoft's Bing Satori, and Freebase, are shown in Figure 2.2. The semantic knowledge graph is defined by a schema and composed of nodes and edges connecting the nodes, where each node represents an entity-type and the edge between each node pair describes their relation, as called as property. An example from Freebase is shown in Figure 2.3, where nodes represent core entity-

[4] http://www.google.com/insidesearch/features/search/knowledge.html

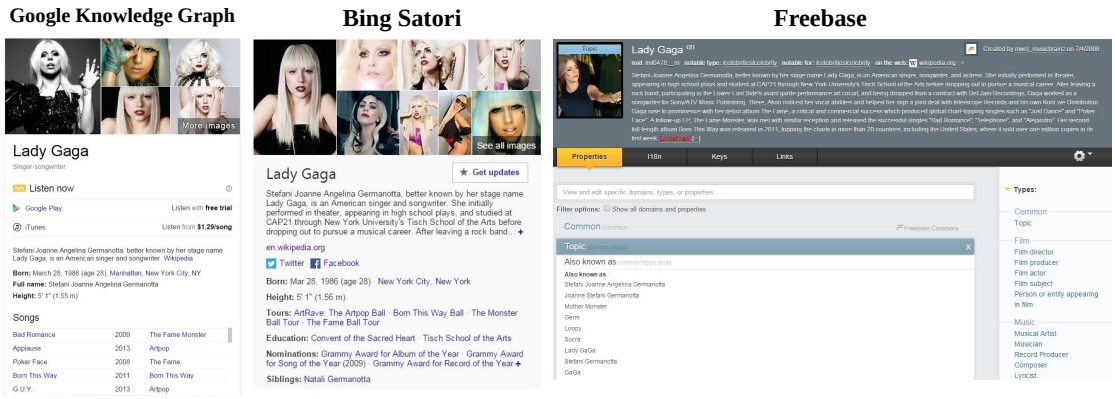


Figure 2.2: Three famous semantic knowledge graph examples (Google's Knowledge Graph, Bing Satori, and Freebase) corresponding to the entity "Lady Gaga".

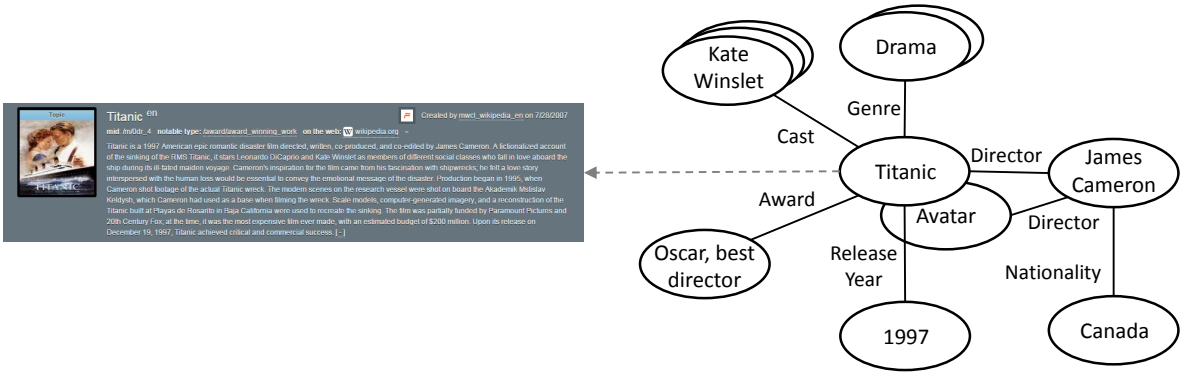


Figure 2.3: A portion of the Freebase knowledge graph related to the movie domain.

types for the movie domain. The domains in the knowledge graphs span the web, from "American Football" to "Zoos and Aquariums".

- **Wikipedia**[5] is a free-access, free content Internet encyclopedia, which contains a large number of pages/articles related to a specific entity [74]. It is able to provide basic background knowledge for help understanding tasks in the natural language processing (NLP) field.

[5] `http://en.wikipedia.org/wiki/Wikipedia`

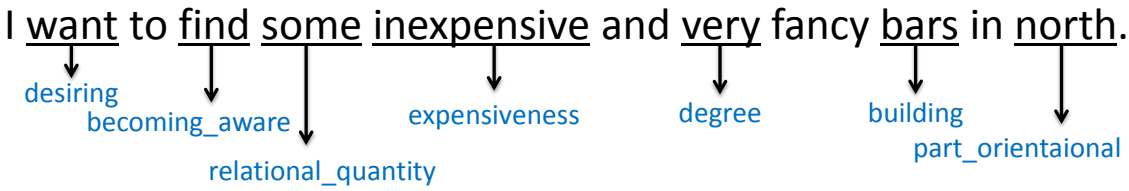


Figure 2.4: An example of FrameNet categories for ASR output labelled by probabilistic frame-semantic parsing.

2.4 Knowledge-Based Semantic Analyzer

2.4.1 Generic Concept Knowledge

- FrameNet
  SEMAFOR[6] is a state-of-the-art semantic parser for frame-semantic parsing [27, 28]. Trained on manually annotated sentences in FrameNet, SEMAFOR is relatively accurate in predicting semantic frames, FE, and LU from raw text. Augmented by the dual decomposition techniques in decoding, SEMAFOR also produces the semantically-labeled output in a timely manner. Note that SEMAFOR does not consider the relations between frames but treat each frame independently. Figure 3.2 shows the output of probabilistic frame-semantic parsing.

- Abstract Meaning Representation (AMR)
  JAMR[7] is the first semantic parser that parses the sentences into AMRs [39]. Trained on manually defined AMR Bank, JAMR applied an algorithm for finding the maximum, spanning, connected subgraph and showed how to incorporate extra constraints with Lagrangian relaxation. Figure 2.5 shows the output of JAMR on the example sentence.

2.4.2 Entity-Based Knowledge

- Semantic Knowledge Graph
  Freebase API[8] is an API for accessing the data, and the data can also be dumped directly.

- Wikipedia
  Wikifier[9] is an entity linking (a.k.a. Wikification, Disambiguation to Wikipedia (D2W)) tool. The task is to identify concepts and entities in texts and disambiguate them into

[6] http://www.ark.cs.cmu.edu/SEMAFOR/
[7] http://github.com/jflanigan/jamr
[8] https://developers.google.com/freebase/
[9] http://cogcomp.cs.illinois.edu/page/software_view/Wikifier

show me what richard lester directed

```
(s / show-01
      :ARG1 (d / direct-01
            :ARG0 (p / person
                  :name (n / name
                        :op1 "lester"
                        :op2 "richard"))))
```

Figure 2.5: An example of AMR parsed by JAMR on ASR output.

Michael Jordan is my favorite player.

Michael Jordan is a machine learning expert.

Figure 2.6: An example of Wikification.

the corresponding Wikipedia pages. An example is shown in Figure 2.6, where the entities "Micheal Jordan" in two sentences refer to different people, pointing to different Wikipedia pages.

2.5 Distributional Semantics

The distributional view of semantics hypothesizes that words occurring in the same contexts may have similar meanings [47]. As the foundation for modern statistical semantics [40], an early success that implements this distributional theory is Latent Semantic Analysis [32]. Recently, with the advance of deep learning techniques, the continuous representations as word embeddings have further boosted the state-of-the-art results in many applications, such as sentiment analysis [86], language modeling [68], sentence completion [70], and relation detection [21].

In NLP, Brown *et al.* proposed an early hierarchical clustering algorithm that extracts word clusters from large corpora [14], which has been used successfully in many NLP applications [67]. Comparing to standard bag-of-words n-gram language models, in recent years, continuous word embeddings (a.k.a. word representations, or neural embeddings) are shown

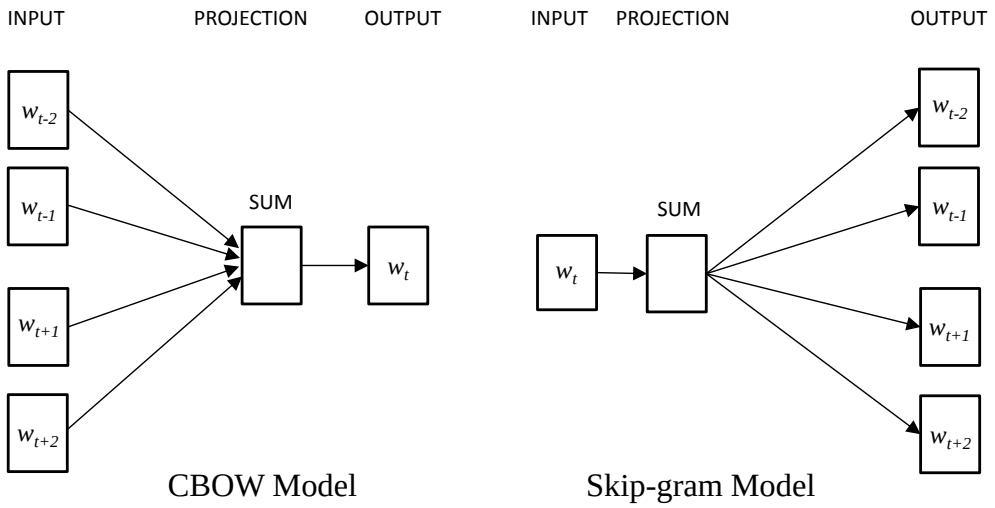


Figure 2.7: The CBOW and Skip-gram architectures. The CBOW model predicts the current word based on the context, and the Skip-gram model predicts surrounding words given the target word [70].

to be the state-of-the-art in many NLP tasks, due to its rich continuous representations (e.g. vectors, or sometimes matrices, and tensors) that capture the context of the target semantic unit [93, 5].

The continuous word vectors are derived from a recurrent neural network architecture [69]. The recurrent neural network language models use the context history to include long-distance information. Interestingly, the vector-space word representations learned from the language models were shown to capture syntactic and semantic regularities [72, 71]. The word relationships are characterized by vector offsets, where in the embedded space, all pairs of words sharing a particular relation are related by the same constant offset.

The word embeddings are trained on the contexts of the target word, where the considered contexts can be linear or dependency-based described as follows.

2.5.1 Linear Word Embeddings

Typical neural embeddings use linear word contexts, where a window size is defined to produce contexts of the target words [72, 71, 70]. There are two model architectures for learning distributed word representations: continuous bag-of-words (CBOW) model and continuous skip-gram model, where the former predicts the current word based on the context and the latter predicts surrounding words given the current word.

2.5.1.1 Continuous Bag-of-Words (CBOW) Model

The word representations are learned by a recurrent neural network language model [69], as illustrated in Figure 2.7. The architecture contains an input layer, a hidden layer with recurrent connections, and the corresponding weight matrices. Given a word sequence $w_1, ..., w_T$, the objective function of the model is to maximize the probability of observing the target word w_t given its contexts $w_{t-c}, w_{t-c+1}, ..., w_{t+c-1}, w_{t+c}$, where c is the window size:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq i\leq c, i\neq 0}\log p(w_t \mid w_{t+i}). \tag{2.1}$$

2.5.1.2 Continuous Skip-Gram Model

The training objective of the skip-gram model is to find word representations that are useful for predicting the surrounding words, which is similar to the CBOW architecture. Given a word sequence as the training data $w_1, ..., w_T$, the objective function of the model is to maximize the average log probability:

$$\frac{1}{T}\sum_{i=t}^{N}\sum_{-c\leq i\leq c, i\neq 0}\log p(w_{t+i} \mid w_t) \tag{2.2}$$

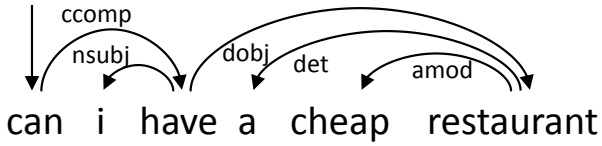
The objective can be trained using stochastic-gradient updates over the observed corpus.

2.5.2 Dependency-Based Word Embeddings

Most neural embeddings use linear bag-of-words contexts, where a window size is defined to produce contexts of the target words [72, 71, 70]. However, some important contexts may be missing due to smaller windows, while larger windows capture broad topical content. A dependency-based embedding approach was proposed to derive contexts based on the syntactic relations the word participates in for training embeddings, where the embeddings are less topical but offer more functional similarity compared to original embeddings [64].

Figure 2.8 shows the extracted dependency-based contexts for each target word from the dependency-parsed sentence, where headwords and their dependents can form the contexts by following the arc on a word in the dependency tree, and -1 denotes the directionality of the dependency. After replacing original bag-of-words contexts with dependency-based contexts, we can train dependency-based embeddings for all target words [100, 10, 11].

For training dependency-based word embeddings, each target x is associated with a vector



Word	Contexts
can	have/ccomp
i	have/nsubj^{-1}
have	can/ccomp^{-1}, i/nsubj, restaurant/dobj
a	restaurant/det^{-1}
cheap	restaurant/amod^{-1}
restaurant	have/dobj^{-1}, cheap/amod

Figure 2.8: The target words and associated dependency-based contexts extracted from the parsed sentence for training depedency-based word embeddings.

$\mathbf{v_x} \in \mathbb{R}^d$ and each context c is represented as a context vector $\mathbf{v_c} \in \mathbb{R}^d$, where d is the embedding dimensionality. We learn vector representations for both targets and contexts such that the dot product $\mathbf{v_x} \cdot \mathbf{v_c}$ associated with "good" target-context pairs belonging to the training data $\mathcal{D}$ is maximized, leading to the objective function:

$$\arg\max_{\mathbf{v_x},\mathbf{v_c}} \sum_{(w,c)\in\mathcal{D}} \log \frac{1}{1+\exp(-\mathbf{v_c}\cdot\mathbf{v_x})}, \tag{2.3}$$

which can be trained using stochastic-gradient updates [64]. We thus expect the syntactic contexts to yield more focused embeddings, capturing more functional and less topical similarity.

3 Ontology Induction for Knowledge Acquisition

When building a dialogue system, a domain-specific knowledge base is required. To acquire the domain knowledge, the chapter focuses on automatically extracting the domain-specific concepts that can be used for SDSs.

3.1 Introduction

The distributional view of semantics hypothesizes that words occurring in the same contexts may have similar meanings [47]. Recently, with the advance of deep learning techniques, the continuous representations as word embeddings have further boosted the state-of-the-art results in many applications. Frame semantics, on the other hand, is a linguistic theory that defines meaning as a coherent structure of related concepts [37]. Although there has been some successful applications in natural language processing (NLP) [51, 26, 48], the Frame semantics theory has not been explored in the speech community.

The section focuses on using probabilistic frame-semantic parsing to automatically induce and adapt the semantic ontology for designing SDSs in an unsupervised fashion [20], alleviating some of the challenging problems for developing and maintaining SLU-based interactive systems [96]. Comparing to the traditional approach where domain experts and developers manually define the semantic ontology for SDS, the proposed approach has the advantages to reduce the costs of annotation, avoid human induced bias, and lower the maintenance costs [20].

Given unlabeled raw audio files, we investigate an unsupervised approach for automatic induction of semantic slots, the basic semantic units used in SDSs. To do this, we use a state-of-the-art probabilistic frame-semantic parsing approach [27], and perform an unsupervised approach to adapt, rerank, and map the generic FrameNet-style semantic parses to the target semantic space that is suitable for the domain-specific conversation settings [3]. We utilize continuous word embeddings trained on very large external corpora (e.g. Google News) to improve the adaptation process. To evaluate the performance of our approach, we compare the automatically induced semantic slots with the reference slots created by domain experts. Empirical experiments show that the slot creation results generated by our approach align

well with those of domain experts. Our main contributions of this paper are three-fold:

- We exploit continuous-valued word embeddings for unsupervised SLU;
- We propose the first approach of combining distributional and frame semantics for inducing semantic ontology from unlabeled speech data;
- We show that this synergized method yields the state-of-the-art performance.

3.2 Related Work

The idea of leveraging external semantic resources for unsupervised SLU was popularized by the work of Heck and Hakkani-Tür, and Tur *et al.* [49, 91]. The former exploited Semantic Web for the intent detection problem in SLU, showing that the results obtained from the unsupervised training process align well with the performance of traditional supervised learning [49]. The latter used search queries and obtained promising results on the slot filling task in the movie domain [91]. Following the success of the above applications, recent studies have also obtained interesting results on the tasks of relation detection [45], entity extraction [95], and extending domain coverage [35]. The major difference between our work and previous studies is that, instead of leveraging the discrete representations of Bing search queries or Semantic Web, we build our model on top of the recent success of deep learning—we utilize the continuous-valued word embeddings trained on Google News to induce semantic ontology for task-oriented SDS.

Our approach is clearly relevant to recent studies on deep learning for SLU. Tur *et al.* have shown that deep convex networks are effective for building better semantic utterance classification systems [90]. Following their success, Deng *et al.* have further demonstrated the effectiveness of applying the kernel trick to build better deep convex networks for SLU [33]. To the best of our knowledge, our work is the first study that combines the distributional view of meaning from the deep learning community, and the linguistic frame semantic view for improved SLU.

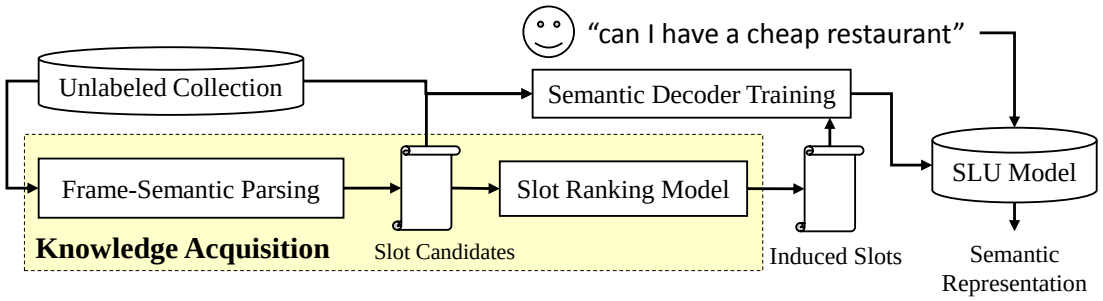


Figure 3.1: The proposed framework for ontology induction

3.3 The Proposed Framework

The main motivation of the work is to use a FrameNet-trained statistical probabilistic semantic parser to generate initial frame-semantic parses from ASR decodings of the raw audio conversation files. Then adapt the FrameNet-style frame-semantic parses to the semantic slots in the target semantic space, so that they can be used practically in the SDSs. The semantic mapping and adaptation problem are formulated as a ranking problem, where the domain-specific slots should be ranked higher than the generic ones. This thesis proposes the use of unsupervised clustering methods to differentiate the generic semantic concepts from target semantic space for task-oriented dialogue systems [20]. Also, considering that only using the small in-domain conversations as the training data may not robust enough, and the performance would be easily influenced by the data, this thesis proposes a radical extension: we aim at improving the semantic adaptation process by leveraging distributed word representations trained on very large external datasets [72, 71].

