The Use of Belief Networks in Natural Language Understanding and Dialog Modeling

衛智文
WAI, Chi Man Carmen

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Systems Engineering and Engineering Management

Abstract

This thesis describes a methodology of using machine learning techniques for natural language understanding and dialog modeling in a human-computer conversational system. We aim to reduce the amount of handcrafting in the development of spoken language systems and ease portability of the systems across different application domains. We explore the use of Belief Networks (BNs) to capture domain-specific knowledge within a restricted domain, by modeling the causal relationships between semantic concepts in the user's query and its underlying informational goal. The BNs are used to infer the users' underlying intentions in their informational-seeking queries. Concept identification and goal inference constitute natural language understanding (NLU). Our BN-based framework for NLU includes a methodology whereby the BN topology can be automatically learned to capture more sophisticated inter-concept dependencies to improve the understanding accuracy. We have also shown that our framework is robust against speech recognition errors in spoken queries. We use backward inference to ensure that the user's query is compatible with the domain-specific constraints. Backward inference can automatically detect and identify spurious or missing concepts in the query. Our BN-based framework can reject the spurious concepts or invoke the dialog model to clarify for the spurious concepts. Alternatively, the dialog model

can be invoked to prompt for missing concepts. As a result, our framework can be extended to drive mixed-initiative dialog modeling. We have demonstrated the scalability of our BN-based dialog model from the simple domain of foreign exchange to a more complex domain of air travel information service. To ease portability across domain, which implies the lack of training data, we have proposed a set of principles for probability assignment to the BNs, to reflect the "degree of belief" in the relationships between the concepts and the goals. Applications of our BN-based framework in natural language understanding and dialog modeling gave promising results and the performances fell within a reasonable range when we compared with the benchmark.

摘要

本論文主要是研究以機器學習的技術，運用於自然語言的理解及人機對話系統的模型塑造。爲了在建立話音語言系統的過程中減省人手及增加該系統對不同領域 (domain) 的可移植性，我們採用了信念網絡 (Belief Networks) 去擷取在特定領域的對話當中，一些不同語意概念的配合與所帶出的意向之間的因果關係。模型化後的因果關係可用作推斷人們在查詢資訊時的概念及目的，這有助於自然語言理解的運用。我們所設計的信念網絡更可從訓練數據中自動學習，去擷取各種語意概念之間的依存關係來改良自然語言理解的準確性。在存在語音識別錯誤的情況下，使用者的查詢會被信念網絡利用追溯推理 (backward inference) 所確認，一些被懷疑因識別錯誤而產生的概念會被偵測出來。這信念網絡還可伸延至處理語音對話的查詢。當追溯推理從使用者的查詢中偵測出一些可能是欠缺或是錯誤的概念的時候，系統便會提示使用者作出確認或矯正。這個偵測流程使到我們的混合對答 (mixed-initiative) 對話模型能夠自動操作。我們証實這個以信念網絡爲本的對話模型從較簡單的外幣兌換 (Foreign Exchange) 領域到較複雜的航空資訊 (Air Travel Information Service) 領域的可變比例性。對於領域可移植性方面，一個新的領域通常會缺乏訓練數據，因此我們提出了一系列的法則爲信念網絡裡的或然率賦值，來反映出不同概念與其目的之間的關係的可信度 (“degree of belief”)。基於我們的信念網絡架構

的應用系統，在自然語言理解及對話模型方面，有著一定的效果。而且在基準測試當中，更有合理的表現。

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Professor Helen Meng for her guidance, support and patience in guiding me through the completion of my thesis. I am grateful for being her Project Engineer as well as her student in the past three years. It has been a great pleasure and honor to learn from her and work with her closely. She has cultivated my interest in spoken dialog system and her continual support was the driving force behind my graduate study. She has also offered me different opportunities to explore myself — being intern at the SpeechWorks International Ltd. in New York was an invaluable experience and having presentation in front of hundreds of people in a conference was also challenging. In addition, I wish to thank her for endless proofreading and valuable suggestions for continually putting this thesis with perfection.

I wish to thank the members of my thesis committee, Dr. Stephanie Seneff from MIT; Professor Wai Lam and Professor Boon-Toh Low at the Department of Systems Engineering and Engineering Management from CUHK, for their careful reading of my thesis and their helpful feedback and suggestions.

