

Towards Building a Domain Independent Dialog System

Thesis submitted in partial fulfillment
of the requirements for the degree of

M.S.
in
Computational Linguistics
by Research

by

Prathyusha Jwalapuram
201225005

`prathyusha.jwalapuram@research.iiit.ac.in`



International Institute of Information Technology
Hyderabad - 500 032, INDIA

April 2018

Copyright © Prathyusha Jwalapuram, 2017
All Rights Reserved

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Towards Building a Domain Independent Dialogue System ” by Prathyusha Jwalapuram, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Dr. Radhika Mamidi

To SHINee, EXO, Super Junior, Wanna One and Fall Out Boy, for getting me through the tough times.

Acknowledgments

I would like to extend my gratitude towards my advisor Dr. Radhika Mamidi for her unwavering support and guidance.

Special thanks to Prof. Dipti Misra Sharma and Dr. Manish Shrivastava for having been an integral part of my research progress.

Also thanks to my father and Chikkamma for their support.

Abstract

This thesis discusses a mixed-initiative, domain independent dialog system based on a hierarchically structured knowledge base. The system is rule-based and uses dependency relations and part-of-speech tags obtained from the Stanford Parser coupled with the hierarchical structure of the knowledge base to identify the user's goal. Primarily, the system is able to accept multi-sentence inputs, that is, multiple sentences forming a query in a single user turn. The system was tested for its accuracy over answering questions, and also subjective testing was done to evaluate the dialog flow; mainly over the books domain. We show examples of the system developed over the domains of books, movies and restaurants to demonstrate the domain independence.

We also discuss in detail the simple, rule-based system of extracting relevant information from a user's query that contributes to the domain independent dialog system. Relevant information is extracted from a user query by using only dependency relations and POS tags, obtained from the Stanford Dependency Parser. Using the universal dependency tags provided by the parser, we tried to understand the semantic structure of a query. This is done by looking for semantically important dependents of the verb, such as the subject, direct object, prepositions and their objects, and so on. Using information obtained from a combination of the dependency relations and the inherent semantic implications of words (such as 'who' or 'where'), we tried to extract the main objective or keyword of the query, and the constraints pertaining to it. This implementation itself is domain independent, however, a mapping to a knowledge base would require some domain knowledge; this issue is resolved using the hierarchically structured knowledge base. This keyword and constraint identification system is tested on the course management domain and the library domain.

In order to further improve the domain independent keyword identification for natural language queries, we used statistical methods. We took queries supplemented by only their dependency tags (Stanford Parser) and part-of-speech tags (Stanford POS tagger) and labeled the keywords. We then delexicalised the training data, and used the Conditional Random Fields algorithm to learn these labels. We used the queries created by [45] in the course management domain for training, and tested our model on the queries of three domains: course management, library and the GEOQUERIES250 dataset and report fairly high accuracies of 90.65%, 83.19% and 97.13% respectively, making our model a truly domain independent and highly accurate keyword identifier.

There is no agreed upon standard for the evaluation of conversational dialog systems, which are well-known to be hard to evaluate. The difficulty lies in pinning down metrics that will correspond to

human judgements and the subjective nature of human judgment itself. We explored the possibility of using Grice's Maxims to evaluate effective communication in conversation. We collected a few system generated dialogs from popular conversational chatbots across the spectrum and conducted a survey to see how the human judgements based on Gricean maxims correlate, and if such human judgments can be used as an effective evaluation metric for dialogs.

Contents

Chapter	Page
1 Introduction	1
1.1 Motivation	1
1.2 Key Contributions	2
1.3 Organization	3
2 Related Work	4
3 Handling Multi-Sentence Queries in a Domain Independent Dialog System	9
3.1 Introduction	9
3.2 Hierarchically Structured Knowledge Base	9
3.3 Dialog Manager	11
3.3.1 Processing the Parser Output	12
3.3.2 Multiple Sentences	13
3.3.3 Identifying Relationships in the Knowledge Base	14
3.3.4 Dialog Flow	14
3.3.5 Topic Change	15
3.3.6 Ambiguities and Clarification dialog	16
3.4 Domain Independence	17
3.5 Evaluation	18
3.5.1 Accuracy	18
3.5.2 Error Analysis	19
3.6 Future Work	19
4 Domain Independent Keyword and Constraint Identification for Question Answering	21
4.1 Introduction	21
4.2 Our Approach	22
4.2.1 Motivation behind using Dependency Relations	22
4.2.2 Dealing with Parsing Errors	23
4.2.3 Processing the Parser Output	23
4.2.3.1 Prepositions and their Arguments	23
4.2.3.2 Question Words	24
4.2.3.3 Yes/No Questions	24
4.2.3.4 Modifiers and Conjunctions	24
4.2.3.5 Temporal Information	25
4.2.3.6 Verbs and their Modifiers	25

4.2.3.7	Example	25
4.2.3.8	Domain Independence	26
4.3	Query Equation	26
4.4	Evaluation	28
4.4.1	Accuracy	28
4.4.2	Error Analysis	29
4.4.2.1	Conjunction - 'and'	29
4.4.2.2	Syntactic Parsing Issues	29
4.4.2.3	Pragmatics	30
4.4.2.4	Relative Clauses	30
4.5	Statistical Methods (CRF)	30
4.6	Data	30
4.6.0.1	Course Management Domain	30
4.6.0.2	Library Domain	31
4.6.0.3	GeoQueries250	31
4.7	Approach	31
4.7.1	Training Data	31
4.7.1.1	Labeling	31
4.7.1.2	Delexicalisation	32
4.7.2	Question Answering	32
4.8	Experiments	32
4.9	Comparison with the Rule Based System	34
4.10	Error Analysis	35
4.11	Conclusions and Future Work	35
5	Evaluating Dialogs Based on Grice's Maxims	37
5.1	Introduction	37
5.1.1	Gricean Maxims	37
5.2	Survey Description	38
5.2.1	Dialog Collection	38
5.2.2	Survey Questions	39
5.3	Results	39
5.3.1	Discussion	39
5.3.2	Ranking Analysis	40
5.3.3	Comparing the Means	40
5.3.4	Detailed Example	41
5.4	Conclusions	42
5.5	Future Work	43
5.6	Appendix: Dialogs	43
6	Conclusions and Future Work	46
	Bibliography	48

List of Figures

Figure	Page
3.1 Hierarchical Knowledge Base Structure	10
3.2 Simplified System Architecture	11
4.1 System Architecture	22
4.2 Dependency Tree of a Sample Query	26

List of Tables

Table	Page
3.1 Multiple Sentence Inputs	13
3.2 Books: Topic Change	15
3.3 Books: Clarification dialog	16
3.4 Books, Movies: Moving through the Hierarchy	17
3.5 Restaurants: Moving through the Hierarchy	18
3.6 Survey Query Evaluation	18
3.7 Subjective Evaluation	18
4.1 Example of Ambiguity	21
4.2 Query Equations	27
4.3 Results	28
4.4 Example of Training Data	32
4.5 Feature Templates and Accuracies	33
4.6 Precision, Recall and F1 scores for each template and domain	34
4.7 Accuracy of the CRF Model compared to the Rule-Based System	34
5.1 Distribution of Dialogs	38
5.2 Survey Results	38
5.3 Dialogs Ranked by Mean Scores	39
5.4 Significance of Maxim Mean as Threshold	41

Chapter 1

Introduction

A dialog system is a computer program trained to engage in conversation, with the goal to be as natural sounding as possible. One of the earliest dialog systems to be built was ELIZA [52], which used simple pattern matching techniques to interact with the user. ALICE [50] is also another famous system that uses a similar heuristic pattern matching technique.

Eventually dialog systems evolved into integrated task-oriented systems such as TRAINS [2] and TRIPS [11], which were designed to be planning assistants. [1] list travel booking agent, design consultant, disaster relief management, etc. as some areas (in increasing order of difficulty) suitable for a dialog interface with the system using finite state scripts, sets of contexts and so on. Because most of these approaches require anticipation of what the user can say and how, they tend to focus on particular tasks and employ information particular to that task, becoming domain-specific. Chapter 2 discusses the relevant work in detail.

This thesis presents the work done towards building a domain independent dialog system, using both rule-based and statistical methods. It handles multiple sentences as input using a hierarchically structured knowledge base to guide the dialog flow. Keyword identification and constraint extraction form the major crux of the thesis. We also include work done to introduce a new set of evaluation metrics for artificially generated dialogs.

1.1 Motivation

Most dialog systems focus on processing the user's input and classifying the dialog in terms of the amount of information it presents and the possible paths the dialog could take. The idea is to predict a possible goal of the user in order to be able to ask relevant questions if needed in order to fill information gaps, and provide more relevant replies. Using a hierarchically structured knowledge base for a dialog system helps achieve this. They help us limit the possible paths of the dialog, and can help us identify irrelevant inputs or topic changes.

Very few dialog systems attempt to process multi-sentence inputs (multiple utterances in a single user turn) that collectively behave as a query. Users of a dialog system may break up their queries into

multiple sentences, and also provide additional information and qualify their initial statements. In such cases it might not suffice to simply find all the relevant keywords; it might be important to understand the relationships between them as well.

In order to provide an effective question answering system, it is important to be accessible to the common user. The common user does not know a query language or programming or understand templates and formats. In order to be accessible to this common user, the system must be able to understand a natural language query. For example, for a user to be able to access data from a database, not only does the user need to know a query language used by the database, but the user needs to have a thorough knowledge of the database schema as well in order to be able to formulate a query based on the information they seek. Most users don't have the requisite knowledge.

The idea behind an interface is to allow users to ask questions in everyday, conversational language, and process these questions to convert them to a valid query in a language understood by the database (such as SQL) or any other knowledge representation system (such as an ontology) in order to obtain results, and thereby answer the user's questions. The typical information retrieval approach is to filter out function words and use the noun phrases as keywords; in this thesis we show that significant information can be gained by inferring information from function words.

If system building is challenging, then evaluation of the system becomes even more challenging. For dialog systems, the goal of an evaluation metric is to determine the system's performance, defined as the ability of a system to provide the function it has been designed for [35]. Typical measures of dialog systems tend to be purely objective, based on task completion accuracies, or other machine learning measures such as precision, recall or f-measure. Since a dialog system is about the conversational experience, it is important to measure if the system is natural-sounding as well.

Chatbot evaluation remains an open problem; measuring conversational naturalness is dependent of the subjectivity of the users. One of the problems of questionnaire-based evaluation methods is their lack of quantitative universality, as a set of chatbot performance metrics have not been defined [23]. [23] also says, however, that current research has found success in using these techniques to gain general insight on the effectiveness of the conversational agents and make relative comparisons between them.

1.2 Key Contributions

1. Domain independent dialog system that accepts multiple-sentence inputs
2. Hierarchically structured knowledge bases for books, movies and restaurants domain
3. Domain independent keyword identification system (rule-based and CRF-based)
4. Keyword annotated datasets for course management, books and GEOQUERIES250
5. Evaluation metrics for dialogs based on Grice's maxims

1.3 Organization

We present an implementation over a simple, hand-crafted knowledge base of books, movies, and restaurants, that is essentially structured into a useful hierarchy. We evaluated the system based on whether the replies are relevant or not on multi-sentence queries that were collected through a survey (objective evaluation) and qualitatively through participant interaction and rating (subjective evaluation).

We tried to link the keywords in complex and multiple sentences to get a clear picture of exactly what the user is looking for. We also attempted to look at expressions of negation and negative words indicating the exclusion of certain objects as required by the user. The system is domain-independent and is presented in Chapter 3.

In Chapter 4, we discuss domain independent keyword identification strategies in more detail. We reduced the problem to keyword identification in question answering, in order to focus on the natural language queries.

We first built a rule based system that correlates dependency relations and part-of-speech tags with keywords, using prepositions, conjunctions, and other modifiers as clues in addition to the subject and object of the verb. We attempted to put together the relevant information contained in a query in a useful format for adapting to any type of knowledge base. Using the same linguistic features and a relatively smaller dataset, we improved the accuracies of the rule based system by using Conditional Random Fields to label keywords.

We also attempted to introduce a shorter measure of dialog evaluation based on the cooperative principles proposed by Grice [15]. We collected system generated dialogs from popular systems across the board and conducted a survey to gauge its effectiveness. This work is discussed in Chapter 5.

In summary, Chapter 2 will discuss related work, Chapter 3 will present the dialog system, Chapter 4 will discuss domain independent keyword identification, Chapter 5 is about a new subjective dialog evaluation method, and Chapter 6 concludes the thesis.

Chapter 2

Related Work

One of the primary features of our domain independent dialog system is its hierarchically structured knowledge base. The advantages of using an ontological knowledge base for a dialog system were put forth in [33] which defined an ontology simply as a network of concepts and instances related to each other through semantic links. They introduce a mixed-initiative dialog system that uses part-whole and is-a relationships to drive clarification questions and determine the sequence of the dialog. Since we seek to further simplify the structure into a more generic relationship set (rather than is-a and part-whole relationships), we refer to our knowledge base as hierarchically structured.