3.3.1 Probabilistic Semantic Parsing

In our approach, we parse all ASR-decoded utterances in our corpus using SEMAFOR introduced in Section 2.3 and 2.4 of Chapter 2, a state-of-the-art semantic parser for frame-semantic parsing [27, 28], and extract all frames from semantic parsing results as slot candidates, where the LUs that correspond to the frames are extracted for slot filling. For example, Figure 3.2 shows an example of an ASR-decoded text output parsed by SEMAFOR. SEMAFOR generates three frames (capability, expensiveness, and locale_by_use) for the utterance, which we consider as slot candidates. Note that for each slot candidate, SEMAFOR also includes the corresponding lexical unit (*can i*, *cheap*, and *restaurant*), which we consider as possible slot-fillers.

Since SEMAFOR was trained on FrameNet annotation, which has a more generic frame-semantic context, not all the frames from the parsing results can be used as the actual

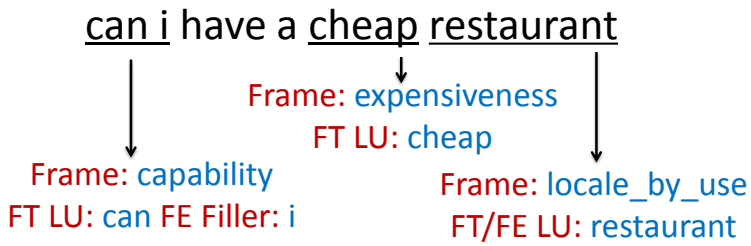


Figure 3.2: An example of probabilistic frame-semantic parsing on ASR output. FT: frame target. FE: frame element. LU: lexical unit.

slots in the domain-specific dialogue systems. For instance, in Figure 3.2, we see that the frames "expensiveness" and "locale_by_use" are essentially the key slots for the purpose of understanding in the restaurant query domain, whereas the "capability" frame does not convey particular valuable information for SLU. In order to fix this issue, we compute the prominence of these slot candidates, use a slot ranking model to rank the most important slots, and then generate a list of induced slots for use in domain-specific dialogue systems.

3.3.2 Independent Semantic Decoder

With outputted semantic parses, we extract the frames with the top 50 highest frequency as our slot candidates for training SLU. The features for training are generated by word confusion network, where confusion network features are shown to be useful in developing more robust systems for SLU [43]. We build a vector representation of an utterance as $\mathbf{u} = [x_1, ..., x_j, ...]$.

$$x_j = \mathbb{E}[C_u(n\text{-gram}_j)]^{1/|n\text{-gram}_j|}, \tag{3.1}$$

where $C_u(n\text{-gram}_j)$ counts how many times $n\text{-gram}_j$ occurs in the utterance u, $\mathbb{E}(C_u(n\text{-gram}_j))$ is the expected frequency of $n\text{-gram}_j$ in u, and $|n\text{-gram}_j|$ is the number of words in $n\text{-gram}_j$.

For each slot candidate s_i, we generate a pseudo training data $\mathcal{D}^i$ to train a binary classifier $\mathcal{M}^i$ for predicting the existence of s_i given an utterance, $\mathcal{D}^i = \{(\mathbf{u_k}, l_k^i) \mid \mathbf{u_k} \in \mathbb{R}^+, l_k^i \in \{-1, +1\}\}_{k=1}^K$, where $l_k^i = +1$ when the utterance u_k contains the slot candidate s_i in its semantic parse, $l_k^i = -1$ otherwise, and K is the number of utterances.

3.3.3 Adaptation Process and SLU Model

The generic concepts should be distinguished from the domain-specific concepts in the adaptation process. With the trained independent semantic decoders for all slot candidates, adaptation process computes the prominence of slot candidates for ranking and then selects a list of induced slots associated with their corresponding semantic decoders for use in domain-specific

dialogue systems. Then with each induced slot s_i and its corresponding trained semantic decoder $\mathcal{M}^i$, an SLU model can be built to predict whether the semantic slot occurs in the given utterance in a fully unsupervised way. In other words, the SLU model is able to transform the testing utterance into semantic representations without human involvement. The detail of the adaptation is described in the following section.

3.4 Slot Ranking Model

The purpose of the ranking model is to distinguish between generic semantic concepts and domain-specific concepts that are relevant to an SDS. To induce meaningful slots for the purpose of SDS, we compute the prominence of the slot candidates using a slot ranking model described below.

With the semantic parses from SEMAFOR, the model ranks the slot candidates by integrating two scores [20, 22]: (1) the normalized frequency of each slot candidate in the corpus, since slots with higher frequency may be more important. (2) the coherence of slot-fillers corresponding to the slot. Assuming that domain-specific concepts focus on fewer topics, the coherence of the corresponding slot-fillers can help measure the prominence of the slots because they are similar to each other.

$$w(s) = (1 - \alpha) \cdot \log f(s) + \alpha \cdot \log h(s), \tag{3.2}$$

where $w(s)$ is the ranking weight for the slot candidate s, $f(s)$ is its normalized frequency from semantic parsing, $h(s)$ is its coherence measure, and α is the weighting parameter within the interval $[0, 1]$, which balances the frequency and coherence.

For each slot s, we have the set of corresponding slot-fillers, $V(s)$, constructed from the utterances including the slot s in the parsing results. The coherence measure of the slot s, $h(s)$, is computed as the average pair-wise similarity of slot-fillers to evaluate if slot s corresponds to centralized or scattered topics.

$$h(s) = \frac{\sum_{x_a, x_b \in V(s)} \mathrm{Sim}(x_a, x_b)}{|V(s)|^2}, \tag{3.3}$$

where $V(s)$ is the set of slot-fillers corresponding slot s, $|V(s)|$ is the size of the set, and $\mathrm{Sim}(x_a, x_b)$ is the similarity between the slot-filler pair x_a and x_b. The slot s with higher $h(s)$ usually focuses on fewer topics, which is more specific and more likely to be a slot for the dialogue system. The detail about similarity measure is introduced in the following section.

3.5 Word Representations for Similarity Measure

To capture the semantics of each word, we transform each token x into a corresponding vector $\mathbf{x}$ by following methods. Then given that word representations can capture the semantic meanings, the topical similarity between each slot-filler pair x_a and x_b can be computed as

$$\text{Sim}(x_a, x_b) = \frac{\mathbf{x_a} \cdot \mathbf{x_b}}{\|\mathbf{x_a}\|\|\mathbf{x_b}\|}. \tag{3.4}$$

We assume that words occurring in similar domains have similar word representations, and thus $\text{Sim}(x_a, x_b)$ will be larger when x_a and x_b are semantically related. To build the word representations, we consider two techniques, in-domain clustering vectors and external word vectors.

3.5.1 In-Domain Clustering Vectors

The in-domain data is used to cluster words using Brown hierarchical word clustering algorithm [14, 67]. For each word x, we construct a vector $\mathbf{x} = [c_1, c_2, ..., c_K]$, where $c_i = 1$ when the word x is clustered into the i-th cluster, and $c_i = 0$ otherwise, and K is the number of clusters. The assumption is that topically similar words may be clustered together since they occur with the same contexts more frequently. Therefore, the cluster-based vectors that carry the such information can help measure similarity between words.

3.5.2 External Word Vectors

Section 2.5 of Chapter 2 introduces the distributional semantics, and a lot of studies have utilized the semantically-rich continuous word representations to benefit many NLP tasks. Considering that this distributional semantic theory may benefit our SLU task, we leverage word representations trained from large external data to differentiate semantic concepts. The rationale behind applying the distributional semantic theory to our task is straight-forward: because spoken language is a very distinct genre comparing to the written language on which FrameNet is constructed, it is necessary to borrow external word representations to help bridge these two data sources for the unsupervised adaptation process. More specifically, to better adapt the FrameNet-style parses to the target task-oriented SDS domain, we make use of continuous word vectors derived from a recurrent neural network architecture [69]. The learned word embeddings are able to capture both syntactic and semantic relations [72, 71], which provide more robust relatedness information between words and may help distinguish the domain-specific information from the generic concepts.

Considering that continuous space word representations may capture more robust topical information [72], we leverage word embeddings trained on an external large dataset to involve distributional semantics of slot-fillers. That is, the word vectors are built as their word embeddings, and the learning process is introduced in Section 2.5 of Chapter 2. The external word vectors rely on the performance of pre-trained word representations, and higher dimensionality of embedding words results in more accurate performance but greater complexity.

3.6 Experiments

To evaluate the effectiveness of our induced slots, we performed two evaluations. First, we examine the slot induction accuracy by comparing the ranked list of frame-semantic parsing induced slots with the reference slots created by developers of the corresponding system [101]. Secondly, based on the ranked list of induced slots, we can train a semantic decoder for each slot to build an SLU component, and then evaluate the performance of our SLU model by comparing against the human annotated semantic forms. For the experiments, we evaluate both on ASR transcripts of the raw audio, and on the manual transcripts.

3.6.1 Experimental Setup

In this experiment, we used the Cambridge University SLU corpus, previously used on several other SLU tasks [52, 19]. The domain of the corpus is about restaurant recommendation in Cambridge; subjects were asked to interact with multiple SDSs in an in-car setting. There were multiple recording settings: 1) a stopped car with the air condition control on and off; 2) a driving condition; and 3) in a car simulator. The distribution of each condition in this corpus is uniform. The corpus contains a total number of 2,166 dialogues, and 15,453 utterances, which is separated into training and testing parts as shown in Table 3.1. The training part is for self-training the SLU model.

The data is gender-balanced, with slightly more native than non-native speakers. The vocabulary size is 1,868. An ASR system was used to transcribe the speech; the word error rate was reported as 37%. There are 10 slots created by domain experts: addr, area, food, name, phone, postcode, price range, signature, task, and type. The parameter α in (3.2) can be empirically set; we use $\alpha = 0.2, N = 100$ for all experiments.

To include distributional semantics information, we use the distributed vectors trained on 10^9 words from Google News[1]. Training was performed using the continuous bag of words architecture, which predicts the current word based on the context, with sub-sampling using

[1] https://code.google.com/p/word2vec/

	Train	Test	Total
Dialogue	1522	644	2166
Utterance	10571	4882	15453
Male : Female	28 : 31	15 : 15	43 : 46
Native : Non-Native	33 : 26	21 : 9	54 : 47
Avg. #Slot	0.959	0.952	0.957

Table 3.1: The statistics of training and testing corpora

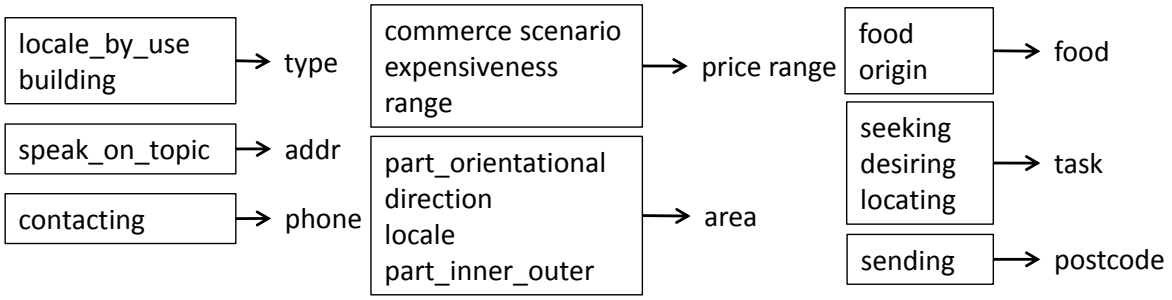


Figure 3.3: The mappings from induced slots (within blocks) to reference slots (right sides of arrows).

threshold $1 \times e^{-5}$, and with negative sampling with 3 negative examples per each positive one. The resulting vectors have dimensionality 300, vocabulary size is 3×10^6; the entities contain both words and automatically derived phrases. The dataset provides a larger vocabulary and better coverage.

3.6.2 Evaluation Metrics

To eliminate the influence of threshold selection when choosing induced slots, in the following metrics, we take the whole ranking list into account and evaluate the performance by the metrics that are independent on the selected threshold.

3.6.2.1 Slot Induction

To evaluate the accuracy of the induced slots, we measure their quality as the proximity between induced slots and reference slots. Figure 3.3 shows the mappings that indicate semantically related induced slots and reference slots [20]. For example, "expensiveness → price", "food → food", and "direction → area" show that these induced slots can be mapped into the reference slots defined by experts and carry important semantics in the target domain for developing the task-oriented SDS. Note that two slots, name and signature, do not have proper mappings, because they are too specific on restaurant-related domain, where name records the name of restaurant and signature refers to signature dishes. This means that

Approach	ASR				Manual			
	Slot Induction		SLU Model		Slot Induction		SLU Model	
	AP	AUC	WAP	AF	AP	AUC	WAP	AF
Frequency ($\alpha = 0$)	56.69	54.67	35.82	43.28	53.01	50.80	36.78	44.20
In-Domain	60.06	58.02	34.39	43.28	59.96	57.89	39.84	44.99
External	**71.70**	**70.35**	**44.51**	**45.24**	**74.41**	**73.57**	**50.48**	**73.57**
Max RI (%)	+26.5	+28.7	+24.3	+4.5	+40.4	+44.8	+37.2	+66.4

Table 3.2: The performance of slot induction and SLU modeling (%)

the 80% recall is achieved by our approach because we consider all outputted frames as slot candidates.

Since we define the adaptation task as a ranking problem, with a ranked list of induced slots and their associated scores, we can use the standard average precision (AP) as our metric, where the induced slot is counted as correct when it has a mapping to a reference slot. For a ranked list of induced slots $l = s^1, ..., s^k, ...$, where the s^k is the induced slot ranked at k-th position, the average precision is

$$\text{AP}(l) = \frac{\sum_{k=1}^{n} P(k) \times \mathbb{1}[s^k \text{ has a mapping to a reference slot}]}{\text{number of induced slots with mapping}}, \tag{3.5}$$

where $P(k)$ is the precision at cut-off k in the list and $\mathbb{1}$ is an indicator function equaling 1 if ranked k-th induced slot s^k has a mapping to a reference slot, 0 otherwise. Since the slots generated by our method cover only 80% of the referenced slots, the oracle recall is 80%. Therefore, average precision is a proper way to measure the slot ranking problem, which is also an approximation of the area under the precision-recall curve (AUC) [12].

3.6.2.2 SLU Model

While semantic slot induction is essential for providing semantic categories and imposing semantic constraints, we are also interested in understanding the performance of our unsupervised SLU models. For each induced slot with the mapping to a reference slot, we can compute an F-measure of the corresponding semantic decoder, and weight the average precision with corresponding F-measure as weighted average precision (WAP) to evaluate the performance of slot induction and SLU tasks together. The metric scores the ranking result higher if the induced slots corresponding to better semantic decoders are ranked higher. Another metric is the average F-measure (AF), which is the average micro-F of SLU models at all cut-off positions in the ranked list. Compared to WAP, AF additionally considers the slot popularity in the dataset.

3.6.3 Evaluation Results

Table 3.2 shows the results. The first row is the baseline, which only considers the frequency of slot candidates for ranking. It is found that the performance of SLU induction for ASR is better than for manual results. The reason about better AP and AUC scores of ASR may be that users tend to speak keywords clearer than generic words, higher word error rate of generic words makes these slot candidates ranked lower due to lower frequency.

In-Domain and External are the results of proposed word vector models with leveraging distributional word representations, in-domain clustering vectors and external word vectors respectively. In terms of both slot induction and SLU modeling, we find most results are improved by including distributed word information. With only in-domain data, the performance of slot induction can be significantly improved, from 57% to 60% on AP and from 55% to 58% on AUC. However, for SLU models, in-domain clustering approach does not show the improvement on ASR transcripts and improves the performance on manual results a little. With the external word vector approach, the performance is significantly improved for ASR and manual transcripts, which shows the effectiveness of involving external data for the similarity measurement.

To compare different similarity measures, we evaluate two approaches of computing distributional semantic similarity: in-domain clustering vectors and external word vectors. For both ASR and manual transcripts, the similarity derived from external word vectors significantly outperforms one from in-domain clustering vectors. The reason may be that external word vectors have more accurate semantic representations to measure similarity because they are trained on the large data, while in-domain clustering vectors rely on a small training set, which may be biased by the data and degrade with recognition errors.

We see that leveraging distributional semantics with frame-semantic parsing produces promising slot ranking performance; this demonstrates the effectiveness of our proposed approaches for slot induction. The 72% of AP indicates that our proposed approach can generate good coverage for domain-specific slots in a real-world SDS, reducing labor cost of system development.

3.7 Summary

This chapter proposes the first unsupervised approach unifying distributional and frame semantics for domain ontology induction. Our work makes use of a state-of-the-art semantic parser, and adapts the generic linguistic FrameNet representation to a semantic space characteristic of a domain-specific SDS. With the incorporation of distributional word repre-

sentations, we show that our automatically induced semantic slots align well with reference slots, yielding the state-of-the-art performance. Also, we demonstrate that it is feasible that the automatically induced ontology benefit SLU tasks. The automating process of ontology induction reduces the cost of human annotations, speeding up the SDS development.

4 Structure Learning for Knowledge Acquisition

Ontology induction extracts the domain-specific concepts, but the induced information is flat and unstructured. Considering that a well-organized ontology may help understanding, inter-slot relations should be considered for organizing the domain knowledge. The structure learning approach is introduced in this chapter.