My heartfelt thanks to Dr. Roberto Pieraccini for being an amazing supervisor while I was an intern at the SpeechWorks. Thanks for his invaluable suggestions and giving me a deeper understanding in the spoken dialog systems. Thanks to all the colleagues I had met at SpeechWorks for sharing their experience and also taking care of me, especially Cynthia Wellington, Asli Binal, Bobbi Manian, Bob Carpenter, Eric Woudenberg, Ezra Story, Jim Chadwick, Krishna Dayanidhi, Mark Murphy, Mary McAna, Roy Feldhusen, Ryan Osborn and Sasha Caskey.

My gratitude goes out to Tony Siu for providing of technical assistance, for sharing his idea and for his endless support for this thesis. Special thanks to Kon-Fan Low, for his useful feedback and suggestions on his earlier related projects in Belief Networks. My thanks go to Chi-Kin Keung for his enlightening conversations and for giving different enjoyable activities. Thanks to Ada Luk for her friendships and for sharing my happiness and sadness with her. Thanks to the rest of the wonderful members in the HCCL for creating a stimulating and entertaining environment. Ida Hui, Julia Wong and Cheong-Chat Chan for running the laboratory so smoothly, making sure the machines are well utilized and keeping the snack boxes overflown. Tiffany Fung, Yuk-Chi Li, Connie Tsui and Sally Yau for sharing the knowledge of their research interest and for the mutually encouragement during the hard time. Brenda Chan, Silvia Lo and Sunny Wong for sharing joy with me. I also thank Nelson Wong for being a cool officemate and for giving the weird but always interesting games and readings.

I reserve my most heartfelt to my family for their enduring love and concern over the years. Thanks to my parents for the encouragement throughout the good and bad times. Thanks to my sister for her unending support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

In this information age with proliferous usage of computers and connectivity to the Internet, many people are interested in the ability to access real-time information at any time and from anywhere. Many different applications have emerged that automate the services that have previously been performed by human operators. Nowadays, people can easily interact with the computers or Web browsers for bill payment, retrieve real-time stock quotes from menu-driven touch tone or voice telephone and view e-mail through WAP phones, etc. Effective human-computer communication plays an important role to improve overall user's satisfaction and eliminate his / her frustration in using the menu-driven services. Speech is one of the most *natural* and *flexible* human-computer communication as it comes very naturally to most people. The use of spoken language offers ease-of-use which reduces caller training and allows for both new and experienced users to use the same interface. As a result, there has been an increasing number of commercial applications that enable users to converse with computers.

One of the key enabling technologies for a human-computer conversation is spoken language understanding. State-of-the-art conversational systems can respond to the user's information-seeking queries for a restricted domain. These queries can often be classified into several domain-specific categories. However, for a given query type, the informational goal for a query and the possible ways of expression are legion. Natural language understanding (NLU) requires more than spotting keywords or key phrases in the spoken request. It involves identifying the informational goal(s) from the query's semantics, and subsequently retrieving the relevant information to produce a coherent response.

As an example, we can consider an enormously simplified weather domain, which only consists of three semantic concepts: <WEATHER>, <LOCATION> and <DATE>. A query which specifies two of the three concepts is likely to be asking for the missing one, e.g. "*What do we expect for Hong Kong tomorrow?*". A query containing all three concepts is likely to be asking for a yes / no response, e.g. "*Will there be sunshine in Hong Kong tomorrow?*". Another example is call-routing in AT&T's "*How May I Help You?*" task [1] and other similar call center tasks [5]. Here the caller's informational goal determines the destination for call-routing.

Most of the approaches to spoken language system development require a grammar for parsing the user queries into corresponding semantic concepts. The grammar is usually handcrafted by a domain or linguistic expert. In order for the computer to decode the underlying intentions of the queries for understanding, heuristic rules for the specific domain are needed to map the captured key concepts into informational goal(s). Besides, background knowledge on domain-specific constraints is necessary for handling discourse context as well as resolving ambiguities. Due to the extensive handcrafting in the heuristic design of spoken language systems, portability to a new application domain is an expensive and time-consuming task.