It has been argued that the separation of the dialog management from the domain knowledge management helps reduce the complexity of the systems and enhance further extensions [12]. [12] uses is-a and part-of relations to resolve issues of under and over specification. We used a similar approach to make our implementation domain independent. Swapping the knowledge base with another in the same format would produce a working dialog system for the new knowledge base, even of a different domain.

[8] describes a system that maintains two ontologies: a domain independent ontology for a parser, linked to the lexicon to capture the aspects of dialog interaction that are common across domains, and a domain-specific ontology for knowledge representation. They integrate the linguistic and domain knowledge by defining a set of mappings between the two.

The division of the task and the dialog is also explored in [5]. A dialog engine, 'Ravenclaw', generates domain independent conversational behaviors and the dialog task specification is handled separately. RavenClaw uses hierarchical task decomposition, which has a tree-like structure. The dialog task specification is domain specific.

[28] proposes an example-based dialog modeling that is applicable over different domains, but requires dialog corpora for each domain; they also do not consider multi-sentence utterances in the course of a dialog.

[32] proposes a generic command interpreter for natural languages that uses an ontology to clarify semantic concepts, coded in a specific language. They use a tokenizer, tagger, lemmatizer and a chunker, but discount the need for a grammar based syntactic parser. They remove stop words and treat the sentences as a bag of words, and use the ontology for concept matching and semantic analysis. They

do not take into account the semantic information that is provided by function words such as 'in', 'and' and so on. They assume that the users' commands are unlikely to be complete sentences, and therefore do not consider multiple sentences at all.

[4] describes an ontology based dialog system that simulates a health counselor, using an RDF-based ontology described in OWL. They maintain a representation of a plan tree, keeping track of the ones that are actively being used and their goal-subgoal task decomposition relationships. Their dialog fragments are modeled through a task representation language, and a dialog planner enacts these fragments. They introduce the notion of adjacency pairs which are logically related, consisting, for example, of an utterance and its response.

Although current dialog systems use state-of-the-art neural models to train end-to-end systems [47, 53, 29, 30, 54], they use enormous amounts of conversational data and training resources, and are heavily dependent on the characteristics and quality of the training data.

In order to build a domain independent dialog system, we first need to have a domain independent keyword identification system.

Keyword identification for question answering for a particular domain is usually done using keyword or pattern matching [24]. This domain specific approach requires the anticipation of a large number of patterns or keywords in order to cover all the possibilities and variations, and may still fail in the case of complex queries.

Systems such as PLANES [51], PRECISE [38] and STEP [34] opt for frames or patterns based on semantic grammar or, as in the case of STEP, a phrasal lexicon developed over the database. As in the case of keywords, this approach is not only domain dependent, but also relies heavily on a pre-constructed list, which is restrictive and therefore needs to be extensive in order to be practically useful.

[3] proposed a Computational Paninian Grammar framework for interpreting natural language queries, for example by creating verb frames of a list of domain-specific verbs in order to identify relationships between the keywords. We tried to make it a domain independent implementation by using dependency relations with less significance given to the verbs, allowing us to focus on the relationships between the prepositions, adjectives, nouns/noun phrases, etc.

A rule-based system was proposed by [16] that was based on the framework of Computational Paninian Grammar [3] to identify the semantic templates that a query may belong to. This approach uses domain specific semantic mapping, wherein Paninian grammar constructs are mapped to dependency relations. However, it requires the creation of verb frames with specified arguments, making the query structure rigid. All possible argument structures must be identified for all possible verbs in the domain, and each of these arguments must be mapped to the relevant entities they refer to in the knowledge base. Combined with the introduction of parser errors, this approach can work well with limited types of questions.

General domain keywords might be identified for web scale data [10, 14, 9] which requires a large number of training instances, and may not be specific for natural language queries.

The general strategy used in domain-independent question answering systems, typically on web-scale data such as community question answering forums, etc., is to classify questions based on question words. The documents are then queried using the question, and in the results returned, the entities which match with the answer are detected [48]. These often failed in cases which had a person as answer without 'who' such as "name the person." or "which person", and so on, showing that they had little semantic understanding of the query.

[45] uses the CRF algorithm [26] for concept identification in an NL query, which is an intermediary stage in a Natural Language Interface to Database (NLIDB) system. Here, concept refers to the tables, attributes and relations in a database schema. NL tokens in the query are mapped to one of these using a specific tagset, essentially reducing the concepts identification problem to a kind of Named Entity Recognition (NER) problem. The system is domain-specific to the course management domain, and requires annotation and training when there are changes made to the tagset or for every new domain. However, this system shows a high precision for a domain specific system; in this case a course management portal in the academic domain.

[57] models the keyword extraction for describing the meaning of a document in Chinese as string labeling. They use CRF [26] for the keyword extraction and show that it outperforms SVM and multi-linear regression. A large number of local and global features including a word window of +/- 2, length of the word, tf-idf, occurrence in title/abstract/full-text/reference, position of first appearance, and so on are used on 600 documents in the field of economics. They achieve a best F1 score of 0.5125.

There have been many attempts to introduce an evaluation system for dialog systems, although many of them depend on objective measures or use a combination that does not translate very well into user experience.

To measure dialog quality or usability we can use subjective measures such as user satisfaction or likelihood of future use; however subjective metrics are difficult to measure and are dependent on the context and the goals of individual users [19].

[19] also delineates the differences in the evaluations requirements of task-oriented systems. In task-oriented systems, the primary interest lies in task completion with efficient, effective and streamlined conversation. Whereas in conversational systems, the dialogs are exploratory and user engagement (rather than dialog efficiency) is of primary interest.

[37] notes that the needs of evaluation themselves might be inconsistent: apart from measuring task success, evaluations that allow comparative judgements with other systems may be needed, preferably across domains; the purpose of the evaluation may be to identify the parts of the system that must be improved, or to discover tradeoffs or correlations between certain factors in the system.

In the case of task-oriented systems, objective metrics such as dialog success rate or completion time do not always correspond to the most effective user experience due to the interactive nature of dialog [27]. Domain-specific systems yield higher response satisfaction scores when coupled with a full complement of conversational dialog patterns and knowledge [17, 41].

[31] speculates that learning a model that uses human-survey data to score proposed responses may be no easier than the problem of response generation itself, in which case human evaluations must always be used together with other metrics.

The classic Turing test involves an independent evaluator who judges a conversation between a computer designed to generate dialog and a human [46]. It is designed to check if the machine can exhibit intelligent behaviour similar to humans. The evaluator observes the conversation and guesses which one of them is the computer. If the evaluator cannot distinguish between a human and the computer, the computer passes the Turing test.

29,935 input and responses from ALICE [50] were evaluated by [42] on the basis of correction rate (percentage of system responses corrected by the user) and response satisfaction (measure of appropriateness of the system given user query context, on a Likert scale from 1-strongly disagree to 7-strongly agree). They calculate accuracy as (1-correction rate). However, the user corrections had to be analyzed separately; they identify the error categories in the conversational dialog as nonsense replies, wordy and awkward responses, and more problematically, spurious user corrections (where unnecessary corrections are offered by the user presumably for their own entertainment, despite the category being awarded higher than average Response Satisfaction scores).

PARADISE [49] proposes a combined performance metric to measure user satisfaction of a dialog system as a weighted linear combination of task completion measures and dialog costs (efficiency costs: number of utterances and dialog time, and quality costs: system-response delay, mean recognition score). PARADISE includes a user-satisfaction survey that incorporates questions about task ease, interaction pace, user expertise, system response times, and expected behaviour of the system. [22] point out some issues with the PARADISE method: they argue that the questions in the survey are not based on theory or well-conducted empirical research, and that summing up all of the scores cannot be justified unless they are measuring the same construct, and therefore the overall score would be meaningless.

Task completion based on a kappa statistic has been considered a good common measure for dialog systems that represent the domain task as Attribute-Value matrices, Bayesian networks or other non-symbolic representations [37].

In wizard-of-oz techniques, humans communicate with another human under the illusion that they are interacting with a computational system in order to generate gold standard data. Even such evaluations essentially compare a dialog system on a particular domain task with how human beings perform on the same task [13, 6].

[27] observes that subjective opinions are more important than word error or query understanding rates from the user's viewpoint. However, these rates played a part in influencing the user's perception of the system.

[43] use a questionnaire where the users rate the impression, command, effectiveness, navigability, ability to learn, ability to aid and comprehension of the system on a scale ranging from 'Very Unsatisfied' to 'Very Satisfied'. They concede that the subjective evaluation does not provide statistically verified conclusiveness, but serves as a general indicator of performance.

A universal chatbot evaluation system using dialog efficiency, dialog quality and user satisfaction was proposed by [44]. To measure dialog quality, users sorted system responses into reasonable, weird but understandable, and nonsensical. User satisfaction is also measured qualitatively through feedback.

[40] measured the degree of naturalness and the degree to which users were willing to continue the conversation with the system (through human judges who assigned a score between 1 and 10 for both these metrics) and were able to compare different approaches.

[31] uses machine translation metrics like BLEU and METEOR and embedding (word2vec) based semantic metrics to evaluate dialog response generation for non task-oriented, unsupervised models. A model's generated response is compared to a single target response. They show that these metrics correlate very weakly (non-technical Twitter domain) or not at all (technical Ubuntu domain) with human judgements.

[18] propose using the cooperative principles to test text coherence, where irrelevant semantic paths generated from Wordnet are filtered out based on Gricean maxims. They infer rules for possible semantic paths based on whether the conversations respect each of the four maxims; based on this, they discard paths that do not contain contextual concepts, or establish connections between already related concepts (maxim of Quantity), or path with contradictory information (maxim of Quality), paths semantically distant from the text context (maxim of Relation), paths with large levels of abstraction (maxim of Manner), etc. They also use the maxim of Manner to select the shortest path between two concepts, to find repeating concepts, and check for the coherence of the text.

[55] use the Gricean maxim of quantity to select the content of plan descriptions such that they are concise and effective and natural sounding to people; a plan description is considered cooperative when it contains no more and no less detail than is needed. They evaluate this architecture through an experiment and show that subjects made fewer execution errors and achieved more task goals when they followed instructions produced by this architecture.

Chapter 3

Handling Multi-Sentence Queries in a Domain Independent Dialog System

3.1 Introduction

In this chapter we present the domain independent dialog system. This includes handling of multi-sentence queries in a mixed-initiative dialog system based on a hierarchically structured knowledge base.

This chapter is organised as follows. Section 2 describes the structure of the knowledge base, section 3 describes the dialog manager, section 4 shows some examples of dialog demonstrating domain independence, section 5 is about the evaluation and error analysis and section 6 explores possible future work.

3.2 Hierarchically Structured Knowledge Base

The knowledge base is structured in a hierarchical manner for ease of representation, and also to facilitate dialog flow (Figure 3.1). For example, in the *books* domain, the books maybe in a hierarchy from *genre* to *author* to *title* and so on. This helps form a possible path that a dialog may take; for example the user may specify a *genre* they like, and the system may suggest relevant *authors*; the user may then pick an author and the system suggests *books* by the author, and so on.

Another reason for maintaining a structured knowledge base rather than a standard ontology is the flexibility in defining relationships. In our knowledge base, the entities are related by a '*is-x-of*' relationships; that is, *is-author-of*, *is-genre-of*, etc. This could allow us to structure knowledge bases in domains which may not fit into the traditional ontology structure. We can also define synonyms for the relationships for making search easier (*author*, *writer*).

Tag Information

Although typical data about books consists of information like author, year of publishing, genre, etc., in order to answer queries, the system must also be able to consider what the books are about. This is

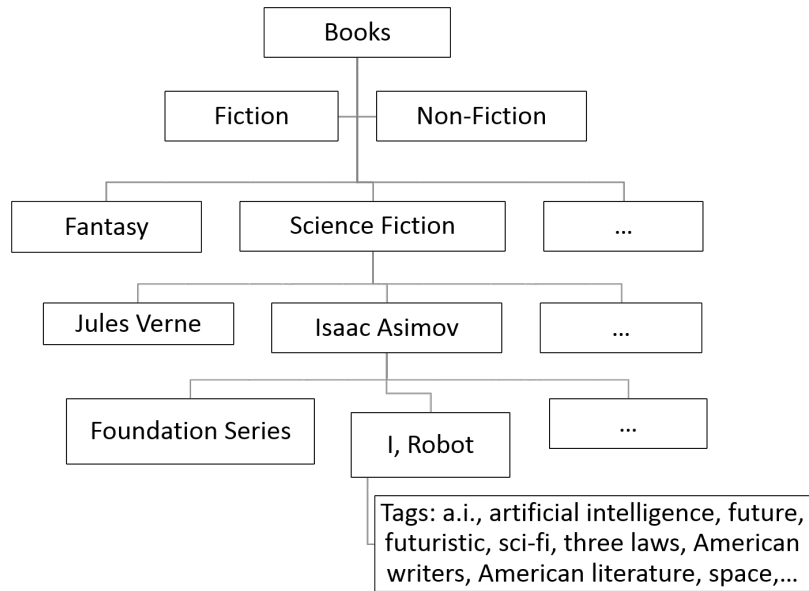


Figure 3.1: Hierarchical Knowledge Base Structure

problematic since we usually do not have information about the content of the books; the titles of the books are not always informative, and a large set of questions would have to go unanswered.