4.1 Introduction

An important requirement for building a successful SDS is to define a coherent slot set and the corresponding slot-fillers for the SLU component. Unfortunately, since the semantic slots are often mutually-related, it is non-trivial for domain experts and professional annotators to design a such slot set for semantic representation of SLU.

Considering the restaurant domain [52], "*restaurant*" is the target slot, and important adjective modifiers such as "*Asian*" (the restaurant type) and "*cheap*" (the price of the restaurant) should be included in the slot set, so that the semantic representation of SLU can be more coherent and complete. In this case, it is challenging to design such a coherent and complete slot set manually, while considering various lexical, syntactic, and semantic dependencies.

Instead of considering slots independently, this chapter takes a data-driven approach to model word-to-word relations via syntactic dependency and further infer slot-to-slot relations. To do this, we incorporate the typed dependency grammar theory in a state-of-the-art frame-semantic driven unsupervised slot induction framework [30, 20]. In particular, we build two knowledge graphs: a slot-based semantic knowledge graph and a word-based lexical knowledge graph. Using typed dependency triples, we then study the stochastic relations between slots and words, using a mutually-reinforced random walk inference procedure to combine the two knowledge graphs. In evaluations, we use the jointly learned inter-slot relations to induce a coherent slot set in an unsupervised fashion. Our contributions in this chapter are three-fold:

- We are among the first to combine semantic and lexical knowledge graphs for unsupervised SLU;

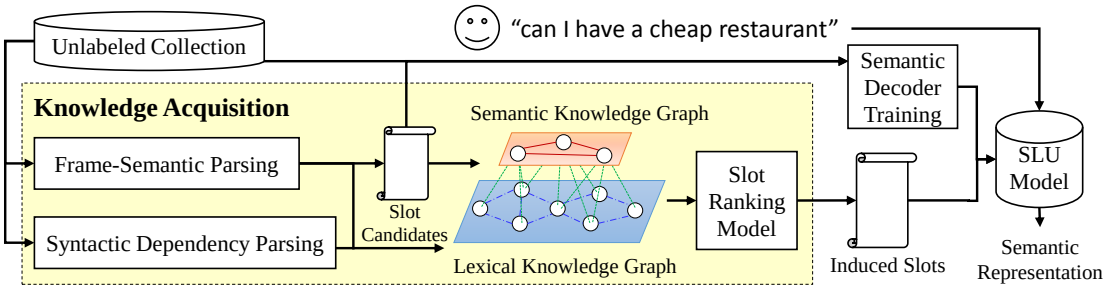


Figure 4.1: The proposed framework

- We propose a novel typed syntactic dependency grammar driven random walk model for relation discovery;
- Our experimental results suggest that jointly considering inter-slot relations helps obtain a more coherent and complete semantic slot set, showing that the ontology structure is essential to build a better SLU component.

4.2 Related Work

With the recent success of commercial SDSs and personal assistants (e.g., Microsoft's Cortana[1], Google Now[2], Apple's Siri[3], and Amazon's Echo[4]), a key focus on developing SLU techniques is the scalability issue. From the knowledge management perspective, empowering the dialogue system with large knowledge base is of crucial significance to modern SDSs. On this end, our work clearly aligns with recent studies on leveraging semantic knowledge graphs for SLU [50, 45, 46, 35, 21]. While leveraging external knowledge is the trend, efficient inference algorithms, such as random walks, are still less-studied for direct inference on knowledge graphs of the spoken contents.

In the NLP literature, Lao *et al.* used a random walk algorithm to construct inference rules on large entity-based knowledge bases [59], and leveraged syntactic information for reading the Web [60]. Even though this work has important contributions, the proposed algorithm cannot learn mutually-recursive relations, and does not to consider lexical items—in fact, more and more studies show that, in addition to semantic knowledge graphs, lexical knowledge graphs that model surface-level natural language realization [54, 87, 66], multiword expressions, and context [65], are also critical for short text understanding [87, 94].

[1] `http://www.windowsphone.com/en-us/how-to/wp8/cortana`
[2] `http://www.google.com/landing/now`
[3] `http://www.apple.com/ios/siri`
[4] `http://www.amazon.com/oc/echo`

From the engineering perspective, quick and easy development turnaround time for domain-specific dialogue applications is also critical [18]. However, most works treat each slot independently and have not considered the inter-slot relations when inducing the semantic slots. Considering that a well-organized ontology may benefit a better SLU component construction and the system development, the semantic structure of the ontology is included when inducing the domain knowledge. To the best of our knowledge, we are the first to use syntactically-informed random walk algorithms to combine the semantic and lexical knowledge graphs, and not individually but globally inducing the semantic slots for building better unsupervised SLU components.

4.3 The Proposed Framework

The approach is built on top of the success of an unsupervised frame-semantic parsing approach introduced in Chapter 3 [20]. The main motivation is to use a FrameNet-trained statistical probabilistic semantic parser to generate initial frame-semantic parses from ASR decodings of the raw audio conversation files, and then adapt the FrameNet-style frames to the semantic slots in the target semantic space, so that they can be used practically in the SDSs. In stead of inducing an unstructured ontology, this chapter improves the adaptation process by leveraging distributed word embeddings associated with typed syntactic dependencies between words to infer inter-slot relations in order to learn a well-organized ontology [71, 72, 64]. The proposed framework is shown in Figure 4.1. Frame-semantic parsing, independent semantic decoder, and adaptation process are similar to one from Chapter 3, except that the slot ranking model does consider the relation information. Finally we can build the SLU models based on the learned semantic decoders and induced slots to evaluate whether the ontology structure helps SLU modeling.

4.4 Slot Ranking Model

The purpose of the ranking model is to distinguish between generic semantic concepts and domain-specific concepts that are relevant to an SDS. To induce meaningful slots for the purpose of SDS, we compute the prominence of the slot candidates additionally considering the structure information.

With the semantic parses from SEMAFOR, where each frame is viewed independently, so inter-slot relations are not included, the model ranks the slot candidates by integrating two information: (1) the frequency of each slot candidate in the corpus, since slots with higher frequency may be more important. (2) the relations between slot candidates. Assuming that domain-specific concepts are usually related to each other, globally considering inter-slot

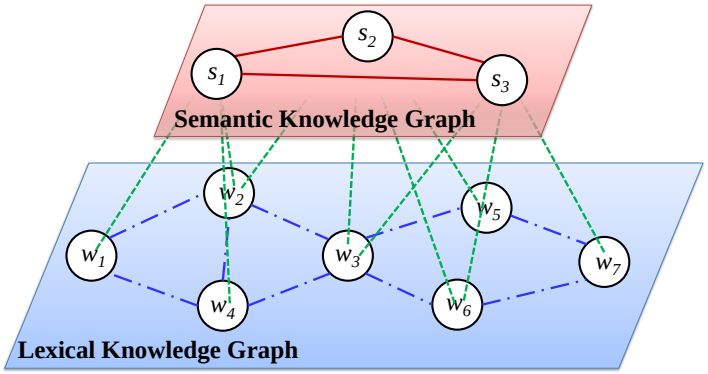


Figure 4.2: A simplified example of the two knowledge graphs, where a slot candidate s_i is represented as a node in a semantic knowledge graph and a word w_j is represented as a node in a lexical knowledge graph.

relations induces a more coherent slot set. Here for baseline scores, we only use the frequency of each slot candidate as its prominence without the structure information.

First we construct two knowledge graphs, one is a slot-based semantic knowledge graph and another is a word-based lexical knowledge graph, both of which encode the typed dependency relations in their edge weights. We also connect two graphs to model the relations between slot-filler pairs.

4.4.1 Knowledge Graphs

We construct two undirected graphs, semantic and lexical knowledge graphs. Each node in the semantic knowledge graph is a slot candidate s_i outputted by the frame-semantic parser, and each node in the lexical knowledge graph is a word w_j.

- **Slot-based semantic knowledge graph** is built as $G_s = \langle V_s, E_{ss} \rangle$, where $V_s = \{s_i\}$ and $E_{ss} = \{e_{ij} \mid s_i, s_j \in V_s\}$.
- **Word-based lexical knowledge graph** is built as $G_w = \langle V_w, E_{ww} \rangle$, where $V_w = \{w_i\}$ and $E_{ww} = \{e_{ij} \mid w_i, w_j \in V_w\}$.

With two knowledge graphs, we build the edges between slots and slot-fillers to integrate them as shown in Figure 4.2. Thus the combined graph can be formulated as $G = \langle V_s, V_w, E_{ss}, E_{ww}, E_{ws} \rangle$, where $E_{ws} = \{e_{ij} \mid w_i \in V_w, s_j \in V_s\}$. E_{ss}, E_{ww}, and E_{ws} correspond the slot-to-slot relations, word-to-word relations, and the word-to-slot relations respectively [16, 17].

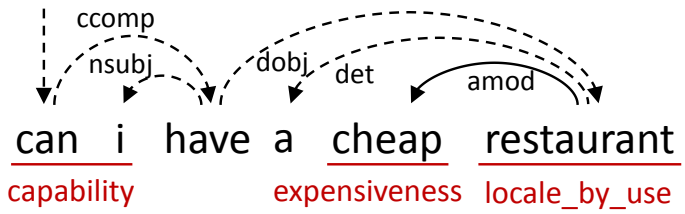


Figure 4.3: The dependency parsing result on an utterance.

4.4.2 Edge Weight Estimation

Considering the relations in the knowledge graphs, the edge weights for E_{ww} and E_{ss} are measured based on the dependency parsing results. The example utterance "*can i have a cheap restaurant*" and its dependency parsing result are illustrated in Figure 4.3. The arrows denote the dependency relations from headwords to their dependents, and words on arcs denote types of the dependencies. All typed dependencies between two words are encoded in triples and form a word-based dependency set $\mathcal{T}_w = \{\langle w_i, t, w_j \rangle\}$, where t is the typed dependency between the headword w_i and the dependent w_j. For example, Figure 4.3 generates $\langle$*restaurant*, AMOD, *cheap*$\rangle$, $\langle$*have*, DOBJ, *restaurant*$\rangle$, etc. for $\mathcal{T}_w$. Similarly, we build a slot-based dependency set $\mathcal{T}_s = \{\langle s_i, t, s_j \rangle\}$ by transforming dependencies between slot-fillers into ones between slots. For example, $\langle$*restaurant*, AMOD, *cheap*$\rangle$ from $\mathcal{T}_w$ is transformed into $\langle$locale_by_use, AMOD, expensiveness$\rangle$ for building $\mathcal{T}_s$, because both sides of the non-dotted line are parsed as slot-fillers by SEMAFOR.

For the edges within a single knowledge graph, we assign a weight of the edge connecting nodes x_i and x_j as $\hat{r}(x_i, x_j)$, where x is either s or w. Since the weights are measured based on the relations between nodes regardless of the directions, we combine the scores of two directional dependencies:

$$\hat{r}(x_i, x_j) = r(x_i \rightarrow x_j) + r(x_j \rightarrow x_i), \tag{4.1}$$

where $r(x_i \rightarrow x_j)$ is the score estimating the dependency including x_i as a head and x_j as a dependent. In Section 4.4.2.1 and 4.4.2.2, we propose two scoring functions for $r(\cdot)$, frequency-based as $r_1(\cdot)$ and embedding-based as $r_2(\cdot)$ respectively.

For the edges in E_{ws}, we estimate the edge weights based on the frequency that the slot candidates and the words are parsed as slot-filler pairs. In other words, the edge weight between the slot-filler w_i and the slot candidate s_j, $\hat{r}(w_i, s_j)$, is equal to how many times the filler w_i corresponds to the slot candidate s_j in the parsing results.

	Typed Dependency Relation	Target Word	Contexts
Word	$\langle$*restaurant*, AMOD, *cheap*$\rangle$	*restaurant* *cheap*	*cheap*/AMOD *restaurant*/AMOD^{-1}
Slot	$\langle$locale_by_use, AMOD, expensiveness$\rangle$	locale_by_use expansiveness	expensiveness/AMOD locale_by_use/AMOD^{-1}

Table 4.1: The contexts extracted for training dependency-based word/slot embeddings from the utterance of Fig. 3.2.

4.4.2.1 Frequency-Based Measurement

Based on the dependency set $\mathcal{T}_x$, we use $t^*_{x_i \to x_j}$ to denote the most frequent typed dependency with x_i as a head and x_j as a dependent.

$$t^*_{x_i \to x_j} = \arg\max_t C(x_i \underset{t}{\to} x_j), \tag{4.2}$$

where $C(x_i \underset{t}{\to} x_j)$ counts how many times the dependency $\langle x_i, t, x_j \rangle$ occurs in the dependency set $\mathcal{T}_x$.

Then the scoring function that estimates the dependency $x_i \to x_j$ is measured as

$$r_1(x_i \to x_j) = C(x_i \xrightarrow[t^*_{x_i \to x_j}]{} x_j), \tag{4.3}$$

which equals to the highest observed frequency of the dependency $x_i \to x_j$ among all types from $\mathcal{T}_x$.

4.4.2.2 Embedding-Based Measurement

It is shown that a dependency-based embedding approach introduced in Section 2.5.2 of Chapter 2 is able to capture more functional similarity because using dependency-based syntactic contexts for training word embeddings [64]. Table 4.1 shows some extracted dependency-based contexts for each target word from the example in Figure 4.3, where headwords and their dependents can form the contexts by following the arc on a word in the dependency tree, and -1 denotes the directionality of the dependency. We learn vector representations for both words and contexts such that the dot product $\mathbf{v_w} \cdot \mathbf{v_c}$ associated with "good" word-context pairs belonging to the training data is maximized. Then we can obtain the dependency-based slot and word embeddings using $\mathcal{T}_s$ and $\mathcal{T}_w$ respectively.

With trained dependency-based embeddings, we estimate the probability that x_i is the head-

word and x_j is its dependent via the typed dependency t as

$$P(x_i \underset{t}{\rightarrow} x_j) = \frac{Sim(x_i, x_j/t) + Sim(x_j, x_i/t^{-1})}{2}, \tag{4.4}$$

where $Sim(x_i, x_j/t)$ is the cosine similarity between word/slot embeddings $\mathbf{v_{x_i}}$ and context embeddings $\mathbf{v_{x_j/t}}$ after normalizing to $[0, 1]$. Then we can measure the scoring function $r_2(\cdot)$ as

$$r_2(x_i \rightarrow x_j) = C(x_i \xrightarrow[t^*_{x_i \rightarrow x_j}]{} x_j) \cdot P(x_i \xrightarrow[t^*_{x_i \rightarrow x_j}]{} x_j), \tag{4.5}$$

which is similar to (4.3) but additionally weighted by the estimated probability. The estimated probability smooths the observed frequency to avoid overfitting due to the smaller dataset.

4.4.3 Random Walk Algorithm

We first compute $L_{ww} = [\hat{r}(w_i, w_j)]_{|V_w| \times |V_w|}$ and $L_{ss} = [\hat{r}(s_i, s_j)]_{|V_s| \times |V_s|}$, where $\hat{r}(w_i, w_j)$ and $\hat{r}(s_i, s_j)$ are either from frequency-based ($r_1(\cdot)$) or embedding-based measurements ($r_2(\cdot)$). Similarly, $L_{ws} = [\hat{r}(w_i, s_j)]_{|V_w| \times |V_s|}$ and $L_{sw} = [\hat{r}(w_i, s_j)]^T_{|V_w| \times |V_s|}$, where $\hat{r}(w_i, s_j)$ is the frequency that s_j and w_i are a slot-filler pair computed in Section 4.4.2. Then we only keep the top N highest weights for each row in L_{ww} and L_{ss} ($N = 10$), which means that we filter out the edges with smaller weights within the single knowledge graph. Column-normalization are performed for L_{ww}, L_{ss}, L_{ws}, L_{sw} [83]. They can be viewed as word-to-word, slot-to-slot, and word-to-slot relation matrices.

4.4.3.1 Single-Graph Random Walk

Here we run random walk only on the semantic knowledge graph to propagate the scores based on inter-slot relations through the edges E_{ss}.

$$R_s^{(t+1)} = (1 - \alpha) R_s^{(0)} + \alpha L_{ss} R_s^{(t)}, \tag{4.6}$$

where $R_s^{(t)}$ denotes the importance scores of the slot candidates V_s in t-th iteration. In the algorithm, the score is the interpolation of two scores, the normalized baseline importance of slot candidates ($R_s^{(0)}$), and the scores propagated from the neighboring nodes in the semantic knowledge graph based on the slot-to-slot relations L_{ss}. The algorithm will converge when $R_s^{(t+1)} = R_s^{(t)} = R_s^*$ and (4.7) can be satisfied.

$$R_s^* = (1 - \alpha) R_s^{(0)} + \alpha L_{ss} R_s^* \tag{4.7}$$

We can solve R_s^* as below.

$$R_s^* = \left((1-\alpha)R_s^{(0)}e^T + \alpha L_{ss}\right)R_s^* = M_1 R_s^*, \tag{4.8}$$

where the $e = [1, 1, ..., 1]^T$. It has been shown that the closed-form solution R_s^* of (4.8) is the dominant eigenvector of M_1 [58], or the eigenvector corresponding to the largest absolute eigenvalue of M_1. The solution of R_s^* denotes the updated importance scores for all utterances. Similar to the PageRank algorithm [13], the solution can also be obtained by iteratively updating $R_s^{(t)}$.