In order to reduce the amount of handcrafting in the development of spoken language systems, this thesis explores the use of machine learning techniques to capture the domain-specific knowledge (semantic constraints) automatically. We adopt a statistical approach that uses Belief Networks (also known as Bayesian Networks, BNs) to capture the causal relationships between the semantic concepts in the user's query and its underlying informational goal. BNs are then able to infer the users' informational goal(s) for natural language understanding. We will also focus on improving the spoken language understanding by recovering from speech recognition errors. Besides, our BN-based framework is also extended for dialog modeling in a spoken language system.

1.2 Natural Language Understanding

Our work is motivated by the idea that the understanding component should utilize the domain-specific knowledge to infer the speaker's informational goal(s) in the spoken query. Also, the query is usually accompanied with incomplete information and requires reasoning the underlying speaker's intention under uncertainty. As a result, we devised a methodology that explores the use of Belief Networks to capture the statistical dependencies between the key semantic concepts and the domain-specific goal(s). The BNs are then used to identify the user's informational goal(s) out of a finite set of within-domain goals. We believe that the Belief Networks offer several advantages to our problem [16]:

1. The dependencies between a query's informational goal(s) and the relevant semantic concepts may be effectively captured in the topology of the BN. The BN topology can also be automatically learned to capture more sophisticated dependencies.

2. BNs identify the informational goal by means of probabilistic inferencing. Under situations where massive data is involved, this provides an attractive alternative to handcrafting the heuristics between parses and their interpretations.

3. BNs can handle situations where the input observations are incomplete or contain uncertainty, and thus may model spoken queries well.

4. The BN framework is suited for the optional incorporation of prior knowledge in order to aid the inference process.

1.3 BNs for Handling Speech Recognition Errors

The accuracy of spoken language systems depends critically on the speech recognition component. Speech recognition output may have errors which lead to mismatches between the system's and the user's expectation. The errors can be associated with word deletions, insertions or substitutions. However, sometimes the errors cannot be recovered, and the user is forced into a spiral of confusions during the interaction. In order to improve the robustness against the speech recognition errors to prevent misunderstanding between the system and the user, recognition errors within a spoken query should be detected and rejected automatically. Most of the current approaches incorporate the likelihood of the acoustic models in which the N-best recognition hypotheses are computed. The utterance / word whose acoustic score is below a certain level will be identified as potential errors and the system will then prompt for the user.

We have explored a method for automatic post-processing the speech recognition output which attempts to recover the recognition errors during high level NLU processing. We will not make use of any acoustic scores. Only the semantic constraints captured by the Belief Networks are used to detect the mis-recognized concepts. While the BNs are used to infer the informational goals of the user's query for natural language understanding, the input query is verified against the domain-specific constraints by backward inference. The BNs can thus detect the possibly spurious concepts caused by mis-recognition.

1.4 BNs for Dialog Modeling

So far, our emphasis on natural language understanding is on self-contained queries whose meaning is complete. However, when the user converses with a real spoken dialog system, the interpretation of the queries may also depend on the discourse context. For example, consider the queries below:

> "*I want to go from The Chinese University to Shatin.*"
> "*How far is Shatin from here?*"

The former query is self-contained as its context is independent of the other query, while the interpretation of the latter query depends on earlier discourse (it is referred as a discourse query). The context of the discourse will affect the meaning of each individual utterance of the dialog and the information of a discourse query carries beyond what is simply expressed in the individual sentence. Besides, when the users interact with the system to access or retrieve the desire information, it may be impractical for users to specify all their requirements within a single query. The query is usually incomplete, and sometimes inconsistent with respect to the discourse history. As a result, the spoken dialog system should be capable of understanding the discourse queries as well as providing proper guidance to the users.

Dialog management plays an important role in spoken dialog systems in assisting users to achieve their goal efficiently. It requires keeping an updated history and analyzing the intention of the user incrementally, as the context may change while the dialog proceeds. A suitable strategy should also be determined to drive the dialog model. Currently, there are several approaches for dialog modeling. The system-initiative dialog model assumes complete control in guiding the user through an interaction towards task completion. This model often attains high task completion rates, but the user is bound by many constraints throughout the interaction. Conversely, the user-initiative model offers maximum flexibility to the user in determining the preferred course of interaction. However this model often has lower task completion rates relative to the system-initiative model, especially when the user's request falls beyond the system's competence level. To strike a balance between these two models, the mixed-initiative dialog model allows both the user and the system to influence the course of interaction. It is possible to build effective mixed-initiative interactions by handcrafting flexible transitions between the system-initiative and user-initiative models. While handcrafting can produce a sophisticated dialog flow, this process is expensive, and may become intractable with increasingly complex domains.