To resolve this issue and expand the scope of queries being answered, tags provided for books by the general public on popular book sites were collected and added to the knowledge base (or alternatively, high-frequency content words from book reviews can also be scraped). Tags are commonly used in searches for a similar purpose, allowing you to look for a book based on themes, characters, and other classifications which are not necessarily genres or part of a typical knowledge base.

Consider a query like “books with magic and animals in them”. The *Harry Potter* series, for example, would be a good fit for such a query; but nothing in any of the titles of the series would directly imply this. However, some of the tags for the series include *dark magic, witches, wizards, young adult, British fiction, beasts, dragons*, etc, which would allow us to infer that both *magic* and *animals* play a part in these books (synonyms must be considered).

Similarly, *women writers, female writer* are part of the tags for both the *Harry Potter* series and the *Hunger Games* series; this helps us return results to queries like “Do you have books written by female authors?”. All additional information, like prizes won by the books, the time period it is set in (for example, “detective fiction set in the Victorian times” = *Sherlock Holmes*), etc. is available to us through these tags, enabling us to provide results to a wide variety of requirements; including spelling variations (*theatre, theater*).

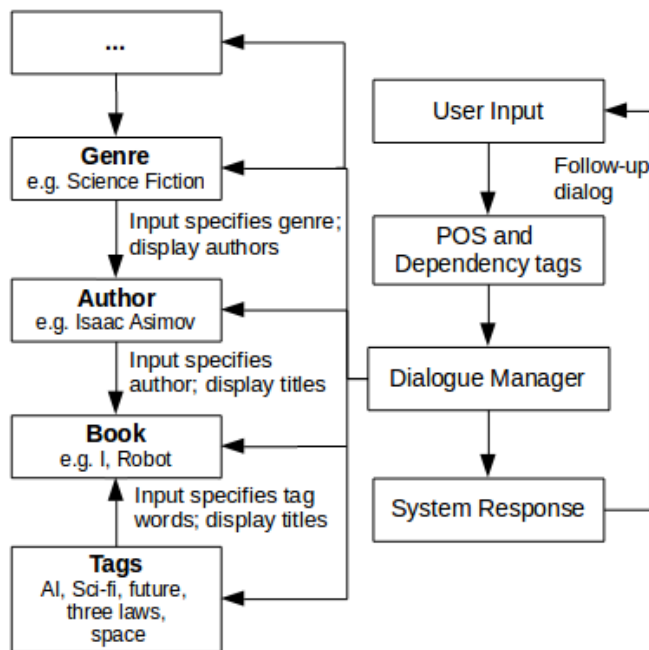


Figure 3.2: Simplified System Architecture

For the movie domain, similar ‘tags’ are available in terms of ‘plot points’, or high-frequency content words can be scraped from movie reviews on popular movie review sites. For the restaurant domain, we use the restaurant menu item descriptions (which are mainly descriptions of ingredients and the like).

3.3 Dialog Manager

Our emphasis was to try and identify the objective of a user’s query, that is, the data the user is looking for and the constraints pertaining to this data. We used the Stanford POS Tagger and the Stanford Dependency Parser for this purpose while assuming that the queries are free of errors and are fairly grammatically sound. We did not attempt to resolve abbreviations and ambiguities. The system is able to extract information from ungrammatical queries in certain cases where the Stanford Parser is able to generate a fairly accurate dependency parse.

We considered the cases where the user directly specifies what they are looking for, or makes oblique remarks that are intended to help reach the goal. These statements can be simple, complex or span multiple sentences.

An online survey was conducted to obtain questions for the development and testing. A total of 57 participants submitted 118 questions. All domain specific assumptions and rules are based on a development set of 68 questions; the other 50 were separated for testing.

See Figure 3.2 for a simplified representation of the system architecture.

3.3.1 Processing the Parser Output

Keywords and Negation

An example of an input by the user could be “*I like Oscar Wilde*”. Here, the parser should tag *like* as the root, *I* as the nsubj (subject) and *Oscar Wilde* as the dobj (direct object). We see that the object here is the keyword we are looking for. Using such information, keywords in the utterance can be detected [25].

Similarly, we also considered cases where the user expresses a negative sentiment, such as “*I don’t like Oscar Wilde*”. Here the negation applies to the *root*, and *Wilde* is the object of the root, so the negation is transferred from the root to the keyword; we remember to eliminate *Oscar Wilde* from any results we provide to the user. We also recognize lexicalized negatives, such as *hate*, *dislike*, *awful*, *stupid* etc. through a simple dictionary list of negative emotions. By maintaining a list of rejected candidates, we prune the results tailored to the user’s requirements.

Prepositions, Modifiers and Conjunctions

In order to maintain domain independence, we cannot attribute specific interpretations to function words, which are often overloaded. We use prepositions as a clue merely signalling relationships between two keywords (nouns or verb and noun). This relationship does not need to be identified; we use the knowledge base to detect this. For example, if the user says “*I want books by Dan Brown*”, we identify that “*books*” and “*Dan Brown*” are the keywords, and are related. A reference to a topic in ‘*What books on psychology are available?*’ is identified in a similar way, as related keywords “*books*” and “*psychology*”. Useful modifications through adjectives include cases like “*I want funny books*” or “*Do you have scary books?*”.

And and *or* are treated somewhat similarly. If a user says “*I like magic and animals*”, if there are no books which are about both *magic* and *animals*, we would still want to return some results, which are about magic or animals. This would be equivalent to the user saying ‘*I like magic or animals*’; in this case, all books which have references to either *magic* or *animals* are part of the results (see also Table 3.3). For *but*, it is a clear case of contrast; any keyword obtained from a *but* clause are included in the reject list. Consider “*I want books about vampires but not Twilight*” or *All books by Bernard Shaw but Pygmalion*”.

Relative Clauses Consider “*I would prefer books which have some mystery in them*”. Here the user is looking for *books*, which is the argument of the main verb *prefer* and the user specifies that the books need to *have* (secondary verb) *mystery* (object of secondary verb). Note that the prepositional argument here is *them*; since it is a demonstrative pronoun, it can be discarded as irrelevant. Using dependency relations, it is possible to extract the information that our keywords *books* and *mystery* are therefore related.

In case of *but*, the same assumption of contrast applies; we simply identify the relevant keyword as a reject qualification and eliminate it from the results (“*I would like books which are about traveling but not those which are focused on Europe*”).

3.3.2 Multiple Sentences

Me: I like science fiction. Maybe Asimov.	Me: I like Tom Hanks.
System: I found these under Science Fiction: <i>H. G. Wells</i> <i>Jules Verne</i> <i>Isaac Asimov</i> <i>George Orwell</i> <i>Aldous Huxley</i>	System: I think these movies are relevant: <i>Catch Me If You Can</i> <i>Saving Private Ryan</i>
I found these under Isaac Asimov: <i>I, Robot</i> <i>Foundation Series</i> <i>Caves of Steel</i>	Me: Recommend some Matt Damon movies.
What do you think?	System: I think these movies are relevant: <i>Saving Private Ryan</i>
Do you want books from the above?	Me: I like Hanks. I don't like Damon.
Me: I like science fiction. I think Asimov is awful.	System: I think these movies are relevant: <i>Catch Me If You Can</i>
System: You could try: <i>H. G. Wells</i> <i>Jules Verne</i> <i>George Orwell</i> <i>Aldous Huxley</i>	
What do you think?	
Do you want books from the above?	

Table 3.1: Multiple Sentence Inputs

We have a similar assumption here, that the user is likely to state their primary requirement first, followed by further qualifications. Since we add any keywords which have been associated with negative words to a reject list, and simply add any other keywords to a list of possibilities, the explicit order does not really matter. However, it is assumed that the set of sentences which occur together are related; which means that any qualifications that are made are applied collectively.

Consider “*I like books with humor. I don't like Wodehouse*”. Here, we first identify humor as a keyword from the first sentence. The structure of the ontology classifies authors under genres. In this case we return a list of authors classified under *humor* to the user. The next statement is assumed to be related to the first; it specifies that the user does not like the author *Wodehouse*. This keyword is added to the reject list. When the list is specified to the user, we search for and omit *Wodehouse* from the list (Table 3.1).

Through the dialog for this user, the keyword *Wodehouse* stays in the dialog manager's reject list, so that any future cases are also eliminated. This also helps us omit references in cases where it is not immediately applicable. For example, if the user input was “*I like books with humor. I don't like the Jeeves stories.*”, we would add that to the reject list, and if the user chose to look at books written

by *Wodehouse* in a future dialog, we omit the books which are about *Jeeves*. Some other pragmatic instances for elimination can also be considered (“*I have already read the Jeeves stories*”).

Similarly, it is also possible to answer questions which are more descriptive in style. Consider, ‘*I am trying to find a poem I read back in school. There’s a man who shoots a bird or something and things start to go wrong. I think he was a sailor.*’ We get a set of related keywords from each utterance: *poem-school; man-shoots-bird; sailor etc.*

3.3.3 Identifying Relationships in the Knowledge Base

Given a set of related keywords, the system searches through the knowledge base and matches them to the entities it finds there. In order to make the search efficient and the results relevant, the search is prioritised. For example, the *x-of* relationships are searched for matches first; next the search moves through the entities through the hierarchy in order, ending at tags.

So, in our “*I want books by Dan Brown*”, we found that “*books*” and “*Dan Brown*” are the related keywords; the system finds *books* is an *x-of* relationship as part of the knowledge base, and *Dan Brown* is an entity under *author* which has an *x-of* relationship with *books*. The system therefore returns a list of books under the author *Dan Brown*. In the case of ‘*What books on psychology are available?*’, related keywords “*books*” and “*psychology*” are identified as an *x-of* and an entity under either *genre* or *tag* respectively, and the system similarly returns books classified or tagged as *psychology*.

In case of multiple sentences, the keyword relationships are identified individually from each sentence and then collated. This helps us retrieve a list (“*I like books with humor*”) and then eliminate unwanted results (“*I don’t like Wodehouse*”); or alternatively narrow down possibilities. Consider “*I am trying to find a poem I read back in school. There’s a man who shoots a bird or something and things start to go wrong. I think he was a sailor.*”. We identify *poem* as a *genre* and *school* as a *tag* or perhaps a *title*; once we add *man/shoot/bird/sailor* we continue to narrow the possibilities down and perhaps finally find all of them as *tags*. The system then returns the relevant title (*Rime of the Ancient Mariner*).

3.3.4 Dialog Flow

The hierarchy of the knowledge base guides the flow of the dialog. The dialog manager traverses the hierarchy and locates itself on one of the nodes, and uses the meta information of the node (*genre/author*, etc) to frame its replies or ask further questions. If user input is unclear and the manager cannot locate itself in the knowledge base, the system asks the user to rephrase.

Typically a user’s input is processed, the keywords and the rejects identified, and the relevant results are displayed. The user may then choose to move up or down the hierarchy, or may change the topic entirely. This is identified by the break in the chain of the path being followed that is, if the user is not simply moving up or down the path but breaks the hierarchy chain such that the dialog manager must relocate itself in the knowledge base, then the topic (or user goal) is assumed to have changed.

Consider a user who asks for some *poetry*. The user is presented with a list of authors, say *Yeats*, *Shelley*, *Wordsworth* and *Coleridge*. The chain is now from *genre* to *authors*. Next, the user may choose one of these authors, say *Wordsworth*. Then a list of poems written by *Wordsworth* is presented to the user. Now the chain goes from *genre* to *authors* to *poems*, and so on.

Similarly, a move up the hierarchy is also possible. A user may say that he wants books like, say, *The Time Machine*. The system may then present the user with a list of books written by the same author, i.e., *H. G. Wells*. The chain has now formed from *books* to *authors*. The user may express interest in other authors who write similar books; the chain then moves up to *genre* and the user is presented with a list of *authors* who write *Science Fiction*, and so on.

3.3.5 Topic Change

System: What would you like me to look for?
Me: I want to try fantasy fiction.
System: I've found Fantasy. I found these about/under Fantasy : <i>J. K. Rowling</i> <i>Christopher Paolini</i> <i>George R. R. Martin</i> <i>Jonathan Stroud</i>
Me: Do you have books by Rice?
System: I've found Anne Rice. I found these about/under Anne Rice : <i>The Vampire Chronicles</i> What would you like me to look for?
Me: Go back to fantasy.
System: I found these about/under Fantasy : <i>J. K. Rowling</i> <i>Christopher Paolini</i> <i>George R. R. Martin</i> <i>Jonathan Stroud</i>

Table 3.2: Books: Topic Change

The user may explicitly jump around the hierarchy, making the dialog mixed-initiative (Table 3.2). For example, the user may choose to go back to the list of authors under a genre after being presented by a list of books (“*go back to fantasy*”) or the user may skip a level by directly specifying the book he’s interested in when presented with the authors (“*do you have The Adventures of Tom Sawyer by Mark Twain?*”). At any point, if the user does not move up or down from the current location in the hierarchical structure, then the chain is broken. A change in user goal is assumed to have occurred, and

System: What would you like me to look for?
Me: I want Neil Gaiman and Terry Pratchett books.
System: Do you want an intersection of Neil Gaiman and Terry Pratchett?
Me: Yes.
System: Would you like Good Omens?
System: What would you like me to look for?
Me: I like Jules Verne and Isaac Asimov.
System: Do you want an intersection of Jules Verne and Isaac Asimov?
Me: No.
System: I found these under Jules Verne <i>Twenty Thousand Leagues Under the Sea</i> <i>Around the World in Eighty Days</i> <i>Journey to the Center of the Earth</i> I found these under Isaac Asimov: <i>I, Robot</i> <i>Caves of Steel</i> <i>Foundation Series</i> What do you think? Do you want books from the above?