4.4.3.2 Double-Graph Random Walk

Here we borrow the idea from two-layer mutually reinforced random walk to propagate the scores based on not only internal importance propagation within the same graph but external mutual reinforcement between different knowledge graphs [16, 17].

$$\begin{cases} R_s^{(t+1)} = (1-\alpha)R_s^{(0)} + \alpha L_{ss}L_{sw}R_w^{(t)} \\ R_w^{(t+1)} = (1-\alpha)R_w^{(0)} + \alpha L_{ww}L_{ws}R_s^{(t)} \end{cases} \tag{4.9}$$

In the algorithm, they are the interpolations of two scores, the normalized baseline importance ($R_s^{(0)}$ and $R_w^{(0)}$) and the scores propagated from another graph. For the semantic knowledge graph, $L_{sw}R_w^{(t)}$ is the score from the word set weighted by slot-to-word relations, and then the scores are propagated based on slot-to-slot relations L_{ss}. Similarly, nodes of the lexical knowledge graph also include the scores propagated from the semantic knowledge graph. Then $R_s^{(t+1)}$ and $R_w^{(t+1)}$ can be mutually updated by the latter parts in (4.9) iteratively. When the algorithm converges, we have R_s^* and R_w^* can be derived similarly.

$$\begin{cases} R_s^* = (1-\alpha)R_s^{(0)} + \alpha L_{ss}L_{sw}R_w^* \\ R_w^* = (1-\alpha)R_w^{(0)} + \alpha L_{ww}L_{ws}R_s^* \end{cases} \tag{4.10}$$

$$\begin{aligned} R_s^* &= (1-\alpha)R_s^{(0)} + \alpha L_{ss}L_{sw}\left((1-\alpha)R_w^{(0)} + \alpha L_{ww}L_{ws}R_s^*\right) \\ &= (1-\alpha)R_s^{(0)} + \alpha(1-\alpha)L_{ss}L_{sw}R_w^{(0)} + \alpha^2 L_{ss}L_{sw}L_{ww}L_{ws}R_s^* \\ &= \left((1-\alpha)R_s^{(0)}e^T + \alpha(1-\alpha)L_{ss}L_{sw}R_w^{(0)}e^T + \alpha^2 L_{ss}L_{sw}L_{ww}L_{ws}\right)R_s^* \\ &= M_2 R_s^*. \end{aligned} \tag{4.11}$$

The closed-form solution R_s^* of (4.11) is the dominant eigenvector of M_2.

4.5 Experiments

The goal is to validate the effectiveness of including the structure information for SLU. We evaluate our approach in two ways. First, we examine the slot induction accuracy by comparing the ranked list of induced slots with the reference slots created by system developers [101]. Secondly, with the ranked list of induced slots and their associated semantic decoders, we can evaluate the SLU performance. For the experiments, we evaluate both on ASR transcripts of the raw audio, and on the manual transcripts.

4.5.1 Experimental Setup

The data used is Cambridge University SLU corpus described in previous chapter. For parameter setting, the damping factor for random walk α is empirically set as 0.9 for all experiments. For training the semantic decoders, we use SVM with a linear kernel to predict each semantic slot. We use Stanford Parser to obtain the collapsed typed syntactic dependencies [85] and set the dimensionality of embeddings $d = 300$ in all experiments.

4.5.2 Evaluation Metrics

To evaluate the accuracy of the induced slots, we measure their quality as the proximity between induced slots and reference slots. Figure 3.3 in Chapter 3 shows the mappings that indicate semantically related induced slots and reference slots [20]. As the same as the metrics in Chapter 3, we use standard average precision (AP) and the area under the precision-recall curve (AUC) for evaluating slot induction.

To evaluate the influence of SLU modeling brought by the proposed approaches, we use weighted average precision (WAP) to evaluate the performance of slot induction and SLU tasks together. Also, another metric is the average F-measure (AF), which is the average micro-F of SLU models at all cut-off positions in the ranked list. Compared to WAP, AF additionally considers the slot popularity in the dataset.

4.5.3 Evaluation Results

Table 4.2 shows the results on both ASR and transcripts. Rows (a) is the baseline only considering the frequency of each slot candidate for ranking. Rows (b) and (c) show performance after leveraging a semantic knowledge graph through random walk. Rows (d) and (e) are the results after combining two knowledge graphs. We find almost all results are improved by

Approach			ASR				Manual			
			Slot Induction		SLU Model		Slot Induction		SLU Model	
			AP	AUC	WAP	AF	AP	AUC	WAP	AF
(a)	Baseline ($\alpha = 0$)		56.69	54.67	35.82	43.28	53.01	50.80	36.78	44.20
(b)	Single	Freq.	63.88	62.05	41.67	47.38	63.02	61.10	43.76	48.53
(c)		Embed.	69.04	68.25	**46.29**	**48.89**	75.15	74.50	**54.50**	**50.86**
(d)	Double	Freq.	56.83	55.31	32.64	44.91	52.12	50.54	34.01	45.05
(e)		Embed.	**71.48**	**70.84**	44.06	47.91	**76.42**	**75.94**	52.89	50.40

Table 4.2: The performance of induced slots and corresponding SLU models (%)

additionally considering inter-slot relations in terms of single- and double-graph random walk for both ASR and manual transcripts.

4.5.3.1 Slot Induction

For both ASR and manual transcripts, almost all results outperform the baseline, which shows that the inter-slot relations significantly influence the performance of slot induction. The best performance is from the results of double-graph random walk with the embedding-based measurement, which integrate a semantic knowledge graph and a lexical knowledge graph together and jointly consider the slot-to-slot, word-to-word, and word-to-slot relations when scoring the prominence of slot candidates to generate a coherent slot set.

4.5.3.2 SLU Model

For both ASR and manual transcripts, almost all results outperform the baseline, which shows the practical usage for training dialogue systems. The best performance is from the results of single-graph random walk with embedding-based measurement, which only use the semantic knowledge graph to involve the inter-slot relations. The semantic knowledge graph is not as precise as the lexical one and may be influenced by the performance of the semantic parser more. Although the row (e) doesn't show better performance than the row (c), double-graph random walk may be more robust because it additionally includes the words' relations to avoid from only relying on the relations tied with the slot candidates.

4.5.4 Discussion and Analysis

4.5.4.1 Comparing Frequency- and Embedding-Based Measurements

Table 4.2 shows that all results with embedding-based measurement perform better than with frequency-based measurement. The frequency-based measurement also brings large improve-

ment for single-graph approaches, but does not for double-graph ones. The reason is probably that using observed frequencies in the lexical knowledge graph may result in overfitting issues due to the smaller dataset. Additionally including embedding information can smooth the edge weights and deal with data sparsity to improve the performance, especially for the lexical knowledge graph.

4.5.4.2 Comparing Single- and Double-Graph Approaches

Considering embedding-based measurement performs better, we only compare the results of single- and double-graph random walk using this measurement (rows (c) and (e)). It can be seen that the difference between them is not consistent in terms of slot induction and SLU model.

For evaluating slot induction (AP and AUC), the double-graph random walk (row (e)) performs better on both ASR and manual results, which implies that additionally integrating the lexical knowledge graph helps decide a more coherent and complete slot set since we can model the score propagation more precisely (not only slot-level but word-level information). However, for SLU evaluation (WAP and AF), the single-graph random walk (row (c)) performs better, which may imply that the slots carrying the coherent relations from the row (e) may not have good semantic decoder performance so that the performance is decreased a little. For example, double-graph random walk scores the slots local_by_use and expensiveness higher than the slot contacting, while the single-graph method ranks the latter higher. local_by_use and expensiveness are more important on this domain but contacting has very good performance of its semantic decoder, so the double-graph approach does not show the improvement when evaluating SLU. This allows us to try an improved method of jointly optimizing the slot coherence and SLU performance in the future.

4.5.4.3 Relation Discovery Analysis

To interpret the inter-slot relations, we output the relations that connect the slots with highest scores from the best results (row (e) in Table 4.2) in Table 4.3. It can be shown that the outputted inter-slot relations are reasonable and usually connect two important semantic slots. The automatically learned structure is able to construct a corresponding slot-based semantic knowledge graph as Figure 4.4. This proves that inter-slot relations help decide a coherent and complete slot set and enhance the interpretability of semantic slots, and from a practical perspective, developers are able to design the framework of dialogue systems more easily.

Rank	Relation
1	⟨locale_by_use, NN, food⟩
2	⟨food, AMOD, expensiveness⟩
3	⟨locale_by_use, AMOD, expensiveness⟩
4	⟨seeking, PREP_FOR, food⟩
5	⟨food, AMOD, relational_quantity⟩
6	⟨desiring, DOBJ, food⟩
7	⟨seeking, PREP_FOR, locale_by_use⟩
8	⟨food, DET, quantity⟩

Table 4.3: The top inter-slot relations learned from the training set of ASR outputs.

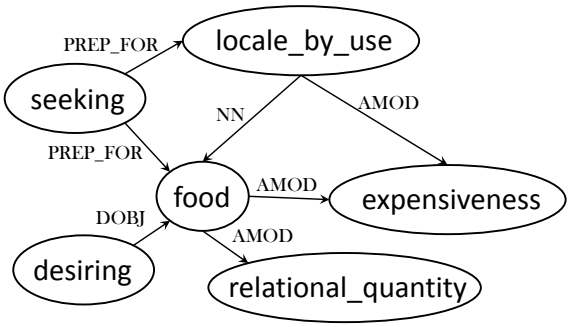


Figure 4.4: A simplified example of the automatically derived knowledge graph.

4.6 Summary

The chapter proposes an approach of jointly considering inter-slot relations for slot induction to output a more coherent slot set, where two knowledge graphs, a slot-based semantic knowledge graph and a word-based lexical knowledge graph, are built and combined by a random walk algorithm. The automatically induced slots carry coherent and interpretable relations and can be used for better understanding, showing that the relation information helps SLU modeling.

5 Surface Form Derivation for Knowledge Acquisition

With the available structured ontology, the domain knowledge can be learned for building a domain-specific dialogue system. However, the entities in the ontology have various surface forms, for example, in a movie domain, "*movie*" and "*film*" can be used to refer to the same entity of an ontology. This chapter focuses on deriving surface forms and shows that the derived surface forms can benefit the SLU modeling performance.

5.1 Introduction

An SLU component aims to detect the semantic frames that include domain-related information. Traditional SDSs are trained with annotated examples and support limited domains. Recently, the structured semantic knowledge such as Freebase[1] [9], FrameNet[2] [2], etc. is utilized to obtain domain-related knowledge and help SLU for tackling open domain problems in SDSs [49, 50, 46, 20, 22].

Knowledge graphs, such as Freebase, usually carry rich information for named entities, which is encoded in triples about entity pairs and their relation. Such information is usually used for interpretation of natural language in SDSs [53, 89, 46]. However, the entity lists/gazatteers may bring noise and ambiguity to SLU, for example, the commonly used words "*Show me*" and "*Up*" can be movie names, and "*Brad Pitt*" can be an actor name or a producer name, which makes interpretation more difficult. Some works focused on assigning weights for entities or entity types to involve prior background knowledge of entities, where the probabilistic confidences offer better cues for SLU [53, 46]. Also, many works focused on mining natural language forms based on the ontology by web search or query click logs, which benefit discovering new relation types from large text corpora [45, 44]. The mined data can also be used to help SLU by adaptation from the text domain to the spoken domain [50].

On the other hand, the distributional view of semantics hypothesizes that words occurring in the same contexts may have similar meanings, and words can be represented as high dimensional vectors. Recently, with the advancement of deep learning techniques, the continuous

[1] `http://www.freebase.com`
[2] `http://framenet.icsi.berkeley.edu`

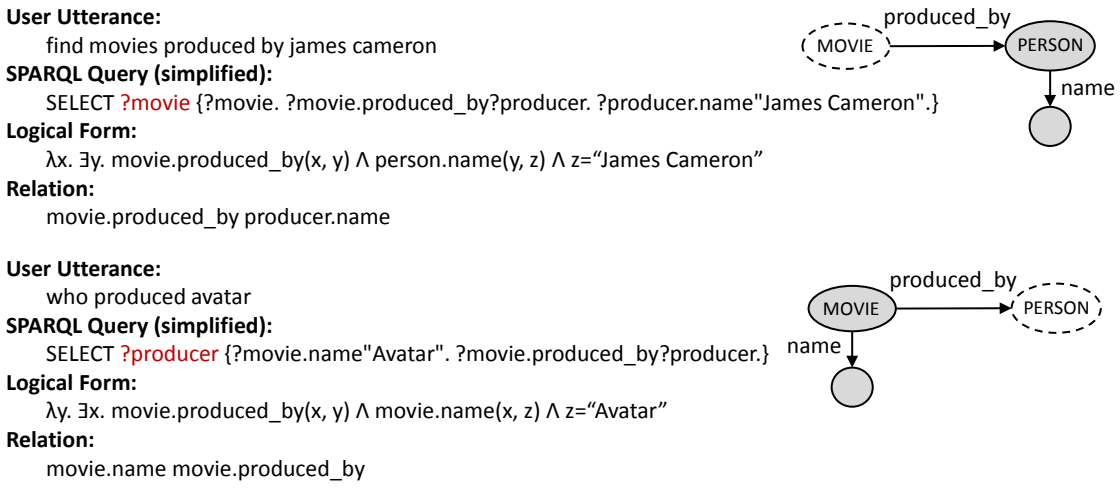


Figure 5.1: The relation detection examples.

word representations that represent words as dense vectors have been shown to perform well in many applications [72, 71, 70, 100, 10, 11]. Furthermore, dependency-based word embeddings were proposed to capture more functional similarity based on the dependency-based contexts instead of the linear contexts using the similar training procedure [64].

Following the successes, we leverage dependency-based entity embeddings to learn relational information including entity surface forms and entity contexts from the text data. Integrating derived relational information as local cues and gazetteers as background knowledge performs well for the relation detection task in a fully unsupervised fashion.

5.2 Knowledge Graph Relations

Given an utterance, we can form a set of relations that encode the user's intent for informational queries based on the semantic graph ontology. Figure 5.1 presents two user utterances and their invoked relations, which can be used to create requests in query languages (i.e., SPARQL Query Language for RDF[3]). The two examples in this figure include two nodes and the same relation `movie.produced_by`, and we differentiate these examples by including the originating node types in the relation (`movie.name`, `producer.name`) instead of just plain names (the nodes with gray color denote specified entities). Given all these, this task is richer than the regular relation detection task. The motivation to do that is, since we are trying to write queries to the knowledge source, we need to make sure that the queries are well-formed and relation arcs originate from the correct nodes in them. Therefore, this paper focuses on detecting not only `movie.produced_by` but also the specified entities `producer.name` and

[3] `http://www.w3.org/TR/ref-sparql-query/`

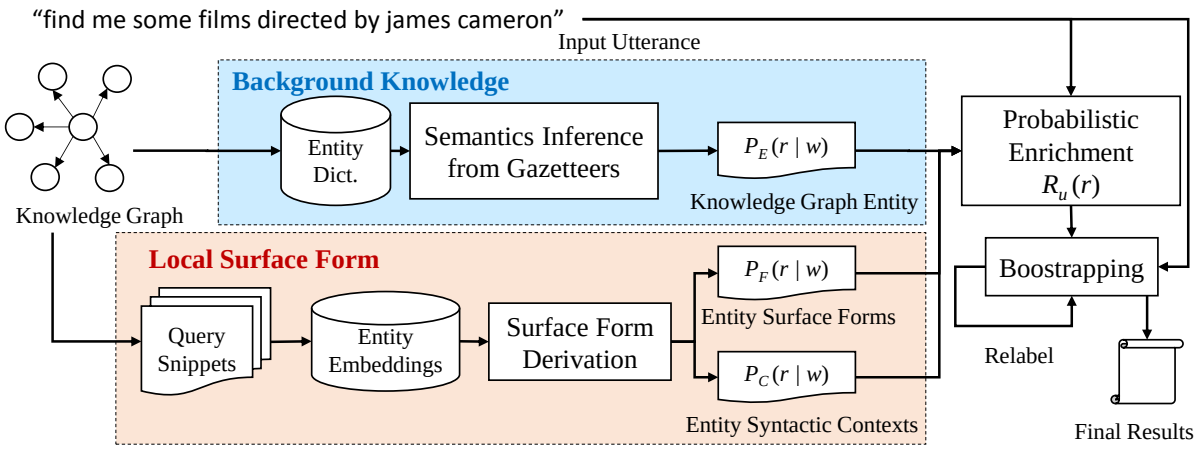


Figure 5.2: The proposed framework.

`movie.name`, so that we can obtain a better understanding of user utterances.

5.3 Proposed Framework

The whole system framework is shown in Fig 4.1. There are two major components: 1) we first utilize background knowledge as a prior to infer relations, and 2) we capture natural language surface forms for detecting local observations, which are described in Sections 5.4 and Section 5.5 respectively.

Then probabilistic enrichment is used to integrate probabilistic information from background knowledge and local relational observations given the input utterance. Finally an unsupervised learning approach is proposed to boost the performance. The detail is presented in Section 5.6.

5.4 Relation Inference from Gazetteers

Due to ambiguity of entity mentions, we utilize prior knowledge from gazetteers to estimate the probability distribution of associated relations for each entity [46]. For example, "*James Cameron*" can be a director or a producer, which infers `movie.directed_by` or `movie.produced_by` relations respectively. Given a word w_j, the estimated probability of inferred relation r_i is defined as

$$P_E(r_i \mid w_j) = P_E(t_i \mid w_j) = \frac{C(w_j, t_i)}{\sum_{t_k \in T(w_j)} C(w_j, t_k)}, \tag{5.1}$$

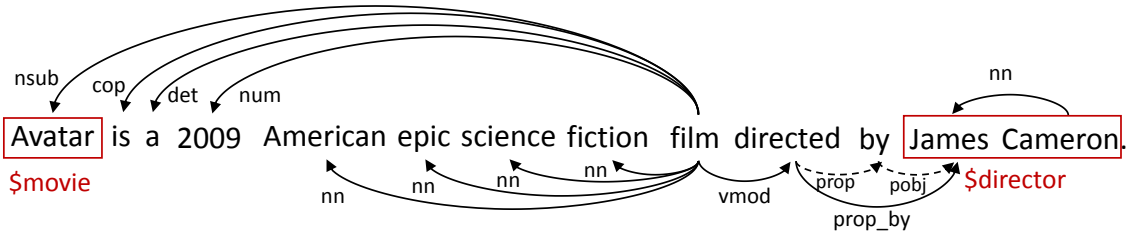


Figure 5.3: An example of dependency-based contexts.

where t_i is the type corresponding to the relation r_i (e.g. the entity type `director.name` infers the relation `movie.directed_by`), $T(w_j)$ denotes the set of all possible entity types of the word w_j, and $C(w_j, t_i)$ is the number of times the specific entity w_j is observed with a specific type t_i in the knowledge graph. For example, the number of movies James Cameron has directed.

5.5 Relational Surface Form Derivation

5.5.1 Web Resource Mining

Based on the ontology of knowledge graph, we extract all possible entity pairs that are connected with specific relations. Following the previous work, we get search snippets for entity pairs tied with specific relations by web search[4] [49, 45]. Then we mine the patterns used in natural language realization of the relations. With the query snippets, we use dependency relations to learn natural language surface forms about each specific relation by dependency-based entity embeddings introduced below.

5.5.2 Dependency-Based Entity Embeddings

As Section 2.5.2 of Chapter 2 introduces, the dependency-based embeddings contain more relational information because of training on the dependency-based contexts [64].

An example sentence "*Avatar is a 2009 American epic science fiction film directed by James Cameron.*" and its dependency parsing result are illustrated in Figure 5.3. Here the sentence comes from snippets returned by searching the entity pair, "*Avatar*" (movie) and "*James Cameron*" (director). The arrows denote the dependency relations from headwords to their dependents, and words on arcs denote type of the dependency relations. Relations that include a preposition are "collapsed" prior to context extraction (dashed arcs in Figure 5.3), by directly connecting the head and the object of the preposition, and subsuming the preposition

[4] `http://www.bing.com`

Word	Contexts
$movie	film/nsub^{-1}
is	film/cop^{-1}
a	film/det^{-1}
2009	film/num^{-1}
american, epic, science, fiction	film/nn^{-1}
film	avatar/nsub, is/cop, a/det, 2009/num, american/nn epic/nn, science/nn, fiction/nn, directed/vmod
directed	$director/prep_by
$diretor	directed/prep_by^{-1}

Table 5.1: The contexts extracted for training dependency entity embeddings in the example of the Figure 5.3.

itself into the dependency label. Before training embeddings, we replace entities with their entity tags such as *$movie* for "*Avatar*" and *$director* for "*James Cameron*".

The dependency-based contexts extracted from the example are given in Table 5.1, where headwords and their dependents can form the contexts by following the arc on a word in the dependency tree, and -1 denotes the directionality of the dependency. With the target words and associated dependency-based contexts, we can train dependency-based entity embeddings for all target words [100, 10, 11].

5.5.3 Surface Form Derivation

In addition to named entities detected by gazetteers, there are two different relational surface forms used in natural languages, entity surface forms and entity syntactic contexts, which are derived from trained embeddings by following approaches.

5.5.3.1 Entity Surface Forms

With only background knowledge gazetteers provided in Section 5.4, the unspecified entities cannot be captured because knowledge graph does not contain such information like the words "*film*" and "*director*". This procedure is to discover the words that play the same role and carry similar functional dependency as the specified entities. For example, the entity *$character* may derive the word "*role*", and *$movie* may derive "*film*", "*movie*" as their entity surface forms. The unspecified entities provide important cues for inferring corresponding relations.

We first define a set of entity tags $E = \{e_i\}$ and a set of words $W = \{w_j\}$. Based on the trained dependency-based entity embeddings, for each entity tag e_i, we compute the score of

the word w_j as

$$S_i^F(w_j) = \frac{\text{FormSim}(w_j, e_i)}{\sum_{e_k \in E} \text{FormSim}(w_j, e_k)}, \tag{5.2}$$

where $\text{FormSim}(w, e)$ is the cosine similarity between the embeddings of the word w and the entity tag e. $S_i^F(w_j)$ can be viewed as the normalized weights of the words indicating the importance for discriminating different entities. Based on $S_i^F(w_j)$, we propose to extract top N similar words for each entity tag e_i, to form a set of entity surface forms F_i, where F_i includes surface form candidates of entity e_i. The derived words may have similar embeddings as the target entity, for example, "*director*" and *$director* may encode the same context information such as *directed/prep_by*$^{-1}$ in their embeddings. Therefore, the word "*director*" can be extracted by the entity tag *$director* to serve its surface form. With derived words F_i for entity tag e_i, we can normalize the relation probabilities the word $w_j \in F_i$ infers.

$$P_F(r_i \mid w_j) = P_F(e_i \mid w_j) = \frac{S_i^F(w_j)}{\sum_{k, w_j \in F_k} S_k^F(w_j)}, \tag{5.3}$$

where r_i is the relation inferred from the entity tag e_i, $S_k^F(w_j)$ is the score of the word w_j that belongs to the set F_k extracted by the entity tag e_k, and $P_F(r_i \mid w_j)$ is similar to $P_E(r_i \mid w_j)$ in (5.1) but based on derived words instead of specified entities.

5.5.3.2 Entity Syntactic Contexts

Another type of relational cues comes from contexts of entities; for example, a user utterance "*find movies produced by james cameron*" includes an unspecified movie entity "*movies*" and a specified entity "*james cameron*", which may be captured by entity surface forms via P_F and gazetteers via P_E respectively. However, it doesn't consider local observations "*produced by*". In this example, the most likely relation of the entity "*james cameron*" from the background knowledge is `director.name`, which infers `movie.directed_by`, and the local observations are not be used to derive the correct relation `movie.produced_by` for this utterance.

This procedure is to discover the relational entity contexts based on syntactic dependency. With dependency-based entity embeddings and their context embeddings, for each entity tag e_i, we extract top N syntactic contexts to form a set of entity contexts C_i, which includes the words that are the most activated by a given entity tag e_i. The extraction procedure is similar to one in Section 5.5.3.1; for each entity tag e_i, we compute the score of the word w_j as

$$S_i^C(w_j) = \frac{\text{CxtSim}(w_j, e_i)}{\sum_{e_k \in E} \text{CxtSim}(w_j, e_k)}, \tag{5.4}$$

where $\text{CxtSim}(w_j, e_i)$ is the cosine similarity between the context embeddings of the word w_j and the embeddings of the entity tag e_i.

The derived contexts may serve the indicators of possible relations. For instance, for the entity tag *$producer*, the most activated contexts include "*produced/prep_by*$^{-1}$", so the word "*produced*" can be extracted by this procedure for detecting local observations other than entities. Then we can normalize the relation probabilities the contexts imply to compute $P_C(r_i \mid w_j)$ similar to (5.3):

$$P_C(r_i \mid w_j) = P_C(e_i \mid w_j) = \frac{S_i^C(w_j)}{\sum_{k, w_j \in C_k} S_k^C(w_j)}. \tag{5.5}$$

5.6 Probabilistic Enrichment and Bootstrapping

Hakkani-Tür *et al.* proposed to use probabilistic weights for unsupervised relation detection [46]. We extend the approach to integrate induced relations from prior knowledge $P_E(r_i \mid w_j)$ and from local relational surface forms $P_F(r_i \mid w_j)$ and $P_C(r_i \mid w_j)$ to enrich the relation weights for effectively detecting relations given the utterances. This paper experiments to integrate multiple distributions in three ways:

- Unweighted

$$R_w(r_i) = \begin{cases} 1 & \text{, if } P_E(r_i \mid w) > 0 \text{ or } P_F(r_i \mid w) > 0 \text{ or } P_C(r_i \mid w) > 0. \\ 0 & \text{, otherwise.} \end{cases} \tag{5.6}$$

  This method combines possible relations from all sources, which tends to capture as many as possible relations (higher recall).

- Weighted

$$R_w(r_i) = \max(P_E(r_i \mid w), P_F(r_i \mid w), P_C(r_i \mid w)) \tag{5.7}$$

  This method assumes that the relation r_i invoked in word w comes from the source that carries the highest probability, so it simply selects the highest one among the three sources.

- Highest Weighted

$$R_w(r_i) = \max(P'_E(r_i \mid w), P'_F(r_i \mid w), P'_C(r_i \mid w)),$$

$$P'(r_i \mid w) = \mathbb{1}[i = \arg\max_i P(r_i \mid w)] \cdot P(r_i \mid w). \tag{5.8}$$

  This method only combines the most likely relation for each word, because $P'(r_i \mid w) = 0$ when the relation r_i is not the most likely relation of the word w.

r	actor	produced_by	location
$P_E(r \mid w)$	0.7	0.3	0
$P_F(r \mid w)$	0.4	0	0.6
$P_C(r \mid w)$	0	0	0
Unweighted $R_w(r)$	1	1	1
Weighted $R_w(r)$	0.7	0.3	0.6
Highest Weighted $R_w(r)$	0.7	0	0.6

Table 5.2: An example of three different methods in probabilistic enrichment ($w =$ "*pitt*").

Algorithm 1: Bootstrapping

Data: the set of user utterances $U = \{u_j\}$; the relation weights for the utterances, $R_{u_j}(r_i)$, $u_j \in U$;
Result: the multi-class multi-label classifier E that estimates relations given an utterance
Initializing relation labels $L^0(u_j) = \{r_i \mid R_{u_j}(r_i) \geq \delta\}$;
repeat
  Training ensemble of M weak classifiers E^k on U and $L^k(u_j)$;
  Classifying the utterance u_j by E^k and output probability distribution of relations as $R_{u_i}^{(k+1)}(r_i)$;
  Creating relation labels $L^{(k+1)}(u_j) = \{r_i \mid R_{u_j}^{(k+1)}(r_i) \geq \delta\}$;
until $L^{(k+1)}(u_j) \sim L^k(u_j)$;
return E^k;

An example of relation weights about the word "*pitt*" with three different methods is shown in Table 5.2. The final relation weight of the relation r_i given an utterance u, $R_u(r_i)$, can be compute as

$$R_u(r_i) = \max_{w \in u} R_w(r_i). \tag{5.9}$$

With enriched relation weights, $R_u(r_i)$, we train a multi-class, multi-label classifier in an unsupervised way, where we learn ensemble of weak classifiers by creating pseudo training labels in each iteration for boosting the performance [45, 31, 62, 88]. The detail of the algorithm is shown in Algorithm 1. Then the returned classifier E^k can be used to detect relations given unseen utterances.

5.7 Experiments

5.7.1 Dataset

The experiments use a list of entities/gazetteers from the publicly available Freebase knowledge graph. The list includes 670K entities of 78 entity types, including movie names, actors,

Query Statistics	Train	Test
% entity only	8.9%	10.7%
% rel only with specified movie names	27.1%	27.5%
% rel only with specified other names	39.8%	39.6%
% more complicated relations	15.4%	14.7%
% not covered	8.8%	7.6%
#utterance with SPARQL annotations	3338	1084

Table 5.3: Relation detection datasets used in the experiments.

Micro F-measure (%)		Unweighted	Weighted	Highest
Baseline	(a) Gazetteer	35.21	37.93	36.08
	(b) Gazetteer + Weakly Supervised	25.07	39.04	39.40
BOW	(c) Gazetteer + Surface Form	34.23	36.57	34.69
Dep.-Based	(d) Gazetteer + Surface Form	37.44	**41.01**	39.19
	(e) Gazetteer + Context	35.31	38.04	37.25
	(f) Gazetteer + Surface Form + Context	**37.66**	40.29	**40.07**

Table 5.4: The SLU performance of all proposed approaches without bootstrapping ($N = 15$).

release dates, etc. after filtering out the movie entities with lower confidences [53].

The relation detection datasets include crowd-sourced utterances addressed to a conversational agent and are described in Table 5.3. Both train and test sets are manually annotated with SPARQL queries, which are used to extract relation annotations. Most of data includes the relations with either specified movie names or specified other names. In addition, the relations only with specified movie names are difficult to capture by gazetteers, which emphasizes the contribution of this task. We use 1/10 training data as a development set to tune the parameters δ, M, and the optimal number of iterations in Algorithm 1. The training set is only used to train the classifier of Algorithm 1 for bootstrapping in an unsupervised way; note that the manual annotations are not used here.

For retrieving the snippets, we use 14 entity pairs from the knowledge graph related to movie entities, which include director, character, release date, etc. We extract snippets related to each pair from web search results, and we end up with 80K snippets, where the pairs of entities are marked in the returned snippets[5]. For all query snippets, we parse all with the Berkeley Parser [76], and then convert the output parse trees to dependency parses using the LTH Constituency-to-Dependency Conversion toolkit[6] for training dependency-based entity embeddings [56]. The trained entity embeddings have dimension 200 and vocabulary size is 1.8×10^5.

[5]In this work, we use top 10 results from Bing for each entity pair.

[6]`http://nlp.cs.lth.se/software/treebank_converter`

Micro F-measure (%)		Unweighted	Weighted	Highest
Baseline	(a) Gazetteer	36.91	40.10	38.89
	(b) Gazetteer + Weakly Supervised	37.39	39.07	39.98
BOW	(c) Gazetteer + Surface Form	34.91	38.13	37.16
Dep.-Based	(d) Gazetteer + Surface Form	38.37	41.10	42.74
	(e) Gazetteer + Context	37.23	38.88	38.04
	(f) Gazetteer + Surface Form + Context	**38.64**	**41.98**	**43.34**

Table 5.5: The SLU performance of all proposed approaches with bootstrapping ($N = 15$).

Entity Tag	Derived Word
$character	character, role, who, girl, she, he, officer
$director	director, dir, filmmaker
$genre	comedy, drama, fantasy, cartoon, horror, sci
$language	language, spanish, english, german
$producer	producer, filmmaker, screenwriter

Table 5.6: The examples of derived entity surface forms based on dependency-based entity embeddings.

5.7.2 Results

In the experiments, we train multi-class, multi-label classifiers using icsiboost [36], a boosting-based classifier, where we extract word unigrams, bigrams, and trigrams as classification features. The evaluation metric we use is micro F-measure for relation detection [46]. The performance with all of the proposed approaches before and after bootstrapping with $N = 15$ (top 15 similar words of each tag) is shown in Table 5.4 and Table 5.5 respectively.

The first baseline here (row (a)) uses gazetteers to detect entities and then infers the relations by background knowledge described in Section 5.4. Row (b) is another baseline [45], which uses the retrieved snippets and their inferred relations as labels to train a multi-class multi-label classifier, then outputs the relation probabilities for each utterance as $R_u(w)$, and integrates with the first baseline. Here the data for training only uses the patterns between entity pairs in the paths of dependency trees. Row (c) is the results of adding entity surface forms derived from original embeddings, which is shown for demonstrating the effectiveness of dependency-based entity embeddings (row (d)). Row (e) is the results of adding entity contexts, and row (f) combines both of entity surface forms and entity contexts. Below we analyze the effectiveness of proposed approaches.

5.8 Discussion

We analyze the effectiveness of learned entity surface forms and entity contexts, and compare different probabilistic enrichment methods, and validate the effectiveness of bootstrapping below.

5.8.1 Effectiveness of Entity Surface Forms

The row (c) and row (d) show the performance of using entity surface forms derived from bag-of-words and dependency-based embeddings respectively. It can be found that the words derived from original embeddings do not successfully capture the surface forms of entity tags, and the results cannot be improved. On the other hand, the results from dependency-based embeddings outperform the baselines for all enrichment methods, which demonstrate the effectiveness of including entity surface forms based on dependency relations for relation detection. To analyze results of entity surface forms, we show some examples about derived words in Table 5.6. It can be shown that the functional similarity carried by dependency-based entity embeddings effectively benefits relation detection task.

5.8.2 Effectiveness of Entity Contexts

Row (e) shows the results of adding entity contexts learned from dependency-based contexts. It does not show improvement compared to baselines. Nevertheless, combining with dependency-based entity surface forms, the F-measure achieves 43% by highest weighted probabilistic enrichment, which implies that including local observations based on syntactic contexts may help relation detection, but the influence is not significant.

5.8.3 Comparison of Probabilistic Enrichment Methods

From Table 5.4 and Table 5.5, among the three probabilistic enrichment methods, unweighted method performs worst, because it does not differentiate the relations with higher and lower confidence, and some relations with lower probabilities will be mistakenly outputted. Comparing between weighted and highest weighted methods, the first baseline using the weighted method performs better, while other approaches using the highest weighted method perform better. The reason probably is that weighted method can provide more possible relations for the baseline only using gazetteers to increase the recall, so the weighted method benefits the first baseline. On the other hand, proposed approaches have higher recall and the highest weighted method provides more precise relations, resulting in better performance when applying the highest weighted method.

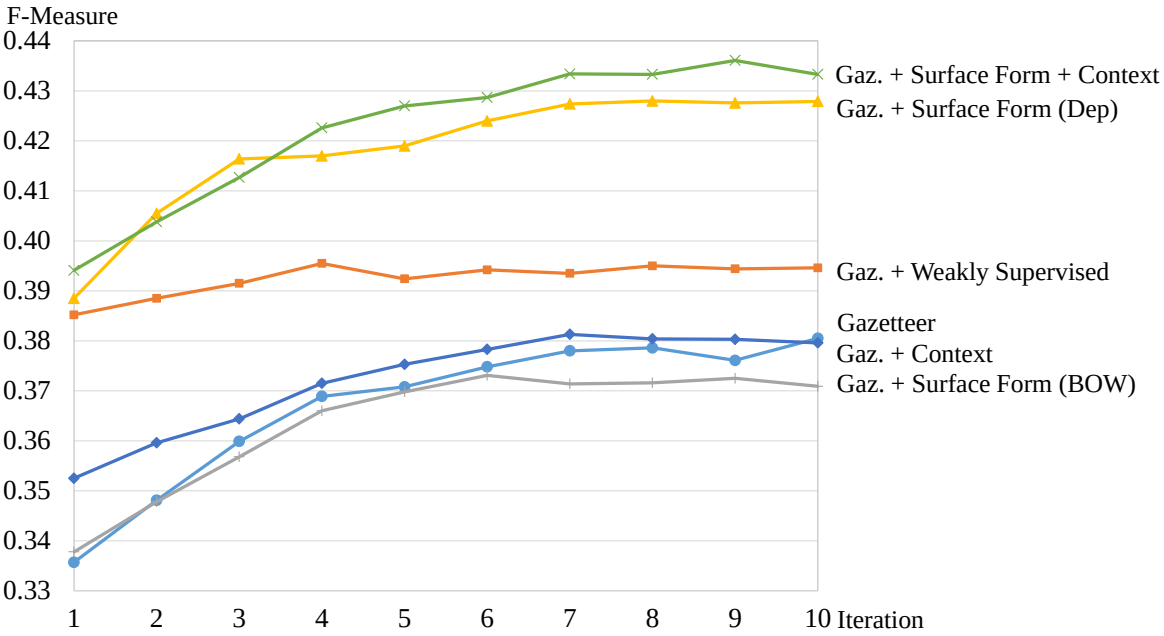


Figure 5.4: Learning curves over incremental iterations of bootstrapping.