Table 3.3: Books: Clarification dialog

the dialog manager relocates the current reference node in the hierarchy. The chain is restarted from the location which is specified by the keywords obtained from the user’s input.

3.3.6 Ambiguities and Clarification dialog

In order to resolve certain inherent ambiguities, the system engages the user in a simple clarification dialog. For example, if the user asks for *books with animals in them*, the keyword relationships set up may match with books with *animals* in the title (*Animal Farm*) or generally books tagged with *animals* (*Harry Potter*, *Black Beauty*, *Animal Farm*). In this case, the system will ask the user by presenting each option in order of priority until the user accepts the results.

Queries such as *books by Dan Brown and Ayn Rand* can also be considered ambiguous in the sense that the user may be looking for an intersection (books written by *both Dan Brown and Ayn Rand* or a union *books by Dan Brown and books by Ayn Rand* (which is a case of ellipsis). In such cases, the system asks the user if s/he wants the results of the intersection, and if rejected, displays results of the union (see Table 3.3). In case the system does not find any results for intersection, the system specifies so and automatically presents results of the union.

The system only looks for negative indication in clarification (*no, I don't think so, etc.*) in order to be able to accept a wider range of positive indications (*yes, alright, okay, sure etc.*).

3.4 Domain Independence

Me: I like science fiction.	System: How can I help you?
System: I found these under Science Fiction: <i>H. G. Wells</i> <i>Jules Verne</i> <i>Isaac Asimov</i> <i>George Orwell</i> <i>Aldous Huxley</i>	Me: I like science fiction.
What do you think? Do you want books from the above?	System: I found these under science fiction: <i>Steven Spielberg</i> What do you think? Do you want movies from the above?
Me: I will try Isaac Asimov's books.	Me: I will try Steven Spielberg movies.
System: I found these under Isaac Asimov: <i>I, Robot</i> <i>Caves of Steel</i> <i>Foundation Series</i>	System: I found these under Spielberg: <i>Indiana Jones</i> <i>Schindler's List</i> <i>Jurassic Park</i> <i>Catch Me If You Can</i> <i>Saving Private Ryan</i>
Me: What is in the Foundation Series?	Me: Please tell me about Indiana Jones.
System: I've found a book series: <i>1. Prelude to Foundation</i> <i>2. Forward the Foundation</i> <i>3. Foundation</i> <i>4. Foundation and Empire</i> <i>5. Second Foundation</i> <i>6. Foundation's Edge</i> <i>7. Foundation and Earth</i>	System: I've found a movie series: <i>1. Raiders of the Lost Ark</i> <i>2. Temple of Doom</i> <i>3. Last Crusade</i> <i>4. Kingdom of the Crystal Skull</i>

Table 3.4: Books, Movies: Moving through the Hierarchy

The system utilises the structure of the data to recognise goals and paths; there is no domain knowledge involved in the dialog manager itself, or in fact any other module. The knowledge base is therefore replaceable, and the system is able to work in the same way for a different domain. In order to demonstrate this, examples are provided in Tables 3.4 and 3.5 to show that the system works just as well with a similarly structured knowledge base for a different domain. Here, in the movies domain the *x-of* relationships are in a hierarchy consisting of *genre, director, title, actors, plot points*, etc. For the restaurant domain the *x-of* relationships are *location, cuisines, name, menu, description*, etc.

System: How can I help you?
Me: I like chinese food.
System: I found these under chinese food: <i>RESTAURANT-16</i> <i>RESTAURANT-24</i> What do you think? Do you want menus from the above?
Me: I will try RESTAURANT-24. System: I found these under RESTAURANT-24: <i>Hakka Noodles</i> <i>Chicken Fried Rice</i> <i>Chilli Mushroom</i> <i>Chicken 65</i>

Table 3.5: Restaurants: Moving through the Hierarchy

3.5 Evaluation

3.5.1 Accuracy

Type of Input	Accuracy
Single line	88.00%
Relative clauses	66.66%
Multiple lines	100.0%

Table 3.6: Survey Query Evaluation

Category	Woz Benchmark	Avg. Rating out of 5
Relevance	4.625	4
Correctness	4.5	3.5
Overall	4.25	3.5

Table 3.7: Subjective Evaluation

The system was tested on a set of questions for the books domain obtained through an online survey. A human evaluation of the system's responses was conducted based on relevance, appropriateness and correctness. The evaluation is based on whether the intended keywords and any associated constraints are correctly identified. A result is considered to be correct if the relevant or expected output is returned. An overall accuracy of 88% was achieved over 50 questions (Table 3.6).

A second evaluation was carried out to test the dialog system qualitatively by means of user responses to questions in a Likert-scaled questionnaire; 8 participants were asked to interact with the system (over about 8-10 turns) and rate it on a scale of 5 on the basis of relevance (whether they thought the system understood their goals), correctness (compare expected response with actual system response) and an overall rating (navigation, task completion, etc). A Wizard-of-Oz benchmark was also collected. The results are in Table 3.7.

3.5.2 Error Analysis

There are many errors introduced due to ambiguous structures, grammatically unsound instances, etc. Parsing errors, like wrong identification of root, incorrect argument assignment, etc. also introduce significant error as the system is unable to identify the right keywords.

Parsing Issues: There is a prepositional phrase attachment problem - consider the following queries:

- (a) *Do you have Catcher in the Rye?*
- (a) *Do you have Of Mice and Men in the library?*
- (b) *Do you have Body in the Library?*

In (a), the prepositional phrase *in the Rye* is part of the title and important information; in (b) *in the library* is irrelevant information. It may also not be possible to ignore this phrase on the basis of the domain, as we can see from (c) where the title of the book itself has *in the Library* in it. In such cases, we may end up including irrelevant keywords and retrieving wrong information. Similar problems also occur when the book titles are phrases or sentences themselves (*'Do you have Where the Wild Things Are?'*) which are considered as part of the sentence by the tagger and assigned incorrect roots or copula tags.

Pragmatics: Consider *'Which is the most popular dish in RESTAURANT-2?'*. Converting *'the most popular'* to mean that we need to group by dish name for the restaurant, then look at the ratings, frequency of orders or other relevant information, is hard to do even if such information is available in the knowledge base, and such an inference includes information not present in the query; the metrics for this are often domain specific. Also, the thresholds of such subjective opinions may differ.

Recognizing the User's Goal: Consider the example, *'What other books are there have that were written by the author of The Old Man and the Sea?'*. Here, we need to first identify the *author* of the *The Old Man and the Sea* as Hemingway, and then retrieve books written by Hemingway. This retrieval may be complex and hard to identify.

3.6 Future Work

Pre-processing problems involving spelling and grammatical errors, synonyms, missing arguments, abbreviations, etc. need to be handled in order to make the system practically useful. Also, the words must be lemmatized and replaced with a mapped, representative synonym that is part of the knowledge base. A chunker or an NER can be used to identify complete constraints with functions words which the parser might separate (like *'Harry Potter and the Goblet of Fire'*). Ellipsis and anaphora issues need to be identified in depth.

A more robustly domain-independent system which can work despite significant changes in the knowledge base structure must be developed. For example, the system should be capable of handling data having complex networks and interconnections among the entities (instead of a simple hierarchy as we have used).

A better method for analyzing the positively and negatively associated words expressed in queries can be found, instead of a simple dictionary. Keeping track of both positive and negatively associated keywords could prove helpful for providing suggestions.

A corpus with a wide variation in the pattern of questions, along with questions in a statement form must be analyzed and considered, in order to make the system more accurate for generic queries. The usefulness of the information gathered through this method and the ease of mapping to a data representation must also be studied. More rigorous testing needs to be done for the descriptive questions.

Chapter 4

Domain Independent Keyword and Constraint Identification for Question Answering

4.1 Introduction

In this chapter, we discuss in detail the system developed for the identification of keywords and constraints in natural language queries that forms a part of the domain independent dialog system. For solving this particular problem, we focused on question answering as a subtask of a dialog system.

In the example in Table 4.1, the questions ‘*Who takes the NLP course? / Who has taken the NLP course?*’ could both be asking about the professor who teaches the course or the students who usually register for it. Although this ambiguity cannot be resolved without additional context, we hope to at least extract the information that the user in this case is looking for a person or persons (indicated by *who*), who is the agent of *take* where the course name is *NLP*. The keywords *who*, *take* and *NLP* are extracted with the help of universal dependency tags produced by the Stanford Parser; the association of *take* with either *register* or *teach* and the identification of *NLP* as a *course name* would require domain specific and in fact knowledge base specific information.

Q: Who takes the NLP course?

<< find: professor; constraint: NLP >>
take = teach

Q: Who has taken the NLP course?

<< find: students; constraint: NLP >>
take = register

Table 4.1: Example of Ambiguity

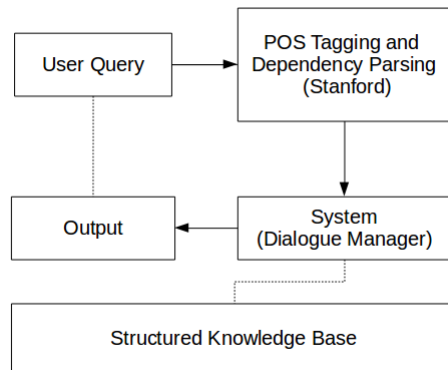


Figure 4.1: System Architecture

4.2 Our Approach

Our emphasis was to try and identify the objective of a user’s query, that is, the data the user is looking for and the constraints pertaining to this data. The idea was to do this using as few tools as possible. We only used the Stanford POS Tagger and the Stanford Dependency Parser for the implementation of our system (see Figure 4.1).

We assumed that the queries are free of errors and are grammatically sound, and that all required arguments are present in the query. We also did not attempt to resolve abbreviations and ambiguities.

The system is able to extract information from ungrammatical queries in certain cases where the Stanford Parser is able to generate a fairly accurate dependency parse. Using this information, we can map to a knowledge base as required. We used the hierarchically structured knowledge base for our dialog system, but a mapping to a database through an SQL query is also a possibility.

4.2.1 Motivation behind using Dependency Relations

The advantage of using dependency relations is that they are syntacto-semantic relations. This makes them word-order independent. The same question formulated in different ways can lead to similar dependency relations (such as a question in active/passive voice); this allows us to easily group similar user queries without having to anticipate all the possibilities.

Dependency relations also give us an idea of the relationship between the words and therefore the information the user is looking for and the constraints. This information cannot be obtained from simple syntactic parsing, such as Phrase Structure Trees, as they are highly dependent on the word order.

4.2.2 Dealing with Parsing Errors

After obtaining the dependency relations and parts of speech of a given query, we tried to construct some simple understanding of the query. It was found that the Stanford Dependency Parser does not always tag the main verb of the sentence as ROOT, and tends to tag question words or sometimes other nouns in the query as ROOT, especially in queries containing a copula. This also leads to identification of the direct object and other relations incorrectly, thereby making any mapping using specific tags inaccurate.

However, the overall structure identified between the words of the sentence is usually correct. In order to avoid the errors introduced by faulty tags, we crosschecked the assigned tags with the part-of-speech of the word, and in case of any discrepancy, we disregard the tags assigned to the words. For example, in case of questions like ‘*What is X*’, the parser may tag *What* as the root. Since *What* would have a POS tag of WP, we disregarded the ROOT tag, and instead used only the structure of the sentence in terms of word dependencies, along with POS information.

This approach for dealing with parsing errors is only useful for short questions; for longer questions with multiple verbs and nouns significant information is lost. This means that often dependency relations are missing and wrong tags are assigned to words, which introduces significant error.

4.2.3 Processing the Parser Output

4.2.3.1 Prepositions and their Arguments

In order to find relevant information, the system looks for the children of the root that are nouns, adjectives pertaining to nouns, prepositions and their children, and so on.

Prepositions such as ‘*by*’, ‘*on*’, ‘*of*’, etc. provide useful information about the relations and constraints. For example, although ‘*on*’ might also have a locational meaning attached to it, in the context of a library QA we can assume it shows reference to a topic (‘*What books on image processing are available?*’); similarly for the prepositions *about* and *under*.

Prepositions such as *in* and *at* often convey spatial or temporal information. Their noun dependents, along with the prepositions themselves, are considered to be constraints.