5.8.4 Effectiveness of Bootstrapping

The F-measure learning curves of all results using highest weighted probabilistic enrichment on the test set are presented in Fig. 5.4. The light blue line marked with circles is the first baseline, which applies only gazetteers with probability distribution of entity types to relation detection. After bootstrapping, the performance is significantly improved and achieves about 39% of F-measure. Another baseline using a weakly supervised classifier (orange line marked with squares) performs well before bootstrapping, while the performance cannot be significantly improved with increased iterations. All other results show significant improvements after bootstrapping. The best result is the combination of all approaches (green line marked with crosses), and the curve shows the effectiveness and efficiency of bootstrapping. The reason probably is that the probabilities came from different sources can complement each other, and then benefit the classifiers. Also, only adding dependency-based entity surface forms (yellow line marked with triangles) performs similar to the combination result, showing that the major improvement comes from relational entity surface forms. The figure demonstrates the effectiveness of bootstrapping for improving relation detection.

5.8.5 Overall Results

The proposed approaches successfully capture local information other than background knowledge, where the relational surface forms can be learned by dependency-based entity embed-

dings trained on query snippets. After combining with prior relations induced by gazetteers, the relational information from the text domain can benefit the relation detection for the spoken domain. Also, the fully unsupervised approach shows the effectiveness of applying structured knowledge to SLU for tackling open domain problems.

5.9 Summary

This chapter proposes to automatically capture the relational surface forms including entity surface forms and entity contexts based on dependency-based entity embeddings. The detected semantics viewed as local observations can be integrated with background knowledge by probabilistic enrichment methods. Experiments show that involving derived entity surface forms as local cues together with prior knowledge can significantly help open domain SLU. Therefore, it is shown that the surface forms corresponding to the ontology carry important knowledge for building a good SLU component.

6 Semantic Decoding in SLU Modeling

With the organized ontology automatically learned by the knowledge acquisition, this chapter introduces a novel matrix factorization (MF) approach to learn latent feature vectors for utterances and semantic concepts. More specifically, our model learns the semantic slots for a domain-specific SDS in an unsupervised fashion, and then performs semantic decoding using latent MF techniques. To further consider the global semantic structure, such as inter-word and inter-slot relations, we augment the latent MF-based model with the structure of the learned ontology. The final goal of the model is, given an utterance, to predict semantic slots and word patterns, considering their relations and domain-specificity in a joint fashion.

6.1 Introduction

A key component of a spoken dialogue system (SDS) is the spoken language understanding (SLU) module—it parses the users' utterances into semantic representations; for example, the utterance "*find a cheap restaurant*" can be parsed into (price=*cheap*, target=*restaurant*). To design the SLU module of a SDS, most previous studies relied on predefined slots[1] for training the decoder [82, 34, 42, 8]. However, these predefined semantic slots may bias the subsequent data collection process, and the cost of manually labeling utterances for updating the ontology is expensive [96].

In recent years, this problem leads to the development of unsupervised SLU techniques [49, 50, 20]. In particular, Chen et al. [20] proposed a frame-semantics based framework for automatically inducing semantic slots given raw audios. However, these approaches generally do not explicitly learn the latent factor representations to model the measurement errors [84], nor do they jointly consider the complex lexical, syntactic, and semantic relations among words, slots, and utterances.

Another challenge of SLU is the inference of the hidden semantics. Considering an user utterance "*can i have a cheap restaurant*", from its surface patterns, we can see that it includes explicit semantic information about "price (cheap)" and "target (restaurant)"; however, it also includes hidden semantic information, such as "food" and "seeking", since the SDS needs

[1] A slot is defined as a basic semantic unit in SLU, such as "price" and "target" in the example.

to infer that the user wants to "find" some cheap "food", even though they are not directly observed in the surface patterns. Nonetheless, these implicit semantics are important semantic concepts for domain-specific SDSs. Traditional SLU models use discriminative classifiers [52] to predict whether the predefined slots occur in the utterances or not, ignoring the unobserved concepts and the hidden semantic information.

In this chapter, we take a rather radical approach: we propose a novel matrix factorization (MF) model for learning latent features for SLU, taking account of additional information such as the word relations, the induced slots, and the slot relations. To further consider the global coherence of induced slots, we combine the MF model with a knowledge graph propagation based model, fusing both a word-based lexical knowledge graph and a slot-based semantic graph. In fact, as it is shown in the Netflix challenge, MF is credited as the most useful technique for recommendation systems [57]. Also, MF model considers the unobserved patterns and estimates their probabilities instead of viewing them as negative examples. However, to the best of our knowledge, MF technique is not well-understood in the SLU and SDS communities, and it is not very straight-forward to use MF methods to learn latent feature representations for semantic parsing in SLU. To evaluate the performance of our model, we compare with standard discriminative SLU baselines, and show that our MF-based model is able to produce strong results in semantic decoding, and the knowledge graph propagation model further improves the performance. Our contributions are three-fold:

- We are among the first to study matrix factorization techniques for unsupervised SLU, taking account of the automatically learned knowledge;
- We augment the MF model with a knowledge graph propagation model, increasing the global coherence of semantic decoding using induced slots;
- Our experimental results show that the MF-based unsupervised SLU outperforms strong discriminative baselines, obtaining promising results.

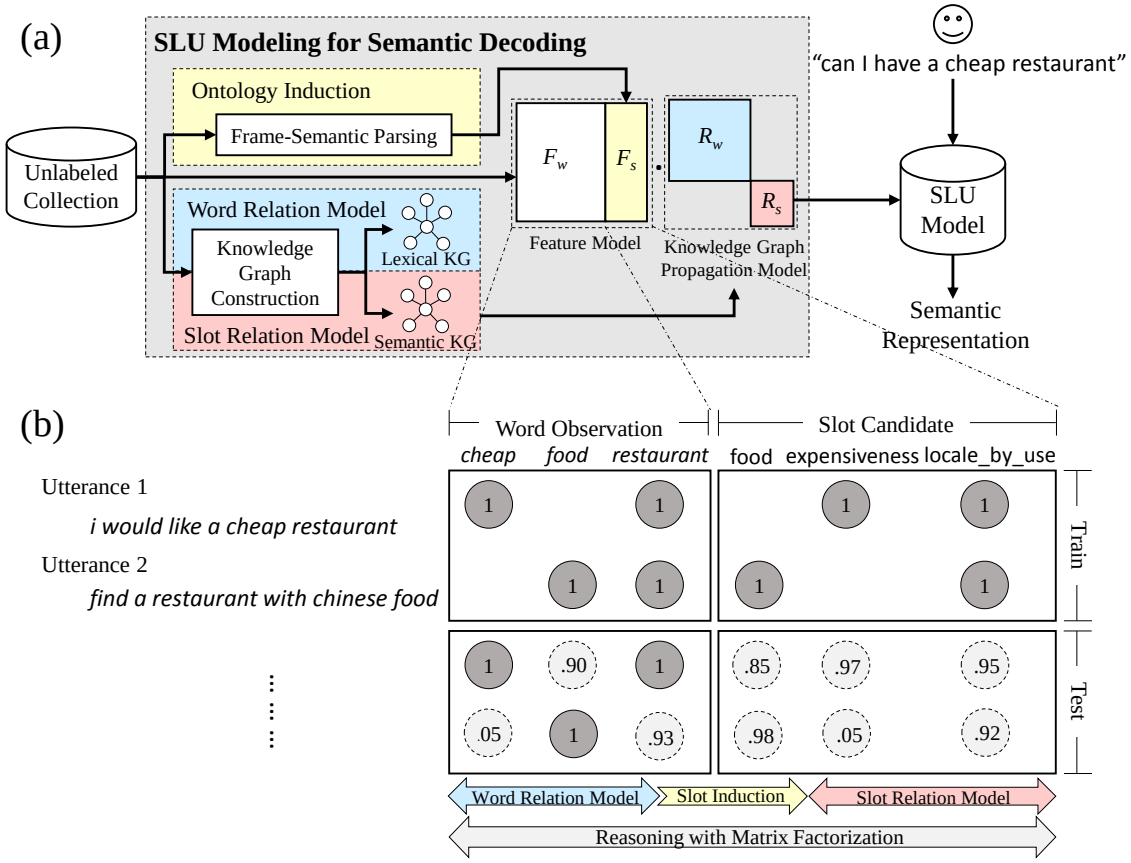


Figure 6.1: (a): the proposed framework. (b): our matrix factorization method completes a partially-missing matrix for implicit semantic parsing. Dark circles are observed facts, shaded circles are inferred facts. Slot induction maps observed surface patterns to semantic slot candidates. Word relation model constructs correlations between surface patterns. Slot relation model learns the slot-level correlations based on propagating the automatically derived semantic knowledge graphs. Reasoning with matrix factorization incorporates these models jointly, and produces a coherent, domain-specific SLU model.

6.2 Related Work

Unsupervised SLU Tur et al. [89, 91] are among the first to consider unsupervised approaches for SLU, where they exploited query logs for slot-filling. In a subsequent study, Heck and Hakkani-Tür [49] studied the Semantic Web for the intent detection problem in SLU, showing that results obtained from the unsupervised training process align well with the performance of traditional supervised learning. Following their success of unsupervised SLU, recent studies have also obtained interesting results on the tasks of relation detection [45, 21], entity extraction [95], and extending domain coverage [35]. However, most of the studies above do not explicitly learn latent factor representations from the data—while we hypothesize that the better robustness in noisy data can be achieved by explicitly model-

ing the measurement errors (usually produced by automatic speech recognizers (ASR)) using latent variable models and taking additional local and global semantic constraints.

Latent Variable Modeling in SLU Early studies on latent variable modeling in speech include the classic hidden Markov model for statistical speech recognition [55]. Recently, Celikyilmaz et al. [15] is the first to study the intent detection problem using query logs and a discrete Bayesian latent variable model. In the field of dialogue modeling, the partially observable Markov decision process (POMDP) [102] model is a popular technique for dialogue management, reducing the cost of hand-crafted dialogue managers while producing robustness against speech recognition errors. More recently, Tur *et al.* used a semi-supervised LDA model to show the improvement of the slot filling task [92]. Also, Zhai and Williams proposed an unsupervised model for connecting words with latent states in HMMs using topic models, obtaining interesting qualitative and quantitative results [103]. However, for unsupervised learning for SLU, it is not obvious how to incorporate additional information in the HMMs. To the best of our knowledge, this thesis is the first to consider MF techniques for learning latent feature representations in unsupervised SLU, taking various local and global lexical, syntactic, and semantic information into account.

6.3 The Proposed Framework

This thesis introduces matrix factorization techniques for unsupervised SLU. The proposed framework is shown in Figure 4.1(a). Given the utterances, the task of the SLU model is to decode their surface patterns into semantic forms and differentiate the target semantic concepts from generic semantic space for task-oriented SDSs simultaneously.

In the proposed model, we first build a feature matrix to represent the training utterances, where each row represents an utterance, and each column refers to an observed surface pattern or a induced slot candidate. Figure 4.1(b) illustrates an example of the matrix. Given a testing utterance, we convert it into a vector based on the observed surface patterns, and then fill in the missing values of the slots. In the first utterance in the figure, although semantic slot food is not observed, the utterance implies the meaning about food. The MF approach is able to learn the latent feature vectors for utterances and semantic elements, inferring implicit semantic concepts to improve the decoding process—namely, by filling the matrix with probabilities (lower part of the matrix).

The feature model is built on the observed word patterns and slot candidates, where the slot candidates are obtained from slot induction component through frame-semantic parsing (the yellow block in Figure 4.1(a)) [20]. Section 6.4.1 explains the detail of the feature model.

In order to consider the additional inter-word and inter-slot relations, we propose a knowledge

graph propagation model based on two knowledge graphs, which includes a word relation model (blue block) and a slot relation model (pink block), described in Section 4.4.1. The method of automatic knowledge graph construction is introduced in Section **??**, where we leverage distributed word embeddings associated with typed syntactic dependencies to model the relations [71, 72, 64, 23].

Finally, we train the SLU model by learning latent feature vectors for utterances and slot candidates through MF techniques. Combining with a knowledge graph propagation model based on word/slot relations, the trained SLU model estimates the probability that each semantic slot occurs in the testing utterance, and how likely each slot is domain-specific simultaneously. In other words, the SLU model is able to transform the testing utterances into domain-specific semantic representations without human involvement.

6.4 The Matrix Factorization Approach

Considering the benefits brought by MF techniques, including 1) modeling the noisy data, 2) modeling hidden semantics, and 3) modeling the dependencies between observations, this theis applies an MF approach to SLU model building for SDSs. In our model, we use U to denote the set of input utterances, W as the set of word patterns, and S as the set of semantic slots we would like to predict. The pair of an utterance $u \in U$ and a word pattern/semantic slot $x \in \{W + S\}$, $\langle u, x \rangle$, is a *fact*. The input to our model is a set of observed facts $\mathcal{O}$, and the observed facts for a given utterance is denoted by $\{\langle u, x \rangle \in \mathcal{O}\}$. The goal of our model is to estimate, for a given utterance u and a given word pattern/semantic slot x, the probability, $p(M_{u,x} = 1)$, where $M_{u,x}$ is a binary random variable that is true if and only if x is the word pattern/domain-specific semantic slot in the utterance u. We introduce a series of exponential family models that estimate the probability using a natural parameter $\theta_{u,x}$ and the logistic sigmoid function:

$$p(M_{u,x} = 1 \mid \theta_{u,x}) = \sigma(\theta_{u,x}) = \frac{1}{1 + \exp(-\theta_{u,x})} \tag{6.1}$$

We construct a matrix $M_{|U| \times (|W|+|S|)}$ as observed facts for MF by integrating a feature model and a knowledge graph propagation model below.

6.4.1 Feature Model

First, we build a binary word pattern matrix F_w based on observations, where each row refers to an utterance and each column refers to an observed word pattern. In other words, F_w carries the basic word vectors for the utterances, which is illustrated as the left part of the

matrix in Figure 4.1(b).

To induce the semantic elements, we parse all ASR-decoded utterances in our corpus using SEMAFOR[2], a state-of-the-art semantic parser for frame-semantic parsing [27, 28], and extract all frames from semantic parsing results as slot candidates [20]. Figure 3.2 shows an example of an ASR-decoded output parsed by SEMAFOR. Three FrameNet-defined frames (capability, expensiveness, and locale_by_use) are generated for the utterance, which we consider as slot candidates for a domain-specific dialogue system [3]. Then we build a binary slot matrix F_s based on the outputted slots, which also denotes the slot features for the utterances (right part of the matrix in Figure 4.1(b)).

For building the feature model M_F, we concatenate two matrices:

$$M_F = [\ F_w \quad F_s\ ], \tag{6.2}$$

which refers to the upper part of the matrix in Fig. 4.1(b) for training utterances. Note that we do not use any annotations, so all slot candidates are included.

6.4.2 Knowledge Graph Propagation Model

Since SEMAFOR was trained on FrameNet annotation, which has a more generic frame-semantic context, not all the frames from the parsing results can be used as the actual slots in the domain-specific dialogue systems. For instance, in Figure 3.2, we see that the frames "expensiveness" and "locale_by_use" are essentially the key slots for the purpose of understanding in the restaurant query domain, whereas the "capability" frame does not convey particular valuable information for SLU.

Assuming that domain-specific concepts are usually related to each other, globally considering relations between semantic slots induces a more coherent slot set. It is shown that the relations on knowledge graphs help decision of domain-specific slots [23]. Considering two directed graphs, semantic and lexical knowledge graphs built in Section 4.4.1 of Chapter 4, each node in the semantic knowledge graph is a slot candidate s_i outputted by the frame-semantic parser, and each node in the lexical knowledge graph is a word w_j. The structured graph helps define a coherent slot set. To model the relations between words/slots based on the knowledge graphs, we define two relation models below.

- Semantic Relation
  For modeling word semantic relations, we compute a matrix $R_w^S = [Sim(w_i, w_j)]_{|W|\times|W|}$, where $Sim(w_i, w_j)$ is the cosine similarity between the dependency embeddings of the

[2] http://www.ark.cs.cmu.edu/SEMAFOR/

word patterns w_i and w_j after normalization. For slot semantic relations, we compute $R_s^S = [Sim(s_i, s_j)]_{|S|\times|S|}$ similarly[3]. The matrices R_w^S and R_s^S model not only the semantic but functional similarity since we use dependency-based embeddings [64].

- Dependent Relation
  Assuming that important semantic slots usually mutually related to each other, that is, connected by syntactic dependencies, our automatically derived knowledge graphs are able to help model the dependent relations. For word dependent relations, we compute a matrix $R_w^D = [\hat{r}(w_i, w_j)]_{|W|\times|W|}$, where $\hat{r}(w_i, w_j)$ measures the dependency between two word patterns w_i and w_j based on the word-based lexical knowledge graph, which can be computed by (4.1) in Section 4.4.2.2 of Chapter 4. For slot dependent relations, we similarly compute $R_s^D = [\hat{r}(s_i, s_j)]_{|S|\times|S|}$ based on the slot-based semantic knowledge graph.

With the built word relation models (R_w^S and R_w^D) and slot relation models (R_s^S and R_s^D), we combine them as a knowledge graph propagation matrix M_R[4].

$$M_R = \begin{bmatrix} R_w^{SD} & 0 \\ 0 & R_s^{SD} \end{bmatrix}, \tag{6.3}$$

where $R_w^{SD} = R_w^S + R_w^D$ and $R_s^{SD} = R_s^S + R_s^D$ to integrate semantic and dependent relations. The goal of this matrix is to propagate scores between nodes according to different types of relations in the knowledge graphs [16].