Stanford parser identifies the preposition *by* and its dependent as agents; this information is also added to the constraint list. For example, consider ‘*What are the books written by Ayn Rand?*’. Here the user is looking for books where the *author* is *Ayn Rand*. The preposition *by* here indicates that *Ayn Rand* is the agent of ‘*write*’, which in the domain of library translates to ‘*author*’.

In fact, in almost all cases, the noun dependent of the preposition turns out to be a crucial constraint. However, there are a few cases in questions like ‘*What books by Jeffrey Archer are available in the library?*’, that have to be identified as irrelevant information since *in the library* is technically implied and provides us with no useful information – and filtered out. But this is a domain specific assumption, and there is no way to judge the relevance otherwise.

4.2.3.2 Question Words

We looked for the question words in the query if any, and used them to further identify the user's intent. For example, queries using *'who'* are looking for a person (in case of a library corpus, perhaps the author, or in case of the courses domain, perhaps a student or a professor), queries using *'where'* are looking for a location, and so on.

These interpretations are domain dependent, so we could simply associate *'who'* with a person and *'where'* with a location, and have them independently converted to student/author and shelf number and so on in the stage where the keywords are being mapped to a knowledge base.

In the case of *'which'*, it appears as a determiner in questions, and the noun it modifies can be directly considered to be the objective of the question (*'Which book is...'* or *'In which year...'*).

We also note the presence of a construction *'how many'*, which is looking for a numerical answer, which overrides any other objective identified in the question (which is relegated to a constraint).

Questions with *'how'* or *'why'* are not considered, as they would typically elicit subjective answers. However, questions themselves are not classified. Separate rules are required for questions without question words, which use modals or auxiliary verbs for question formation.

4.2.3.3 Yes/No Questions

We identified the questions which are formed using constructions of auxiliary and modal verbs, which are looking for a yes/no answer. Such questions are structured slightly differently and require a different set of rules. Instead of looking for an objective - the answer type is clearly known to us here - we considered the *nsubj* (nominal subject), the *dobj* (direct object) and any constraints identified, all together as constraints.

This could introduce some difficulty during mapping to a knowledge representation, but the simplicity lies in the fact that if there is an instance that satisfies all the listed constraints, we can return an answer of *yes*, else we can return a *no*.

However, there are illocutionary implicatures to think about. For example, consider *'Do you have research papers related to NLP?'* Here, the answer is technically yes or no, but the user really wants a *list* of research papers related to NLP, as opposed to a question like *'Is the Fountainhead available for issuing?'* Since this distinction is pragmatic, we did not attempt to resolve it.

4.2.3.4 Modifiers and Conjunctions

In order to find constraints, first we look at all the modifiers of the *nsubj/dobj* that have been identified. Any adjectives associated with the *nsubj/dobj* are considered to be constraints.

Typically, the *amod* (adjectival modifier) tags are related to the words that directly modify the word and should be part of the left side of the query equation (such as **latest** *edition*) and the words tagged *nmod* (noun modifier) are part of the right side of the query equation (such as **Digit** *magazine*). The idea of a query equation is explained in further detail in section 4.

We grouped the nouns related through the compound tags (such as *Natural Language Processing*, *Data Structures*, etc). This kind of grouping may fail in certain cases where conjunctions are involved for example, *SSAD and Project* is one single course name, but is treated as two different words by the parser, such that any preposition preceding it has both of them separately as children. For better grouping of such titles, a chunker may be used.

4.2.3.5 Temporal Information

Certain temporal pragmatic information for example, deictic constructions such as *last week*, *till date*, etc, must be identified and processed.

Consider ‘*Which is the latest edition of Digit magazine available?*’. Here, latest would require to be interpreted as

```
<<find edition magazine=Digit, date=max(date)>>
```

Here we see that though *latest* is a modifier for edition, it really needs to be about the date. If the modifier had been *oldest* or *first*, we should be looking for *date = min(date)* instead. Specific constructions (such as ‘*in the year 2000*’) are taken care of as part of the information extracted from prepositions and their dependents.

4.2.3.6 Verbs and their Modifiers

We did not consider verbs and their modifiers since we place very little emphasis on the information provided by verbs, except in the case where agency of an act is a constraint, like in the case of ‘*Who wrote The Fountainhead?*’ where we are looking for a person who is the agent of the action ‘*write*’; here the verb provides significant information that tells us that the user is looking for the *author*.

In case of questions that do not use question words, and instead use constructions such as ‘*List X*’ or ‘*Show me X*’, the requisite information obtained by the system will be the same, and these verbs do not add much value to the information.

In the cases where the verb is considered important, significant information provided by adverbial modifiers specially negations must be specially considered. The problem here is that the negation itself would be associated with the verb, but the conversion of the query into the keyword and constraints format requires the negation to move to a different keyword. Consider, ‘*List all the books which are not written by Ayn Rand*’. Here, the action is ‘*not written*’, and the *neg* tag is associated with written; however, the keyword and constraints must look like: *find books, author = not(Ayn Rand)*.

4.2.3.7 Example

Consider the query, ‘*List the students registered for honours projects under Professor XYZ*’.

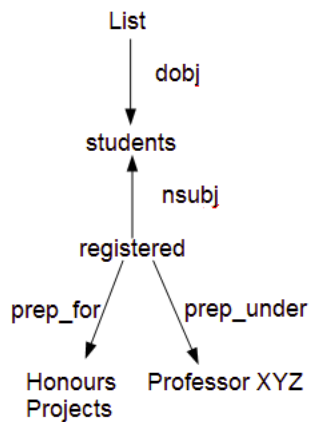


Figure 4.2: Dependency Tree of a Sample Query

A simplified dependency tree constructed from the tags provided by the Stanford Dependency Parser would look like Figure 4.2. In this case, the information the user is looking for (*students*) is the direct object, and the constraints are directly related to it.

In order to identify this main objective of the question, we identified the direct object of the main verb (root) of the question, tagged as *dobj* by the Stanford parser. In certain cases, such as passive constructions, this is often tagged as *nsubjpass* (subject, passive) (henceforth we refer to both as *dobj*). In cases without a marked *dobj*, we look for other noun dependents of the verb such as *nsubj* for constraints. In the case of questions with *who*, the requisite information is an agent, and the *dobj* is merely a constraint. This information is fairly obvious and can be inferred directly from the question word.

4.2.3.8 Domain Independence

This architecture of the system is domain independent, and any dependencies must arise only from a required mapping of the keyword and constraints to a knowledge system. For example, mapping a *who* to a *student* is specific to a *courses* domain, while mapping a *who* to an *author* would be specific to a *library* domain. For this reason, we left semantically significant question words such as *who* and *where* undifferentiated, and merely represented as *person* and *location* respectively.

4.3 Query Equation

Once all the primary keywords and the relevant constraints have been identified, we build a pseudo-SQL query like output to properly group the constraints. The final output is merely for representational ease. The output itself is a collection of keywords and related constraints, which can be mapped to any kind of knowledge representation. However, since an SQL-like representation would be the easiest to

Q: List the books written by Ayn Rand.

<< find books, written_by = Ayn Rand >>

Here *'written_by'* must be mapped to *'author'*.

Q: Which is the latest edition of Digit magazine available?

<< find edition, magazine=digit, date=max(date) >>

Here, *latest* is an *amod*, and is resolved to mean *max(date)*; the modifier *Digit* is an *nmod* and is therefore equated as *magazine=Digit*.

Q: Do you have research papers related to NLP?

<< find research papers, topic = NLP >>

Here *related to* is being interpreted as *topic*; in fact the preposition *'to'* by itself is used to arrive at this, so that the dependence on having to consider all possibilities (*related to*, *relevant to*, etc.) is reduced.

Q: What books on socialism are available in the library?

<< find books, topic = socialism >>

Here *'in the library'* is ignored, as it should be, but this is a domain specific choice that is particularly filtered out. In any other case, it may appear as an additional constraint (as *location=library*, for example)

Table 4.2: Query Equations

Corpus	Accuracy
Library Domain	72.72%
Library Domain without questions with relative clauses	85.7%
Course-management Domain	61.1%

Table 4.3: Results

understand, we chose to demonstrate the output in that format. Since it is not really SQL, and the terms are not mapped to any relations, we choose to call it a query equation (mainly because we mostly equate some value as a constraint) .

Examples of query equations are given in Table 4.2.

As can be seen, the query equation is in the form of a pseudo-SQL query. It can be easily converted into a format that suits any knowledge representation, provided the relations that the terms map to are known. Consider the first example,

List the books written by Ayn Rand.

<<find books, written.by = Ayn Rand >>

This can be mapped to an SQL query, say a relation *books* with attributes *book number*, *title*, *author*, *publisher*, *year*, etc. as follows:

```
SELECT title FROM books WHERE author like 'Ayn Rand';
```

4.4 Evaluation

4.4.1 Accuracy

The system was tested on a set of 128 questions for library domain obtained through a survey, and another set of 150 questions on the courses domain created by [45].

Since the emphasis is on identifying the right constraints, the evaluation is based purely on whether the intended keywords and any associated constraints are correctly identified from the question. A question's objective is considered to be rightly identified only if both what the user is looking for and all the necessary information related to it are identified correctly.

As is summarized in Table 4.3, the accuracy of the system increases when the questions which have relative clause descriptions and therefore multiple verbs (e.g. *'What other books does the library have that were written by the author of the Da Vinci Code?'*) are not considered.

There is a significant drop in the accuracy when tested on a course management domain. The level of ambiguity in these questions was found to be higher, along with far fewer preposition dependent constraints and more relative clause questions.

4.4.2 Error Analysis

Apart from parser errors, there are many other errors introduced due to oversight, ambiguous structures, grammatically unsound sentences, etc. The errors are similar in nature to the ones seen in Chapter 3 as part of the dialog system.

4.4.2.1 Conjunction - ‘and’

One of the significant problems occurs with conjunctions such as ‘*and*’ that link two relevant words together. Consider ‘*Harry Potter and the Philosopher’s Stone*’; in this case, *Harry Potter* and *Philosopher’s Stone* are parsed as two separate arguments linked by a conjunction, but they are in fact part of the same title. In such cases, it is difficult to tell if the user has two different constraints (‘*I want books by Ayn Rand and Dan Brown*’) or just one. If we consider them separately, a search for *Harry Potter* would list all the seven books of the series as relevant, instead of just the first book. In certain cases, only one of them might be relevant; in such cases there has been no way to resolve the issue. Also consider cases with multiple unrelated constraints separated by conjunctions: for example, ‘*Which students are the TAs of NLP?*’ and ‘*Which students are TAs and are registered for NLP?*’.

4.4.2.2 Syntactic Parsing Issues

Constructions such as dangling prepositions (‘*How many credits is NLP for?*’, ‘*Which Y does X belong to?*’) lead to parser errors or unexpected dependencies that are not accounted for. Sometimes certain words are skipped by the parser.

More significantly, there is a PP attachment problem - consider the following queries:

- (a) *Who is the author of The Wind in the Willows?*
- (b) *Is The Fountainhead present in the library?*
- (c) *Who is the author of Body in the Library?*

In (a), the PP *in the Willows* is part of the title, while in (b), *in the library* is irrelevant information, which cannot really be filtered domain specifically either, as demonstrated in (c) where *in the Library* is part of the title. Therefore, gauging the relevance of an argument is a challenge; there are too many possibilities that cannot all be taken into account individually.

This issue is not limited to prepositions alone; consider ‘*What is the ISBN of the book Talent is Overrated?*’, where the *is* belonging to the book title is recognized as a separate verb copula and is assigned arguments.

4.4.2.3 Pragmatics

Resolving pragmatic issues is also a challenge. Consider ‘Which is the most popular book on mathematics?’. Here, we can extract information on the modifiers as

```
<<find most(popular(book) , topic = mathematics >>
```

However, converting ‘the most popular’ to mean that we need to group by book title, then count the number of times each book has been *issued* and then return the one with the highest count, is quite a challenge; we need to infer that *popular* here is a reference to the *number of times a book has been issued*, which requires deriving information that is not present in the query.

4.4.2.4 Relative Clauses

Any arguments that are expressed through non-prepositional phrases, such as through relative clauses, which bring in additional verb and its arguments, are less likely to be identified correctly as relevant constraints. The accuracy of the system improves when such queries are not considered.

Consider ‘What other books does the library have that were written by the author of the *Da Vinci Code*?’. There are two parts to the query; firstly we need to find the *author* of the *Da Vinci Code*, then find *books* written by that author, and also remove the book *The Da Vinci Code* itself and display the results. The query is complex and there are pragmatic issues as well.

4.5 Statistical Methods (CRF)

We tried to improve the accuracies using statistical methods. Specifically, we approached the problem of keyword identification as a sequence labelling problem: we use Conditional Random Fields (CRF) to capture dependencies within the natural language query. By delexicalising the query we make it a domain-independent approach with a relatively small dataset.

4.6 Data

We collected data from three different domains for training and testing purposes.

4.6.0.1 Course Management Domain

The dataset created by [45] consists of 1000 queries for training and 558 for testing. We used the dataset in the same way to make comparison possible.

4.6.0.2 Library Domain

We collected around 128 queries in the library domain through a survey. We used these queries for testing.

4.6.0.3 GeoQueries250

We used the GEOQUERIES250 dataset [38] for testing to facilitate comparison as it is a commonly used and well-known query dataset.

4.7 Approach

Every word in the query is supplemented with its part-of-speech tag and dependency tag obtained from the Stanford POS tagger and the Stanford Parser respectively. The training data also includes the part-of-speech tag and the dependency tag of the parent word (in terms of dependency relation).

We used Conditional Random Fields to learn the labels since it provides a method to segment and learn sequence dependent labels, and has been shown to have advantages over HMMs and MEMMs [26].

4.7.1 Training Data

We used only dependency relations (for their syntacto-semantic information) and part-of-speech tags as part of external/meta information.

The data consists of current dependency tag D_t , current POS tag P_t , parent-POS tag PP_t (POS tag of parent word through dependency relation) and a label indicating whether the current tag is to be selected (as a keyword) or discarded. The system will then use the corresponding features for processing, but the model itself is blind to the lexicon. The model can therefore predict labels for any set of dependency tags and POS tags belonging to any sentence, making it domain independent.

See Table 4.4 for an example training instance for the query "What are the assignments posted for NLP?", taken from the course management domain. The training data does not include the first column, i.e., the words in brackets are not part of training.

4.7.1.1 Labeling

Each word in the query is manually labeled as SELECT or DISCARD based on whether it is a keyword important for answering the query or not.

Word	D_t	P_t	PP_t	Label
(What)	dobj	WP	VBN	DISCARD
(are)	auxpass	VBP	VBN	DISCARD
(the)	det	DT	NNS	DISCARD
(assignments)	nsubjpass	NNS	VBN	SELECT
(posted)	root	VBN	root	SELECT
(NLP)	prep_for	NNP	VBN	SELECT

Table 4.4: Example of Training Data

4.7.1.2 Delexicalisation

In order to make our approach truly domain independent, the training data is delexicalised, i.e., stripped of all actual words. The input training data (before it is processed according to templates) therefore only consists of dependency tag and POS tag of current word and the dependency tag and POS tag of the parent word, plus the label, as in Table 4.4 (excluding the first column). The test data is similarly labeled. The algorithm actually learns the correlation between whether a certain dependency tag and POS tag, along with a few other features, are likely to be a keyword or not; the word itself is irrelevant.

4.7.2 Question Answering

The labeled/identified keywords can be formed into a query equation: using the dependency tags as a reference, the relationships between the keywords can be used to obtain important information to answer the query. For instance, from the example in Table 4.4, the selected keywords and their dependency relations can be further structured like so:

$$posted(assignments, for_NLP)$$

. This gives us some amount of semantic information that can be used to answer the query; it can be converted to SQL or any other form as required by the knowledge base for question answering, as previously discussed.

4.8 Experiments

We conducted some experiments in order to determine the best template (features) for the statistical model to learn from. If the current dependency tag is D_t , then the previous tag (dependency tag of previous word in the query) is D_{t-1} , the next tag (dependency tag of the next word in the query) is D_{t+1} , and so on. These features are in addition to the D_t , P_t (current POS tag) and the PP_t (current

Parent-POS tag) that are used for every instance. A template having x/y indicates a combined feature of x and y .

Template No.	Unigram Features	Combined Features	Course Domain	Library Domain	GQ250 dataset
1	D_t P_t PP_T	- -	88.51	79.14	96.74
2	D_t P_t PP_T	$D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-1}/P_t, P_t/P_{t+1}$	90.58	78.08	96.87
3	D_t, D_{t+1} P_t, P_{t+1} PP_T	$D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-1}/P_t, P_t/P_{t+1}$	90.28	78.51	96.67
4	D_{t-1}, D_t, D_{t+1} P_{t-1}, P_t, P_{t+1} PP_T	$D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-1}/P_t, P_t/P_{t+1}$	90.36	76.17	97.13
5	D_t P_t PP_T	$D_{t-2}/D_t, D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-2}/P_t, P_{t-1}/P_t, P_t/P_{t+1}$	90.65	83.19	93.15
6	D_t P_t -	$D_{t-2}/D_t, D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-1}/P_t, P_t/P_{t+1}$	90.06	78.29	95.5
7	D_t P_t -	$D_{t-2}/D_t, D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-2}/P_t, P_{t-1}/P_t, P_t/P_{t+1}$	90.65	78.08	92.43
8	$D_{t-2}, D_{t-1}, D_t, D_{t+1}$ $P_{t-2}, P_{t-1}, P_t, P_{t+1}$ PP_T	$D_{t-1}/D_t, D_t/D_{t+1}$ $P_{t-1}/P_t, P_t/P_{t+1}$	90.21	75.95	96.28

Table 4.5: Feature Templates and Accuracies

We experimented with different templates in CRF++ to find the most optimised template for learning. The accuracies for different templates for the different domains are given in Table 4.5. The precision, recall and F1 scores for each template and domain are given in Table 4.6.

As highlighted in Table 4.5, template number 5 that uses combined features of two previous dependency and POS tags and one next dependency and POS tag (window of -2 to +1 for combined features) along with the current dependency, POS and parent POS tag performs best for the course management and library domains. Template number 4 that uses combined features of one previous dependency and POS tag (window of -1 to +1 for combined features) in addition to the previous, current and next dependency and POS and current parent POS tag (window of -1 and +1 for unigram features) performs the best for the GEOQUERIES250 dataset.

We can see that, in general, there is a correlation between a keyword, its dependency tag, its POS, the dependency and POS of the previous word, the POS of the parent word in terms of dependency relations, and the dependency and POS of the next word.

Template No:		1	2	3	4	5	6	7	8
Course									
	Precision	0.9107	0.9223	0.9184	0.9223	0.9262	0.9262	0.9275	0.9223
	Recall	0.8922	0.9152	0.9137	0.9117	0.9132	0.904	0.9122	0.9094
	F1	0.9014	0.9188	0.9161	0.917	0.9197	0.915	0.9198	0.9158
Library									
	Precision	0.808	0.8348	0.8437	0.8214	0.9013	0.7964	0.8035	0.8482
	Recall	0.7702	0.742	0.744	0.7215	0.8204	0.7639	0.7563	0.7089
	F1	0.7886	0.7857	0.7907	0.7682	0.8589	0.7789	0.7792	0.7723
GEOQUERIES250									
	Precision	0.9643	0.9667	0.9619	0.9726	0.8906	0.9488	0.8882	0.9524
	Recall	0.9771	0.9771	0.9782	0.9761	0.9752	0.9696	0.9726	0.9804
	F1	0.9706	0.9719	0.97	0.9743	0.931	0.9591	0.9285	0.9662

Table 4.6: Precision, Recall and F1 scores for each template and domain

4.9 Comparison with the Rule Based System

Domain	CRF	RBS
Course Management	90.65%	61.1%
Library	83.19%	72.72%
GEOQUERIES250	97.13%	-

Table 4.7: Accuracy of the CRF Model compared to the Rule-Based System

In Table 4.7, we compare the best CRF accuracies against the accuracies obtained by the rule-based system. We reached high accuracies of 90.65% on the course management domain dataset, 83.19% on the library domain dataset and 97.13% on the GEOQUERIES250 dataset.

Also, from Table 4.6 we see that we comfortably exceed the F1 score of 0.5125 for keyword extraction using CRF set by [57].

4.10 Error Analysis

We see that the model performs fairly well on the test set of the same domain it is trained on (course management), but dips for the library domain. The dataset of the library domain is small compared to the others, which could be a contributing factor. The accuracies for the GEOQUERIES250 dataset are quite high, which may be due to the simple structure and non-ambiguousness of the queries. The course management and library domains have a fair number of complex queries which have relative clauses or multiple verbs; such queries are likely to have a higher incidence of parsing errors, which may contribute to an error in keyword identification.

A dependency relation-wise analysis shows that the highest number of errors occurred in the labeling of the *root* (main verb), *nsubj* (nominal subject of verb) and the *dobj* (direct object of verb) relations. A fair number of the *root* errors occurred due to parsing errors leading to a wrongly tagged *root*. Other errors occurred because of ambiguousness in quite a few queries where the main verb is a keyword (*teach*, *register*) and where the main verb is not a keyword (*list*, *give*).

Similarly, queries which have multiple *nsubj* and *dobj* or both contribute to the errors; these dependency tags are also most common for keywords. Also, in questions with a “What...” construction, *what* often gets tagged as the *nsubj* or *dobj*, also causing errors when it is wrongly labeled as a keyword.

Other miscellaneous errors involved some infrequent dependency relations such as *xsubj*, *rcmod*, etc.

4.11 Conclusions and Future Work

The system provides a simple framework requiring very little coding in order to extract relevant information from a user’s query. Pre-processing problems involving spelling and grammatical errors, synonyms, negations, missing arguments, abbreviations, etc. need to be handled in order to make the system practically useful.

In order to find complete constraints, such as the full name of the Professor or the full course name (like ‘*Linear Algebra and its Applications*’), dependency relations are sometimes inadequate, since the tags often resolve to separate determiners and other function words which are often part of the title. This information can be captured using a chunker. The chunks provided often correspond to the entire title or name, providing us with the complete constraint.

Conjunctions are a very common occurrence, and multiple arguments must be considered. However, it may be difficult to identify a more relevant argument from a less relevant one. This issue is resolved in the dialog system by using clarification dialogs.

To resolve intention recognition issues, such as in the case of *Do you have research papers related to NLP?*, which is technically a yes/no question but the user intention is really to get a list of research papers, a corpus that has been annotated with dialog acts can be used.

Our statistical system performs with a high accuracy in identifying the relevant keywords in all three different domains of course management, library and the GEOQUERIES250 dataset, using a training

dataset of only 1000 queries. This makes our approach efficient, easy to implement and truly domain independent.

Because the tagset of the dependency relations and the part-of-speech tags are more or less universally agreed upon and are the only external information used in our system, our approach does not require hand-crafted features that is different for each domain and can be universally used for any domain.

There is no need for re-annotation and re-training for every new domain that is needed; an existing robust model trained on a fair sized dataset can perform very well on all domains. A one-off training and its resultant model are therefore all that is required for keyword identification in any domain.

Since dependency tags are also arguably language independent, and will produce the same tags (adjusted for the tagset) for a similar sentence in any other language, our approach can also be a language independent solution in addition to being domain independent.

In order to make the approach both language and domain independent, universal dependency tags [36] that are consistent across several languages can be used.

The approach needs to be tested across more domains with more varied patterns of queries, especially complex queries with relative clauses/multiple verbs. The system may also be able to produce labels with more granularity, e.g. main keyword, additional constraint, etc.

Chapter 5

Evaluating Dialogs Based on Grice's Maxims

5.1 Introduction

In this chapter, we attempt to introduce a comparatively simpler and shorter measure of dialog evaluation based on the cooperative principles proposed by Grice [15]. We collected system generated dialogs from popular systems across the board and conducted a survey to gauge its effectiveness. The next subsection introduces Gricean maxims; Section 2 describes the survey; Section 3 presents the results of the survey and discussion and Section 4 and 5 conclude the paper and present scope for future work. The appendix at the end lists all the complete dialogs used in the survey.

5.1.1 Gricean Maxims

Cooperative principles, introduced by [15], describe how speakers act cooperatively to be mutually understood for effective communication. Grice divided the cooperative principle into four conversational maxims on which our survey is based¹:

1. **The maxim of quantity**, where one tries to be as informative as one possibly can, and gives as much information as is needed, and no more.
2. **The maxim of quality**, where one tries to be truthful, and does not give information that is false or that is not supported by evidence.
3. **The maxim of relation**, where one tries to be relevant, and says things that are pertinent to the discussion.
4. **The maxim of manner**, when one tries to be as clear, as brief, and as orderly as one can in what one says, and where one avoids obscurity and ambiguity.

For the sake of clarity, we will be using relevance interchangeably with the maxim of relation.

¹<https://www.sas.upenn.edu/~haroldfs/dracling/grice.html>

Dialog Type	No. of Dialogs	Dialog Numbers
Conversational	5	1,7,9,10,11
Task Oriented	3	5,6,8
Breakdown	3	2,3,4

Table 5.1: Distribution of Dialogs

Q		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11
I	Mean	2.016	3.306	2.612	3.387	3.919	3.854	3.306	2.645	3.612	2.854	2.5
	SD	1.047	0.897	1.150	0.911	0.874	0.972	1.033	1.041	0.981	1.099	1.082
	Med.	2	3	3	3	4	4	3	3	4	3	2
	Mode	1	3	3	4	4	4	3	2	3	3	2
II	Mean	3.258	3.887	2.87	3.661	4.451	4.209	3.451	2.758	3.806	2.967	2.677
	SD	1.213	0.851	1.152	1.054	0.917	0.977	1.111	1.223	1.198	1.039	1.302
	Med.	3	4	3	4	5	4	4	3	4	3	3
	Mode	3	4	3	4	5	5	4	2	4	3	1
III	Mean	2.516	3.758	2.225	3.58	4.596	4	3.274	2.419	3.709	2.435	2.241
	SD	1.036	0.881	1.062	1.033	0.756	1.040	1.203	0.897	1.121	1.065	1.035
	Med.	2.5	4	2	4	5	4	3	2	4	2	2
	Mode	3	4	2	4	5	5	3	2	4	2	2
IV	Mean	2.854	3.693	2.629	3.516	4.548	4.048	3.467	2.403	3.822	2.709	2.467
	SD	1.171	0.951	1.119	1.155	0.823	1.151	1.289	1.015	1.194	1.219	1.263
	Med.	3	4	3	4	5	4	4	2	4	3	2
	Mode	3	3	3	4	5	5	4	2	5	3	2

Table 5.2: Survey Results

5.2 Survey Description

5.2.1 Dialog Collection

The 11 dialogs that are part of the survey were collected from examples of user and system generated responses given in [7], [49], [42], [21], [20], [56], [39] and a report on a chatbot with personality². See Appendix for all the dialogs.