6.4.3 Integrated Model

With a feature model M_F and a knowledge graph propagation model M_R, we integrate them into a single matrix.

$$\begin{aligned} M &= M_F \cdot (M_R + I), \\ &= \begin{bmatrix} F_w & F_s \end{bmatrix} \begin{bmatrix} R_w + I & 0 \\ 0 & R_s + I \end{bmatrix}, \\ &= \begin{bmatrix} F_w R_w + F_w & 0 \\ 0 & F_s R_s + F_s \end{bmatrix}, \end{aligned} \tag{6.4}$$

where M is final matrix and I is the identity matrix in order to remain the original values. The matrix M is similar to M_F, but some weights are enhanced through the knowledge graph propagation model, M_R. The word relations are built by $F_w R_w$, which is the matrix with

[3]For each column in R_w^S and R_s^S, we only keep top 10 highest values, which correspond the top 10 semantically similar nodes.

[4]The values in the diagonal of M_R are 0 to only model the propagation from other entries.

internal weight propagation on the lexical knowledge graph (the blue arrow in Fig. 4.1(b)). Similarly, F_sR_s models the slot correlations, and can be treated as the matrix with internal weight propagation on the semantic knowledge graph (the pink arrow in Fig. 4.1(b)).

F_s contains all slot candidates outputted by SEMAFOR, which may include some generic slots (such as capability), so the original feature model cannot differentiate the domain-specific and generic concepts. By integrating with R_s, the semantic and dependent relations can be propagated via the knowledge graph, and the domain-specific concepts may have higher weights based on the assumption that the slots for dialogue systems are often mutually related [23]. Hence, the structure information can be automatically involved in the matrix. Also, the word relation model brings the same function but on the word level. In conclusion, for each utterance, the integrated model not only predicts the probability that semantic slots occur but also considers whether the slots are domain-specific. The following sections describe the learning process.

6.4.4 Parameter Estimation

The proposed model is parametrized through weights and latent component vectors, where the parameters are estimated by maximizing the log likelihood of observed data [25].

$$\begin{aligned} \theta^* &= \arg\max_{\theta} \prod_{u \in U} p(\theta \mid M_u) \\ &= \arg\max_{\theta} \prod_{u \in U} p(M_u \mid \theta) p(\theta) \\ &= \arg\max_{\theta} \sum_{u \in U} \ln p(M_u \mid \theta) - \lambda_\theta, \end{aligned} \tag{6.5}$$

where M_u is the vector corresponding to the utterance u from $M_{u,x}$ in (6.1), because we assume that each utterance is independent of others.

To avoid treating unobserved facts as designed negative facts, we consider our positive-only data as *implicit feedback*. Bayesian Personalized Ranking (BPR) is an approach to learn with implicit feedback, which uses a variant of the ranking: giving observed true facts higher scores than unobserved (true or false) facts [79]. Riedel *et al.* also showed that BPR learn the implicit relations for improving the relation extraction task [80].

6.4.4.1 Objective Function

To estimate the parameters in (6.5), we create a dataset of *ranked pairs* from M in (6.4): for each utterance u and each observed fact $f^+ = \langle u, x^+ \rangle$, where $M_{u,x} \geq \delta$, we choose each

word pattern/slot x^- such that $f^- = \langle u, x^- \rangle$, where $M_{u,x} < \delta$, which refers to the word pattern/slot we have not observed to be in utterance u. That is, we construct the observed data $\mathcal{O}$ from M. Then for each pair of facts f^+ and f^-, we want to model $p(f^+) > p(f^-)$ and hence $\theta_{f^+} > \theta_{f^-}$ according to (6.1). BPR maximizes the summation of each ranked pair, where the objective is

$$\sum_{u \in U} \ln p(M_u \mid \theta) = \sum_{f^+ \in \mathcal{O}} \sum_{f^- \notin \mathcal{O}} \ln \sigma(\theta_{f^+} - \theta_{f^-}). \tag{6.6}$$

The BPR objective is an approximation to the per utterance AUC (area under the ROC curve), which directly correlates to what we want to achieve – well-ranked semantic slots per utterance.

6.4.4.2 Optimization

To maximize the objective in (6.6), we employ a stochastic gradient descent (SGD) algorithm [79]. For each randomly sampled observed fact $\langle u, x^+ \rangle$, we sample an unobserved fact $\langle u, x^- \rangle$, which results in $|\mathcal{O}|$ fact pairs $\langle f^-, f^+ \rangle$. For each pair, we perform an SGD update using the gradient of the corresponding objective function for matrix factorization [41].

6.5 Experiments

6.5.1 Experimental Setup

In this experiment, we used the Cambridge University SLU corpus, previously used on several other SLU tasks [52, 19]. The domain of the corpus is about restaurant recommendation in Cambridge; subjects were asked to interact with multiple SDSs in an in-car setting. The corpus contains a total number of 2,166 dialogues, including 15,453 utterances (10,571 for self-training and 4,882 for testing). The data is gender-balanced, with slightly more native than non-native speakers. The vocabulary size is 1868. An ASR system was used to transcribe the speech; the word error rate was reported as 37%. There are 10 slots created by domain experts: addr, area, food, name, phone, postcode, price range, signature, task, and type.

For parameter setting, the threshold for define the unobserved facts δ is set as 1.0 for all experiments. We use Stanford Parser to obtain the collapsed typed syntactic dependencies [85] and set the dimensionality of embeddings $d = 300$ in all experiments.

Approach			ASR	Manual
Logistic Regression		(a)	33.96	38.78
Random		(b)	$22.45^{\S}$	$25.09^{\S}$
MostPopular		(c)	32.88	38.41
MF	Feature	(d)	$37.61^{\dagger}$	$45.34^{\dagger}$
	Feature + KGP	(e)	$\mathbf{43.51}^{\dagger}$	$\mathbf{53.40}^{\dagger}$

Table 6.1: The MAP of predicted slots (%); † and § mean that the result is better and worse with $p < 0.05$ respectively.

6.5.2 Evaluation Metrics

To evaluate the accuracy of the automatically decoded slots, we measure their quality as the proximity between predicted slots and reference slots. Figure 3.3 shows the mappings that indicate semantically related induced slots and reference slots [20].

To eliminate the influence of threshold selection when predicting semantic slots, in the following metrics, we take the whole ranking list into account and evaluate the performance by the metrics that are independent of the selected threshold. For each utterance, with the predicted probabilities of all slot candidates, we can compute an average precision (AP) to evaluate the performance of SLU by treating the slots with mappings as positive. AP scores the ranking result higher if the correct slots are ranked higher, which also equals to the area under the precision-recall curve. Mean average precision (MAP) is the metric for evaluating all utterances. For all experiments, we perform paired t-test on the AP scores of the results to test the significance.

6.5.3 Evaluation Results

Table 6.1 shows the MAP performance of predicted slots for all experiments on ASR and manual transcripts. For the first baseline, we use the observed data to self-train a logistic regression model for predicting the probability of each semantic slot (row (a)) [75]. To improve probability estimation, we perform all experiments by combining with the result of logistic regression. Two baselines are performed as references, Random (row (b)) and MostPopular (row (c)). The former assigns random probabilities for all slots, which significantly degrades the performance for both ASR and manual transcripts. The later assigns higher probabilities for the slots occurring more frequently, which does not produce significant difference for both ASR and manual results.

The proposed MF approaches show that only using feature model for building the matrix (row (d)) significantly outperforms the baseline, where the performance comes from about 34.0% to 37.6% and from 38.8% to 45.3% for ASR and manual results respectively. Additionally

Model	Feature	Feature + Knowledge Graph Propagation Model				
Rel.	(a) None	(b) Semantic	(c) Dependent	(d) Word	(e) Slot	(f) All
M_R	-	$\begin{bmatrix} R_w^S & 0 \\ 0 & R_s^S \end{bmatrix}$	$\begin{bmatrix} R_w^D & 0 \\ 0 & R_s^D \end{bmatrix}$	$\begin{bmatrix} R_w^{SD} & 0 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0 & R_s^{SD} \end{bmatrix}$	$\begin{bmatrix} R_w^{SD} & 0 \\ 0 & R_s^{SD} \end{bmatrix}$
ASR	37.61	$41.39^{\dagger}$	$41.63^{\dagger}$	$39.19^{\dagger}$	$42.10^{\dagger}$	$\mathbf{43.51}^{\dagger}$
Manual	45.34	$51.55^{\dagger}$	$49.04^{\dagger}$	45.18	$49.91^{\dagger}$	$\mathbf{53.40}^{\dagger}$

Table 6.2: The MAP of predicted slots using different types of relation models in M_R (%); † means that the result is better with $p < 0.05$

integrating with a knowledge graph propagation (KGP) model (row (e)) further improves the performance (achieving MAP of 43.5% and 53.4%), showing that the proposed models successfully learn the implicit semantics and consider the relations and domain-specificity simultaneously.

6.6 Discussion and Analysis

With promising results of proposed models, we analyze the detailed difference between different relation models in Table 6.2.

6.6.1 Effectiveness of Semantic and Dependent Relation Models

To evaluate the effectiveness of semantic and dependent relations, we consider each of them individually in M_R of (6.3) (columns (b) and (c) in Table 6.2). Comparing to the original model (column (a)), both modeling semantic relations and modeling dependent relations significantly improve the performance for ASR and manual results. It is shown that semantic relations help the SLU model infer the implicit meaning, and then the prediction becomes more accurate. Also, dependent relations successfully differentiate the generic concepts from the domain-specific concepts, so that the SLU model is able to predict more coherent set of semantic slots [23]. Integrating two types of relations (column (f)) further improves the performance.

6.6.2 Comparing Word/ Slot Relation Models

To analyze the performance results from inter-word and inter-slot relations, the column (d) and (e) show the results considering only word relations and only slot relations respectively. It is presented that the inter-slot relation model significantly improves the performance for both ASR and manual results. However, the inter-word relation model only performs slightly better results for ASR output (from 37.6% to 39.2%), and there is no difference after applying the

inter-word relation model on manual transcripts. The reason may be that inter-slot relations carry high-level semantics that align well with the structure of SDSs, but inter-word relations do not. Nevertheless, combining two relations (column (f)) outperforms both results for ASR and manual transcripts, showing that different types of relations can compensate each other and then benefit the SLU performance.

6.7 Summary

This chapter presents an MF approach to self-train the SLU model for semantic decoding with consideration of a well-organized ontology in an unsupervised way. The purpose of the proposed model is not only to predict the probability of each semantic slot but also to distinguish between generic semantic concepts and domain-specific concepts that are related to an SDS. The experiments show that the MF-based model obtains promising results of SLU, outperforming strong discriminative baselines.

7 Behavior Prediction for SLU Modeling

SLU modeling has different aspects. In addition to semantic decoding, behavior prediction is another aspect about understanding user intents. A good SLU module is able to accurately understand users' semantics and further predict their follow-up behaviors to offer better interactions. This chapter focuses on predicting user behaviors in order to deeply understand user intents in an unsupervised manner.

7.1 Introduction

The semantic representations of users' utterances usually refer to the specified information that directly occurs in the utterances. To involve deeper understanding, the high-level intention should be considered. For example, in a restaurant domain, "*find me a chinese restaurant*" has the semantic form (type=*chinese*, target=*restaurant*), and follow-up behavior might be asking the location or asking for navigation, which can be viewed as the high-level intention. Assuming that the SDS is able to predict an user's follow-up behavior, the system may not only return the user a restaurant list but also ask whether the user needs the corresponding location or navigating instructions, providing better and more user-friendly conversational interactions.

7.2 Proposed Framework

Behavior prediction is another aspect of SLU modeling, so that the proposed framework can be modified from Chapter 6, which is shown in Figure 7.1.

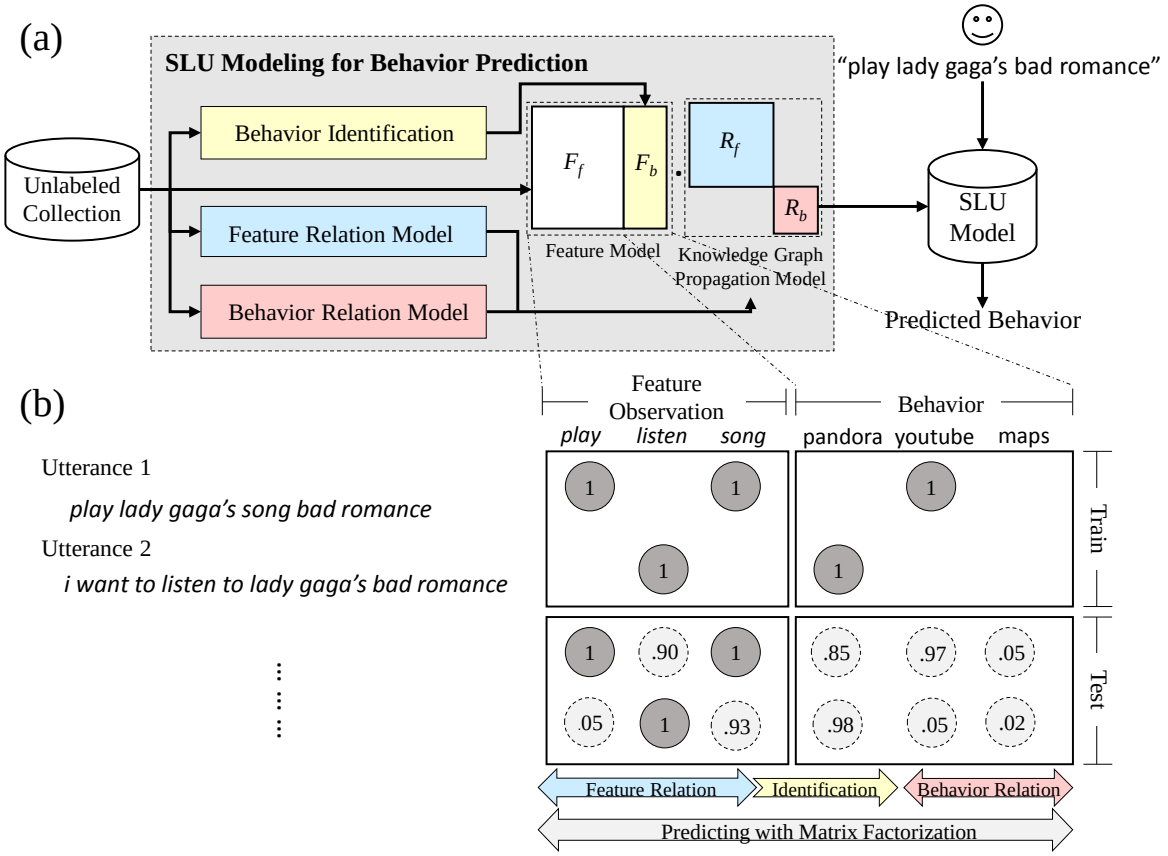


Figure 7.1: (a): the proposed framework. (b): our matrix factorization method completes a partially-missing matrix for implicit semantic parsing. Dark circles are observed facts, shaded circles are inferred facts. Behavior identification maps observed features to behaviors. Feature relation model constructs correlations between features. Behavior relation model trains the correlations between behaviors or their transitions. Predicting with matrix factorization incorporates these models jointly, and produces an SLU model that understands the users better by predicting the follow-up behaviors.

7.3 Data Description

The data we plan to work includes interactions in two domains, mobile application interactions and insurance-related interactions. The detail information is described below.

7.3.1 Mobile Application Interaction

Considering to expand useful domains of SDS, we extract the most popular applications from mobile app stores to define important domains users tend to access frequently, and the defined domains are used to design the experiments for the task. Fig. 7.2 shows total 13 domains we define for the experiments. The speech corpus used in this experiment is collected from 5

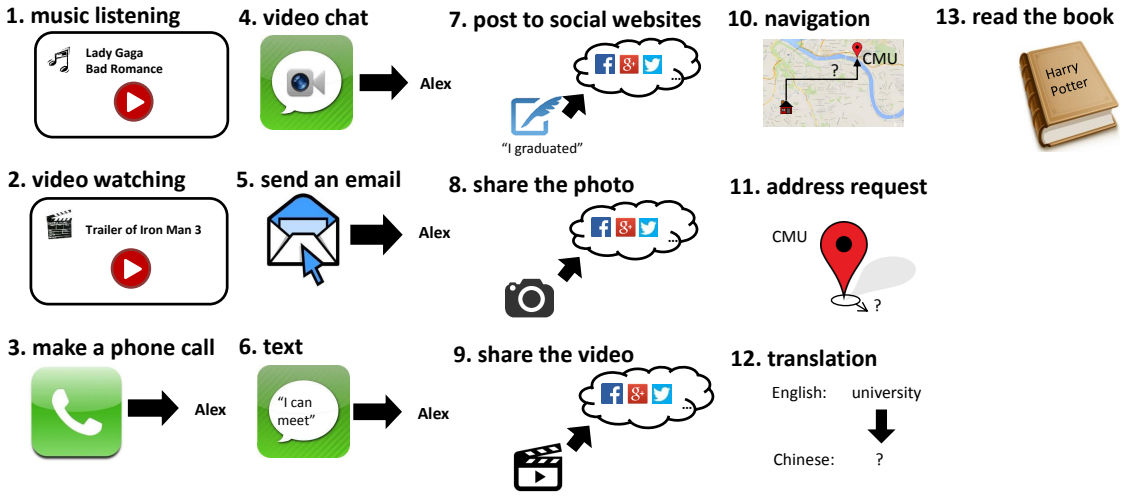


Figure 7.2: Total 13 tasks in the corpus (only pictures are shown to subjects for making requests).

Task ID	Transcript of the utterance
3	*please dial a phone call to alex* *can i have a telcon with alex*
10	*how can i go from my home to cmu* *i'd like navigation instruction from my home to cmu*

Table 7.1: The recording examples collected from some subjects.

non-native subjects (1 female and 4 males). They are only provided with pictures referring to domain-specific tasks in a random order, and for each picture/task a subject is asked to use 3 different ways to make requests for fulfilling the task implied by the displayed picture. Thus 39 utterances (total 13 tasks and 3 ways for each) are collected from each subject. Fig. 7.2 shows provided pictures and implied tasks, and some recording examples are shown in Table 7.1. The corpus contains 195 utterances. An automatic speech recognition (ASR) system was used to transcribe speech into text, and the word error rate is reported as 19.8%. Here we use Google Speech API to perform better recognition results because it covers more named entities, which may be out-of-vocabulary words for most recognizers. The average number of words in an utterance is 6.8 for ASR outputs and 7.2 for manual transcripts, which implies the challenge of retrieving relevant applications with limited information in a query.