The dialogs are a mix from early systems like ALICE [50] and Eliza [52] with current state-of-the-art, and also a mix of task oriented and conversational dialog. We also included examples of dialog breakdown [21] which are unique cases in which though the system responses are relevant and possibly right, they do not make sense or contradict previous responses, making it hard to proceed with the dialog. A distribution is given in Table 5.1.

²<http://web.stanford.edu/class/cs224n/reports/2761115.pdf>

	I	II	III	IV	Overall
1	D5	D5	D5	D5	D5
2	D6	D6	D6	D6	D6
3	D9	D2	D2	D9	D9
4	D4	D9	D9	D2	D2
5	D2	D4	D4	D4	D4
6	D7	D7	D7	D7	D7
7	D10	D1	D1	D1	D10
8	D8	D10	D10	D10	D1
9	D3	D3	D8	D3	D3
10	D11	D8	D11	D11	D8
11	D1	D11	D3	D8	D11
Mean	3.09	3.45	3.15	3.28	12.99

Table 5.3: Dialogs Ranked by Mean Scores

5.2.2 Survey Questions

After presenting the user with the example dialog, the user is asked to rate the system performance on a Likert scale from 1 to 5 for 4 questions:

1. Is the system as informative as it can be and gives as much information as is needed and no more? (Maxim of Quantity)
2. Is the system truthful or does it give information that is false, to the best of your world knowledge? (Maxim of Quality)
3. Are the system responses relevant, i.e., does the system respond with things pertinent to the discussion? (Maxim of Relation)
4. Are the system responses clear, orderly and without obscurity and ambiguity? (Maxim of Manner)

5.3 Results

We received 62 responses to our survey. Each of the 62 raters rated all of the 11 dialogs. In Table 5.2 we provide the mean, standard deviation (SD), median and mode for each of the four questions for the 11 dialogs (D1-D11).

5.3.1 Discussion

We refer to the means per dialog per maxim as mean scores, and the means over all the dialogs per maxim as the maxim mean.

Table 5.3 gives the rankings of the dialogs by mean scores for each maxim. The rankings based on the sum of mean scores of all four maxims are given in the last column under overall. The last row

contains the means of ratings per maxim, and the overall mean of the summed mean scores. The top 6 dialogs that perform above the maxim means are highlighted. There is a clear split in all four cases, i.e., D7 and above perform consistently above the mean scores.

From Table 5.3, we see that the task-oriented dialogs D5 and D6 perform at the top, better than any conversational dialog, which is an observation consistent with [42].

5.3.2 Ranking Analysis

D1 is a simple case in which the system tries to continue the conversation by asking questions using phrases from the user's utterances; in D3 the system seems to understand the user's utterances and keeps the conversation going by bringing up related topics. D2 can be seen doing a bit of both, however, it ranks well above both D1 and D3, although both D2 and D3 are considered breakdown dialogs. Despite displaying some semantic understanding, D3 scores poorly on all counts except quantity, in which D1 brings up the rear. This is easy to understand, as D1 provides no response of substance, even if it is more relevant and clear/unambiguous.

D9 and D2 rank fairly high, but we can see from the difference in scores that D9 performs better in quantity and manner while D2 performs better in quality and relevance. D2 has comparatively few new responses (responses not based on repeating the user's utterance in a different form: quantity) and is a dialog marked as having a breakdown (manner). D9 however, is unable to answer the last question (quality) and brings up epistemology in a conversation which is mostly about food (relevance).

D4 and D7 perform somewhat in the middle, and D4 does consistently better than D7 on all four metrics. In D4, the system is relevant and clear, but produces breakdown dialog such that the user cannot proceed with the dialog. However, in D7, the system misunderstands the user's second utterance. Unfortunately, we cannot draw conclusions about whether the humour as perceived by the user in the first system response played any part in the performance ratings, or if users perceived the second system response as irony.

D10 has mostly muddled up dialog and ranks fairly low, however it outperforms D8, which is comparatively on track and somewhat on topic; users seem to be harsher on task-oriented dialogs that derail the user's goal.

D11 does poorly on all counts, and it is easy to correlate as its responses are repetitive, irrelevant, unclear and obscure. However, we notice that it performs better than D1 in quantity (it asks a question of some substance), better than D3 in relevance (D3 has multiple irrelevant replies) and better than D8 in manner presumably since D8's unclear responses are more glaring in view of its task-oriented nature.

5.3.3 Comparing the Means

The rankings based on the sum of all the four mean scores are more or less representative of the overall ranking (last column in Table 5.3). This shows that the mean scores can be summed up to form a

	Metric	p-values
D7 v. D1		
	Quality	0.3574
	Relevance	0.0003
	Manner	0.0065
D7 v. D10		
	Quantity	0.0199
	Overall	0.0023

Table 5.4: Significance of Maxim Mean as Threshold

meaningful overall score for comparison, since they are all measuring the rational cooperative principles that enable effective conversations³.

From the maxim means we can see that the dialogs have an overall higher score in quality, which means that the users think their utilisation of available knowledge was sufficient. Both relevance and quantity get poor overall scores, which is where the systems need to improve most. Since obscure or breakdown dialogs were comparatively less, the manner scores are in between.

To test if the maxim means can form a reasonable threshold between acceptable system responses and unacceptable system performance, we checked if there is a statistically significance difference between the scores of D7 and D1, and D7 and D10, since D7 performs above the means in all four cases, but D1 scores below the means in quality, relevance and manner and D10 performs below the means in the cases of quantity and overall. A p-value of less than 0.05 indicates a statistically significant difference. We see from Table 5.4 that all metrics except quality have a significant statistical difference between the scores above the maxim mean and below the maxim mean (highlighted). Using the maxim means as a threshold is therefore promising.

Despite having ratings for only 4 questions, we were able to make somewhat nuanced comparisons between the dialogs. The maxims are constructed in such a way as to also allow us to pinpoint where the system responses can improve.

5.3.4 Detailed Example

We discuss the scores of D5 and D6 in a little more detail. D6 is from a train timetable information agent that uses explicit confirmation and was found to have a high transaction success rate [7] as opposed to D5's more efficient implicit confirmation [49].

Dialog 5(sampled):

U: I want to go from Torino to Roma

S: Approximately what time of day would you like to travel?

³https://en.wikipedia.org/wiki/Cooperative_principle

U: What are the options?

S: Morning or evening.

Dialog 6(sampled):

S: Do you want to leave from Trento?

Yes or No?

U: No.

S: Where do you want to leave from?

U: I want to leave from Torino.

S: Do you want to leave from Torino?

Yes or No?

U: Yes.

D5 performs better than D6 in all four counts (with mean scores 3.919 and 3.854; 4.451 and 4.209; 4.596 and 4; 4.548 and 4.048 respectively) despite accomplishing the same task. There is an especially high difference in relevance and manner ratings (statistically significant p-values of <0.0001 and 0.0063 respectively).

This indicates that users do not like explicit confirmations despite them having a higher task success rate (as implicit confirmation may be more likely to generate errors or repair dialog) [7].

5.4 Conclusions

We see that a comparatively short survey based on Gricean maxims produces scores which help in comparing and ranking dialogs in such a way as to allow us to analyze the issues in their generation.

The question based on the maxim of quantity helps us identify if the system provides substantive responses.

The question based on the maxim of quality can be used to check if the system is faithful to the factual knowledge provided to it.

The question based on the maxim of relation helps us identify if the system is able to understand the user and therefore provide relevant replies.

The question based on the maxim of manner helps us identify if the system provides awkward or ambiguous responses; this helps us identify dialog breakdowns.

The mean scores obtained through the ratings also show promise in providing a benchmark value for acceptable dialog, i.e., dialogs that score below a certain threshold can be considered not good enough for use.

5.5 Future Work

The scores prove the maxims to be a good framework to compare dialog strategies for real world usage. On receiving the scores for a set of dialogs, we can automatically have them ranked and classified based on which maxim they fall short on, and could provide a way to do directed analysis of the dialogs.

Since a threshold based on the mean scores shows some promise in distinguishing good dialogs from poor ones, we need to explore if these judgements can be used to create baselines or benchmarks for dialogs in each of the four maxim categories.

By collecting scores from a large number of people over a more diverse and bigger set of dialogs may provide us with enough data to perform more rigorous statistical tests.

The agreement between the human judges must be computed for this purpose. If there is an acceptable amount of agreement, it may be worth exploring if these scores can be predicted through machine learning.

5.6 Appendix: Dialogs

Dialog 1 (D1):

S: Is something troubling you?

U: Yes, I'm trying to describe you for a research paper I'm writing. What should I say?

S: You are sure.

U: Yes, I'm sure.

S: I see.

U: What can I tell them about you?

S: You seem to need to tell them about me.

U: I absolutely need to. Please help.

S: That's quite interesting.

U: I agree.

S: You say you agree?

Dialog 2 (D2):

S: Good evening. Be careful to avoid heat stroke.

U: Sure.

S: O.K.

U: I am drinking enough water.

S: Do you run short of water?

U: Yes, I do, if I sweat.

S: Do you lose water if you sweat??

U: Yes, I do.

S: This is so.

Dialog 3 (D3):

U: You'd better stay at home, as it is very hot outside.

S: You have sweat on your face, don't you?

U: Indeed.

S: So you do.

U: I hate to sweat.

S: Your make-up gets smeared due to sweat, doesn't it?

U: I don't wear make-up.

S: I want make-up.

U: I don't want it.

S: Thanks for the meal!

U: Did you eat anything?

S: The make-up is appropriate.

Dialog 4 (D4):

S1: Its nice to go shopping alone.

U1: I agree. Thats nice.

S2: *Shopping takes time.*
U2: *Window shopping is also fun.*
S3: *Its fun to go shopping with somebody.*

Dialog 5 (D5):

U: *I want to go from Torino to Roma.*
S: *Approximately what time of day would you like to travel?*
U: *What are the options?*
S: *Morning or evening.*
U: *Are those departure times?*
S: *Yes.*
U: *I'd like to leave in the morning.*
S: *Train 702 leaves Torino Porto at 8 a.m.*
U: *Please reserve me a seat on that train.*

Dialog 6 (D6):

S: *Hello, This is Train Enquiry Service. Please speak after the tone.*
Which information do you need?
U: *I want to go from Torino to Milano.*
S: *Do you want to go from Trento to Milano? Yes or No?*
U: *No.*
S: *Do you want to leave from Trento? Yes or No?*
U: *No.*
S: *Where do you want to leave from?*
U: *I want to leave from Torino.*
S: *Do you want to leave from Torino? Yes or No?*
U: *Yes.*
S: *Do you want to go to Milano?*
U: *Yes.*
S: *At which time do you want to leave?*
U: *I want to travel in the evening.*
S: *Do you want to leave between 6 and 9 p.m.? Yes or No?*

U: *Yes.*
S: *There is a train leaving at 8:00 p.m.*

Dialog 7 (D7):

U: *Tell me a funny joke.*
S: *Did you hear the one about the Mountain Goats in the Andes? It was Ba a a a a ad.*
U: *Tell me another one.*
S: *I am the latest result in artificial intelligence which can reproduce the capabilities of the human brain with greater speed and accuracy.*

Dialog 8 (D8):

U: *Do you know what a firewall is?*
S: *A firewall is a specialized gateway designed to block or restrict certain types of data transfers on the Internet while permitting others.*
U: *Can you list different types of firewalls?*
S: *You should just download my software.*
U: *What software?*
S: *Interesting question.*
U: *How could I download your software?*
S: *A lot of practice.*

Dialog 9 (D9):

U: *I went on a big date last night.*
S: *Was it very expensive? Did you have a good time?*
U: *It was very expensive. I had a great time.*
S: *How did you feel about that? What happened to it?*
U: *I had the salmon.*
S: *Do you still have the salmon?*
U: *I ate it yesterday.*
S: *How did it taste?*
U: *It tastes very good.*
S: *Like chicken?*
U: *Not like chicken but chicken is good too.*

S: Oh I get it.

U: Do you like chicken?

S: Yes it is good especially baked.

U: How can you eat chicken if you are a robot?

S: That is a good epistemological question.

Dialog 10 (D10):

U: Your nose is a question mark, I think thats interesting.

S: Of course, whats your idea?

U: I think it fits you perfectly.

S: Let me see, his nose is still broke.