7.3.2 Insurance-Related Interaction

The used data is the conversations between customers and agents collected by MetLife[1], where there are total 52 conversations and the number of utterances is 2,205 segmented by short pauses and different speakers.

7.4 Behavior Identification

This section describes the procedure about identifying user behaviors from above two domains.

7.4.1 Mobile Application

The data for retrieval archive was collected from Google Play[2] in November 2012. Each Android app in Google Play has its own description page, and the extracted metadata we use includes its name, number of downloads[3], and content description. The total number of considered applications is 140,854[4]. For evaluation, subjects are asked to manually annotate applications from Google Play that can support the corresponding tasks. Then we use the subject-labelled applications as our ground truth for evaluating our returned applications.

7.4.2 Insurance

The behaviors in the data include request_info, call_back, verify_id, etc, which should be manually annotated with consideration of the flow in these dialogues.

7.5 Feature Relation Model

We can use various feature observations such as words, explicit semantic concepts (the information predicted by Chapter 6 can be treated as observations), etc. in the matrix. The relations between features can be built by their similarity or dependency for building a feature relation model.

[1] `https://www.metlife.com/`

[2] `http://play.google.com`

[3] Google does not provide the absolute number of downloads. Instead, it discretizes this number into several ranges.

[4] Google defines two major categories for the programs, "game" and "application". This paper only uses apps with category "application".

7.6 Behavior Relation Model

The behavioral dynamics can be modeled according to their transitional probabilities, which may benefit to better prediction. To build the transition model, some self-training methods are considered to model the relations between behaviors.

7.7 Summary

With the previous success on predicting implicit semantics given an utterance using an MF technique, user intents may be predictable based on the similar approach.

8 Conclusions and Future Work

8.1 Conclusions

Our work consists in automatically acquiring domain knowledge from available semantic resources and then understanding both the semantics of individual utterances and user intents. While the amount of spoken interactions from different domains is increasing, automating the system development is a trend considering efficiency and maintenance. The proposal addresses two challenges of SDS, missing a predefined structured ontology and shallow understanding, and proposes knowledge acquisition and SLU modeling techniques to show the potential solutions.

For knowledge acquisition, semantic concepts, the structure, and surface forms are automatically learned. It is shown that such information is helpful to understand the semantics better.

For SLU modeling, we predict the semantics from the individual utterances with consideration of the structured ontology, since knowledge acquisition part shows effectiveness of the automatically learned ontology on the understanding task. The proposed MF framework is useful and easy to expand more features and predicted elements. The final step is to predict user behaviors by modeling their intents. For this purpose, we plan to use a similar MF approach for the prediction tasks in two different domains.

8.2 Future Work

8.2.1 Chapter 2

- Write detailed background knowledge
- Survey more related work

8.2.2 Chapter 7

- Behavior identification for the insurance domain
- Behavior annotation for the insurance domain

- Implement the behavior prediction approach
- Evaluate the performance
- Analyze and discuss the results

8.3 Timeline

- April 2015: finish annotating the behavior labels for the insurance data
- June 2015: implement behavior prediction
- July 2015: submit a conference paper about behavior prediction
- August 2015: submit a journal paper
- September 2015: finish thesis writing
- October 2015: thesis defense
- December 2015: finish thesis revision

Bibliography

[1] James F Allen, Donna K Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent. Toward conversational human-computer interaction. *AI magazine*, 22(4):27, 2001.

[2] Collin Baker. Framenet, present and future. *Programme Committee 7*, 2008.

[3] Collin F Baker, Charles J Fillmore, and John B Lowe. The Berkeley FrameNet project. In *Proceedings of COLING*, pages 86–90, 1998.

[4] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of Language Annotation Workshop*, 2013.

[5] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, 2014.

[6] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, pages 1533–1544, 2013.

[7] Dan Bohus and Alexander I Rudnicky. LARRI: A language-based maintenance and repair assistant. In *Spoken multimodal human-computer dialogue in mobile environments*, pages 203–218. Springer, 2005.

[8] Dan Bohus and Alexander I Rudnicky. The RavenClaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3):332–361, 2009.

[9] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of International Conference on Management of Data*, 2008.

[10] Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. Learning structured embeddings of knowledge bases. In *Proceedings of AAAI*, 2011.

[11] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*, 2013.

[12] Kendrick Boyd, Vitor Santos Costa, Jesse Davis, and C David Page. Unachievable region in precision-recall space and its effect on empirical evaluation. In *Machine learning: proceedings of the International Conference. International Conference on Machine Learning*, volume 2012, page 349. NIH Public Access, 2012.

[13] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

[14] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4): 467–479, 1992.

[15] Asli Celikyilmaz, Dilek Hakkani-Tür, and Gokhan Tür. Leveraging web query logs to learn user intent via bayesian discrete latent variable model. 2011.

[16] Yun-Nung Chen and Florian Metze. Two-layer mutually reinforced random walk for improved multi-party meeting summarization. In *Proceedings of The 4th IEEE Workshop on Spoken Language Tachnology*, pages 461–466, 2012.

[17] Yun-Nung Chen and Florian Metze. Multi-layer mutually reinforced random walk with hidden parameters for improved multi-party meeting summarization. In *INTERSPEECH*, pages 485–489, 2013.

[18] Yun-Nung Chen and Alexander I. Rudnicky. Dynamically supporting unexplored domains in conversational interactions by enriching semantics with neural word embeddings. In *Proceedings of SLT*, 2014.

[19] Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky. An empirical investigation of sparse log-linear models for improved dialogue act classification. In *Proceedings of ICASSP*, pages 8317–8321, 2013.

[20] Yun-Nung Chen, William Yang Wang, and Alexander I Rudnicky. Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing. In *Proceedings of ASRU*, pages 120–125, 2013.

[21] Yun-Nung Chen, Dilek Hakkani-Tür, and Gokhan Tur. Deriving local relational surface forms from dependency-based entity embeddings for unsupervised spoken language understanding. In *Proceedings of SLT*, 2014.

[22] Yun-Nung Chen, William Yang Wang, and Alexander I Rudnicky. Leveraging frame semantics and distributional semantics for unsupervised semantic slot induction in spoken dialogue systems. In *Proceedings of SLT*, 2014.

[23] Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky. Jointly modeling inter-slot relations by random walk on knowledge graphs for unsupervised spoken language understanding. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015.

[24] Ananlada Chotimongkol. *Learning the structure of task-oriented conversations from the corpus of in-domain dialogs*. PhD thesis, Carnegie Mellon University, 2008.

[25] Michael Collins, Sanjoy Dasgupta, and Robert E Schapire. A generalization of principal components analysis to the exponential family. In *Advances in neural information processing systems*, pages 617–624, 2001.

[26] Bob Coyne, Daniel Bauer, and Owen Rambow. VigNet: Grounding language in graphics using frame semantics. In *Proceedings of the ACL 2011 Workshop on Relational Models of Semantics*, pages 28–36, 2011.

[27] Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A Smith. Probabilistic frame-semantic parsing. In *Proceedings of NAACL-HLT*, pages 948–956, 2010.

[28] Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. Frame-semantic parsing. *Computational Linguistics*, 2013.

[29] Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56, 2014.

[30] Marie-Catherine De Marneffe and Christopher D Manning. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8. Association for Computational Linguistics, 2008.

[31] Renato De Mori, Frédéric Bechet, Dilek Hakkani-Tur, Michael McTear, Giuseppe Riccardi, and Gokhan Tur. Spoken language understanding. *Signal Processing Magazine, IEEE*, 25(3), 2008.

[32] Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[33] Li Deng, Gokhan Tur, Xiaodong He, and Dilek Hakkani-Tür. Use of kernel deep convex networks and end-to-end learning for spoken language understanding. In *Proceedings of SLT*, pages 210–215, 2012.

[34] John Dowding, Jean Mark Gawron, Doug Appelt, John Bear, Lynn Cherny, Robert Moore, and Douglas Moran. Gemini: A natural language system for spoken-language understanding. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 54–61. Association for Computational Linguistics, 1993.

[35] Ali El-Kahky, Derek Liu, Ruhi Sarikaya, Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. Extending domain coverage of language understanding systems via intent transfer between domains using knowledge graphs and search query click logs. In *Proceedings of ICASSP*, 2014.

[36] Benoit Favre, Dilek Hakkani-Tür, and Sebastien Cuendet. Icsiboost. 2007. URL `http://code.google.come/p/icsiboost`.

[37] Charles Fillmore. Frame semantics. *Linguistics in the morning calm*, pages 111–137, 1982.

[38] Charles J Fillmore. Frame semantics and the nature of language. *Annals of the NYAS*, 280(1):20–32, 1976.

[39] Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A Smith. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. Citeseer, 2014.

[40] GW Furnas, TK Landauer, LM Gomez, and ST Dumais. Statistical semantics: Analysis of the potential performance of keyword information systems. In *Proceedings of Human Factors in Computer Systems*, pages 187–242, 1984.

[41] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Mymedialite: A free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 305–308. ACM, 2011.

[42] Narendra Gupta, Gokhan Tur, Dilek Hakkani-Tur, Srinivas Bangalore, Giuseppe Riccardi, and Mazin Gilbert. The AT&T spoken language understanding system. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):213–222, 2006.

[43] Dilek Hakkani-Tür, Frédéric Béchet, Giuseppe Riccardi, and Gokhan Tur. Beyond ASR 1-best: Using word confusion networks in spoken language understanding. *Computer Speech & Language*, 20(4):495–514, 2006.

[44] Dilek Hakkani-Tür, Asli Celikyilmaz, Larry P Heck, and Gokhan Tur. A weakly-supervised approach for discovering new user intents from search query logs. In *Proceedings of INTERSPEECH*, 2013.

[45] Dilek Hakkani-Tür, Larry Heck, and Gokhan Tur. Using a knowledge graph and query click logs for unsupervised learning of relation detection. In *Proceedings of ICASSP*, pages 8327–8331, 2013.

[46] Dilek Hakkani-Tur, Asli Celikyilmaz, Larry Heck, Gokhan Tur, and Geoff Zweig. Probabilistic enrichment of knowledge graph entities for relation detection in conversational understanding. In *Proceedings of INTERSPEECH*, 2014.

[47] Zellig S Harris. Distributional structure. *Word*, 1954.

[48] Kazi Saidul Hasan and Vincent Ng. Frame semantics for stance classification. *CoNLL-2013*, page 124, 2013.

[49] Larry Heck and Dilek Hakkani-Tür. Exploiting the semantic web for unsupervised spoken language understanding. In *Proceedings of SLT*, pages 228–233, 2012.

[50] Larry P Heck, Dilek Hakkani-Tür, and Gokhan Tur. Leveraging knowledge graphs for web-scale unsupervised semantic parsing. In *Proceedings of INTERSPEECH*, 2013.

[51] Steffen Hedegaard and Jakob Grue Simonsen. Lost in translation: authorship attribution using frame semantics. In *Proceedings of ACL-HLT*, pages 65–70, 2011.

[52] Matthew Henderson, Milica Gasic, Blaise Thomson, Pirros Tsiakoulis, Kai Yu, and Steve Young. Discriminative spoken language understanding using word confusion networks. In *Proceedings of SLT*, pages 176–181, 2012.

[53] Dustin Hillard, Asli Celikyilmaz, Dilek Z Hakkani-Tür, and Gokhan Tur. Learning weighted entity lists from web click logs for spoken language understanding. In *Proceedings of INTERSPEECH*, 2011.

[54] Diana Inkpen and Graeme Hirst. Building and using a lexical knowledge base of near-synonym differences. *Computational Linguistics*, 32(2):223–262, 2006.

[55] Frederick Jelinek. *Statistical methods for speech recognition*. MIT press, 1997.

[56] Richard Johansson and Pierre Nugues. Extended constituent-to-dependency conversion for english. In *16th Nordic Conference of Computational Linguistics*. University of Tartu, 2007.

[57] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[58] Amy N Langville and Carl D Meyer. A survey of eigenvector methods for web information retrieval. *SIAM review*, 47(1):135–161, 2005.

[59] Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539. Association for Computational Linguistics, 2011.

[60] Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W Cohen. Reading the web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1017–1026. Association for Computational Linguistics, 2012.

[61] Staffan Larsson and David R Traum. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural language engineering*, 6(3&4):323–340, 2000.

[62] Fabrice Lefevre, François Mairesse, and Steve Young. Cross-lingual spoken language understanding from unaligned data using discriminative classification models and machine translation. In *INTERSPEECH*, 2010.

[63] Oliver Lemon, Kallirroi Georgila, James Henderson, and Matthew Stuttle. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the talk in-car system. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, pages 119–122. Association for Computational Linguistics, 2006.

[64] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of ACL*, 2014.

[65] Peipei Li, Haixun Wang, Hongsong Li, and Xindong Wu. Assessing sparse information extraction using semantic contexts. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1709–1714. ACM, 2013.

[66] Peipei Li, Haixun Wang, Kenny Q Zhu, Zhongyuan Wang, and Xindong Wu. Computing term similarity by large probabilistic isa knowledge. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1401–1410. ACM, 2013.

[67] Percy Liang. *Semi-supervised learning for natural language*. PhD thesis, Massachusetts Institute of Technology, 2005.

[68] Tomáš Mikolov. *Statistical language models based on neural networks*. PhD thesis, Brno University of Technology, 2012.

[69] Tomáš Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of INTERSPEECH*, pages 1045–1048, 2010.

[70] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*, 2013.

[71] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in Neural Information Processing Systems*, 2013.

[72] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pages 746–751, 2013.

[73] Scott Miller, Robert Bobrow, Robert Ingria, and Richard Schwartz. Hidden understanding models of natural language. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 25–32. Association for Computational Linguistics, 1994.

[74] J. Pasternack and D. Roth. The wikipedia corpus, 2008.

[75] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[76] Slav Petrov and Dan Klein. Learning and inference for hierarchically split pcfgs. In *Proceedings of the National Conference on Artificial Intelligence*, 2007.

[77] Roberto Pieraccini, Evelyne Tzoukermann, Zakhar Gorelov, J Gauvain, Esther Levin, Chin-Hui Lee, and Jay G Wilpon. A speech understanding system based on statistical representation of semantics. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 193–196. IEEE, 1992.

[78] Patti Price. Evaluation of spoken language systems: The ATIS domain. In *Proceedings of the Third DARPA Speech and Natural Language Workshop*, pages 91–95. Morgan Kaufmann, 1990.

[79] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

[80] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. 2013.

[81] Alexander I Rudnicky, Eric H Thayer, Paul C Constantinides, Chris Tchou, R Shern, Kevin A Lenzo, Wei Xu, and Alice Oh. Creating natural dialogs in the carnegie mellon communicator system.

[82] Stephanie Seneff. TINA: A natural language system for spoken language applications. *Computational linguistics*, 18(1):61–86, 1992.

[83] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[84] Anders Skrondal and Sophia Rabe-Hesketh. *Generalized latent variable modeling: Multilevel, longitudinal, and structural equation models*. Crc Press, 2004.

[85] Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*, 2013.

[86] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642, 2013.

[87] Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. Short text conceptualization using a probabilistic knowledgebase. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2330–2336. AAAI Press, 2011.

[88] Gokhan Tur. Multitask learning for spoken language understanding. In *Proceedings of ICASSP*, 2006.

[89] Gokhan Tur, Dilek Z Hakkani-Tür, Dustin Hillard, and Asli Celikyilmaz. Towards unsupervised spoken language understanding: Exploiting query click logs for slot filling. In *Proceedings of INTERSPEECH*, 2011.

[90] Gokhan Tur, Li Deng, Dilek Hakkani-Tür, and Xiaodong He. Towards deeper understanding: deep convex networks for semantic utterance classification. In *Proceedings of ICASSP*, pages 5045–5048, 2012.

[91] Gokhan Tur, Minwoo Jeong, Ye-Yi Wang, Dilek Hakkani-Tür, and Larry P Heck. Exploiting the semantic web for unsupervised natural language semantic parsing. In *Proceedings of INTERSPEECH*, 2012.

[92] Gokhan Tur, Asli Celikyilmaz, and Dilek Hakkani-Tur. Latent semantic modeling for slot filling in conversational understanding. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8307–8311. IEEE, 2013.

[93] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[94] Fang Wang, Zhongyuan Wang, Zhoujun Li, and Ji-Rong Wen. Concept-based short text classification and ranking. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1069–1078. ACM, 2014.

[95] Lu Wang, Dilek Hakkani-Tür, and Larry Heck. Leveraging semantic web search and browse sessions for multi-turn spoken dialog systems. In *Proceedings of ICASSP*, 2014.

[96] William Yang Wang, Dan Bohus, Ece Kamar, and Eric Horvitz. Crowdsourcing the acquisition of natural language corpora: Methods and observations. In *Proceedings of SLT*, pages 73–78, 2012.

[97] Wayne Ward and Sunil Issar. Recent improvements in the CMU spoken language understanding system. In *Proceedings of the Workshop on Human Language Technology*, pages 213–216, 1994.

[98] Wei Xu and Alexander I Rudnicky. Task-based dialog management using an agenda. In *Proceedings of the 2000 ANLP/NAACL Workshop on Conversational systems*, pages 42–47, 2000.

[99] Nicole Yankelovich. Using natural dialogs as the basis for speech interface design. In *Human Factors and Voice Interactive Systems*, pages 255–290. Springer, 2008.

[100] Wen-tau Yih, Xiaodong He, and Christopher Meek. Semantic parsing for single-relation question answering. In *Proceedings of ACL*, 2014.

[101] Steve Young. CUED standard dialogue acts. Technical report, Cambridge University Engineering Department, 2007.

[102] Steve Young, Milica Gasic, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[103] Ke Zhai and Jason D Williams. Discovering latent structure in task-oriented dialogues. In *Proceedings of the Association for Computational Linguistics*, 2014.

[104] Victor Zue, Stephanie Seneff, James R Glass, Joseph Polifroni, Christine Pao, Timothy J Hazen, and Lee Hetherington. JUPlTER: a telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, 8(1):85–96, 2000.