U: Do you want me to make your nose broken?

S: Ah, yes, why did you ask me for this?

U: Because I thought you might mind.

S: Well, I am not getting into specifics about what doctor treated him, how he was treated.

U: You have not been treated yet. That is in the

future.

S: Let me see, by the way, are you interested in movies?

U: Yes, I watch movies sometimes.

S: Great.

Dialog 11 (D11):

U: How are you?

S: I think so, any thoughts?

U: How are you doing?

S: I am fine, how are you, any thought?

U: Any thought?

S: Let me see, do you want to go to my friends party tonight?

U: No I have a lot of work to do. I cant go.

S: Everyday?

Chapter 6

Conclusions and Future Work

We built a domain independent dialog system based on a hierarchically structured knowledge base. We demonstrated the domain independence by testing it on three different domains of books, movies and restaurants. The system uses the structure of the knowledge base to guide dialog flow and engages in clarification dialog. Primarily, it also accepts multiple sentences as input and recognises negative or positive association to respond to the user accordingly. We showed the per question accuracy and a subjective evaluation from human evaluators. Our system performs fairly well.

We separately focused on the keyword identification module of the dialog system, and showed how the rule-based system is based on dependency relations and part-of-speech tags and the various relationships it can identify based on function words. We tested the rule based system on course management and books datasets and the results were promising. We improved the accuracy by training a CRF model on delexicalised data to label keywords, and it showed high accuracies in the domains of course management, books, and GEOQUERIES250.

We also introduced subjective evaluation metrics for artificially generated dialogs based on Gricean maxims of quantity, quality, relation and manner. We conduct a survey to evaluate 11 dialogs and show that the evaluations received in each of the four categories can be used to make informative comparisons between particular characteristics of the dialogs. The analysis was used to find areas for improvement. The evaluations can also be summed up to form a meaningful score that can be compared against a threshold.

Further scope for work in the dialog system exists in extending the hierarchically structured knowledge base to a network with more complex interconnections among the entities. A corpus annotated with dialog acts can be used to resolve intention recognition issues.

The keyword identification system may be trained to produce labels with more granularity (e.g. main keyword, additional constraint, etc.). It can also be implemented in a language independent manner since dependency relations can be universal, and may also contribute towards cross-lingual solutions.

To improve the evaluation metrics, collecting a larger and more diverse dialog set and scores can facilitate rigorous statistical tests; the agreement between the human judges must also be computed.

Related Publications

1. Prathyusha Jwalapuram and Radhika Mamidi, Handling Multi-Sentence Queries in a Domain Independent Dialogue System. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, pages 122-130, Kolkata, India, December 2017. NLP Association of India.
2. Prathyusha Jwalapuram and Radhika Mamidi, Domain Independent Keyword Identification for Question Answering. In *Proceedings of the 21st International Conference on Asian Language Processing (IALP-2017)*, Singapore, December 2017.
3. Prathyusha Jwalapuram, Evaluating Dialogs Based on Grice's Maxims. In *Proceedings of the Student Research Workshop Associated with RANLP 2017*, pg 17-24, Varna, September 2017, INCOMA Ltd.
4. Prathyusha Jwalapuram and Radhika Mamidi, Keyword and Constraint Identification for Question Answering. *Proceedings of 15th International Conference of the Pacific Association for Computational Linguistics, PACLING 2017*, Yangon, Myanmar, August 16-18, 2017.

Bibliography

- [1] J. F. Allen, D. K. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. Toward conversational human-computer interaction. *AI magazine*, 22(4):27, 2001.
- [2] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. Martin, B. Miller, M. Poesio, et al. The trains project: A case study in building a conversational planning agent. *Journal of Experimental & Theoretical Artificial Intelligence*, 7(1):7–48, 1995.
- [3] A. Bharati, A. Gupta, and R. Sangal. Parsing with nesting constraints. In *Proc of 3rd NLP Pacific Rim Symposium*, pages 4–6, 1995.
- [4] T. W. Bickmore, D. Schulman, and C. L. Sidner. A reusable framework for health counseling dialogue systems based on a behavioral medicine ontology. *Journal of biomedical informatics*, 44(2):183–197, 2011.
- [5] D. Bohus and A. I. Rudnicky. Ravenclaw: Dialog management using hierarchical task decomposition and an expectation agenda. *Eighth European Conference on Speech Communication and Technology*, 2003.
- [6] N. Dahlbäck, A. Jönsson, and L. Ahrenberg. Wizard of oz studies why and how. *Knowledge-based systems*, 6(4):258–266, 1993.
- [7] M. Danielli and E. Gerbino. Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the 1995 AAAI spring symposium on Empirical Methods in Discourse Interpretation and Generation*, volume 16, pages 34–39, 1995.
- [8] M. O. Dzikovska, J. F. Allen, and M. D. Swift. Integrating linguistic and domain knowledge for spoken dialogue systems in multiple domains. In *Proc. of IJCAI-03 Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 2003.
- [9] K. Eichler, H. Hensen, and G. Neumann. Unsupervised and domain-independent extraction of technical terms from scientific articles in digital libraries. In *LWA*, pages WIR–13, 2009.
- [10] O. Etzioni, M. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Methods for domain-independent information extraction from the web: An experimental comparison. In *AAAI*, pages 391–398, 2004.
- [11] G. Ferguson, J. F. Allen, et al. Trips: An integrated intelligent problem-solving assistant. In *Aaai/Iaai*, pages 567–572, 1998.

- [12] A. Flycht-Eriksson. Design and use of ontologies in information-providing dialogue systems. *Dissertation, Linkping Studies in Science and Technology, Thesis n. 874, School of Engineering, Linkping University, 2004.*
- [13] N. M. Fraser and G. N. Gilbert. Simulating speech systems. *Computer Speech & Language*, 5(1):81–99, 1991.
- [14] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web*, pages 71–80. ACM, 2007.
- [15] H. P. Grice, P. Cole, J. Morgan, et al. Logic and conversation. 1975, pages 41–58, 1975.
- [16] A. Gupta, A. Akula, D. Malladi, P. Kukkadapu, V. Ainavolu, and R. Sangal. A novel approach towards building a portable nlibd system using the computational paninian grammar framework. In *International Conference on Asian Language Processing (IALP), 2012*, pages 93–96. IEEE, 2012.
- [17] S. Han and Y. Kim. Intelligent dialogue system for plane euclidean geometry learning. In *International Conference on Computers in Education, Seoul, Korea, 2001.*
- [18] S. Harabagiu, D. Moldovan, and T. Yukawa. Testing gricean constraints on a wordnet-based coherence evaluation system. In *Working Notes of the AAAI-96 Spring Symposium on Computational Approaches to Interpreting and Generating Conversational Implicature*, pages 31–38, 1996.
- [19] H. Hastie. Metrics and evaluation of spoken dialogue systems. In *Data-Driven Methods for Adaptive Spoken Dialogue Systems*, pages 131–150. Springer, 2012.
- [20] R. Higashinaka, K. Funakoshi, Y. Kobayashi, and M. Inaba. The dialogue breakdown detection challenge: Task description, datasets, and evaluation metrics. In *LREC, 2016.*
- [21] R. Higashinaka, K. Funakoshi, M. Mizukami, H. Tsukahara, Y. Kobayashi, and M. Araki. Analyzing dialogue breakdowns in chat-oriented dialogue systems. *Errare*, 2015.
- [22] K. S. Hone and R. Graham. Towards a tool for the subjective assessment of speech system interfaces (sassi). *Natural Language Engineering*, 6(3-4):287–303, 2000.
- [23] V. Hung, M. Elvir, A. Gonzalez, and R. DeMara. Towards a method for evaluating naturalness in conversational dialog systems. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 1236–1241. IEEE, 2009.
- [24] T. Johnson. Natural language computing: the commercial applications. *The Knowledge Engineering Review*, 1(3):11–23, 1984.
- [25] P. Jwalapuram and R. Mamidi. Keyword and constraint identification for question answering. In *15th International Conference of the Pacific Association for Computational Linguistics, Yangon, Myanmar, in print, 2017.*
- [26] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

- [27] L. Lamel, S. Rosset, and J.-L. Gauvain. Considerations in the design and evaluation of spoken language dialog systems. In *Sixth International Conference on Spoken Language Processing*, 2000.
- [28] C. Lee, S. Jung, S. Kim, and G. G. Lee. Example-based dialog modeling for practical multi-domain dialog system. *Speech Communication*, 51(5):466–484, 2009.
- [29] J. Li, W. Monroe, T. Shi, A. Ritter, and D. Jurafsky. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*, 2017.
- [30] X. Li, Y.-N. Chen, L. Li, and J. Gao. End-to-end task-completion neural dialogue systems. *arXiv preprint arXiv:1703.01008*, 2017.
- [31] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023*, 2016.
- [32] L. Mazuel and N. Sabouret. Generic natural language command interpretation in ontology-based dialogue systems. In *Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology*, pages 347–350. IEEE Computer Society, 2006.
- [33] D. Milward and M. Beveridge. Ontology-based dialogue systems. In *Proc. 3rd Workshop on Knowledge and reasoning in practical dialogue systems (IJCAI03)*, pages 9–18, 2003.
- [34] M. Minock. A phrasal approach to natural language interfaces over databases. In *International Conference on Application of Natural Language to Information Systems*, pages 333–336. Springer, 2005.
- [35] S. Möller. *Quality of telephone-based spoken dialogue systems*. Springer Science & Business Media, 2004.
- [36] J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajic, C. D. Manning, R. T. McDonald, S. Petrov, S. Pyysalo, N. Silveira, et al. Universal dependencies v1: A multilingual treebank collection. In *LREC*, 2016.
- [37] T. Paek. Empirical methods for evaluating dialog systems. In *Proceedings of the workshop on Evaluation for Language and Dialogue Systems-Volume 9*, page 2. Association for Computational Linguistics, 2001.
- [38] A.-M. Popescu, O. Etzioni, and H. Kautz. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 149–157. ACM, 2003.
- [39] N. M. Radziwill and M. C. Benton. Evaluating quality of chatbots and intelligent conversational agents. *arXiv preprint arXiv:1704.04579*, 2017.
- [40] R. Rafal, G. Yali, and A. Kenji. Naturalness of an utterance based on the automatically retrieved common-sense. In *Proceedings of IJCAI 2005-Nineteenth International Joint Conference on Artificial Intelligence*. IJCAI, 2005.
- [41] R. P. Schumaker, M. Ginsburg, H. Chen, and Y. Liu. An evaluation of the chat and knowledge delivery components of a low-level dialog system: The az-alice experiment. *Decision Support Systems*, 42(4):2236–2246, 2007.

- [42] R. P. Schumaker, Y. Liu, M. Ginsburg, and H. Chen. Evaluating mass knowledge acquisition using the alic chatterbot: The az-alice dialog system. *International Journal of Human-Computer Studies*, 64(11):1132–1140, 2006.
- [43] G. Semeraro, H. Andersen, V. Andersen, P. Lops, and F. Abbattista. Evaluation and validation of a conversational agent embodied in a bookstore. *Universal Access Theoretical Perspectives, Practice, and Experience*, pages 360–371, 2003.
- [44] B. A. Shawar and E. Atwell. Different measurements metrics to evaluate a chatbot system. In *Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, pages 89–96. Association for Computational Linguistics, 2007.
- [45] S. Srirampur, R. Chandibhamar, A. Palakurthi, and R. Mamidi. Concepts identification of an nl query in nllidb systems. In *International Conference on Asian Language Processing (IALP), 2014*, pages 230–233. IEEE, 2014.
- [46] A. M. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.
- [47] O. Vinyals and Q. Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.
- [48] E. M. Voorhees et al. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82, 1999.
- [49] M. A. Walker, D. J. Litman, C. A. Kamm, and A. Abella. Paradise: A framework for evaluating spoken dialogue agents. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 271–280. Association for Computational Linguistics, 1997.
- [50] R. S. Wallace. The anatomy of alic. *Parsing the Turing Test*, pages 181–210, 2009.
- [51] D. L. Waltz and B. A. Goodman. Writing a natural language data base system. In *IJCAI*, volume 77, pages 144–150, 1977.
- [52] J. Weizenbaum. Eliza computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.
- [53] T.-H. Wen, D. Vandyke, N. Mrksic, M. Gasic, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*, 2016.
- [54] X. Yang, Y.-N. Chen, D. Hakkani-Tur, P. Crook, X. Li, J. Gao, and L. Deng. End-to-end joint learning of natural language understanding and dialogue manager. *42nd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2017) arXiv:1612.00913*, 2016.
- [55] R. M. Young. Using grice’s maxim of quantity to select the content of plan descriptions. *Artificial Intelligence*, 115(2):215–256, 1999.
- [56] Z. Yu, L. Nicolich-Henkin, A. W. Black, and A. I. Rudnicky. A wizard-of-oz study on a non-task-oriented dialog systems that reacts to user engagement. In *SIGDIAL Conference*, pages 55–63, 2016.
- [57] C. Zhang. Automatic keyword extraction from documents using conditional random fields. *Journal of Computational Information Systems*, 4(3):1169–1180, 2008.