In this thesis, our BN-based framework is extensible to handling discourse queries. Each individual discourse query that contains partial information is captured by the BN for goal inferencing. Based on the existing context, the BN of the corresponding inferred goal will perform backward inference to detect the missing / spurious concepts automatically. This detection procedure automatically governs the model transition for mixed-initiative interactions.

The problem of out-of-vocabulary (OOV) words is also crucial in spoken dialog systems. Our BN-based dialog model is also enhanced to identify OOV words. Unseen words which are confirmed by the users will be incorporated into the lexicon automatically, and the system can guide the users with proper instructions.

Scalability of our BN-based dialog model from a simple domain to a more complex task domain will be illustrated. To ease the portability across domains, which often implies the lack of training data for the new domain, we have incorporated prior knowledge for the development of BNs. A set of principles for hand-assigning BN probabilities are developed, based on the "degree of belief" in the relationships between concepts and goals.

1.5 Thesis Goals

In this thesis, we will illustrate the feasibility of using Belief Networks (BNs) in natural language understanding and dialog modeling. The topologies of the Belief Networks are learnable, and are designed to model the relationships between the key concepts and goals. The BNs are used to infer the informational goals of self-contained queries for natural language understanding. Input queries can also be verified against the domain-specific constraints by backward inference, which can detect missing or spurious concepts. The spurious concepts which are often caused by mis-recognition can be rejected to improve the goal identification accuracy. Our BN-based framework is also extensible to handling the discourse query within a conversation. The detected missing or spurious concepts can drive the mixed-initiative interactions. We aim to demonstrate that our BN-based dialog model is scalable and portable to other new domains so as to reduce the amount of handcrafting in the development of spoken language systems.

1.6 Thesis Outline

This thesis is organized as follows: Chapter 2 describes some previous work in tackling the problems in natural language understanding, speech recognition error handling and dialog modeling of a spoken dialog system. As we propose to use Belief Networks, background and applications of the Belief Networks are also introduced. Chapter 3 details the use of Belief Networks in natural language understanding to infer the informational goals and handle mis-recognized words for the spoken queries. Chapter 4 presents the extensibility of our Belief Network framework for mixed-initiative dialog modeling. Chapter 5 demonstrates the scalability and portability of our BN-based dialog model across different application domains. Conclusions and future work are provided in Chapter 6.

Chapter 2

Background

This thesis sets out to reduce the amount of handcrafting in the development of spoken language systems. Language understanding involves inferring the underlying intentions of the self-contained queries and also the queries whose meaning is dependent on discourse context in a conversation. Another challenging issue in spoken language systems is handling speech recognition output. The presence of mis-recognized words may result in the miscommunication between the user and the system. Therefore, our natural language understanding framework should be robust against recognition errors in spoken queries, as well as extensible to dialog modeling to handle discourse queries. Natural language understanding, speech recognition errors handling and dialog modeling are well known problems in the area of speech and language technology research. In this chapter, we will describe the background information in these areas. In Section 2.1, we will review the previous work on understanding natural language queries. In Section 2.2, general approaches for handling recognition errors will be presented. In Section 2.3, several approaches adopted in the current spoken language systems are described. Since, we are proposing to use Belief Networks for natural language understanding and dialog modeling, we will provide the background information of Belief Networks in Section 2.4.

2.1 Natural Language Understanding

Understanding with natural language (both in text or spoken queries) depends heavily on the knowledge of the application domain. It requires inference about the speaker's informational goal(s) and semantic concepts. Besides, the most challenging task for understanding spoken language is that the spontaneous speech is often agrammatical, containing false starts, disfluencies, filled pauses ("umm", "ah"), etc. Also, it is inevitable to encounter new words uttered by speakers that are out of the system's vocabulary [11]. Hence, an understanding component designed for text input which assumes the entire word string is grammatically correct should be enhanced for spoken input.

Research in natural language understanding has been going on for decades, but most efforts are focused on restricted domains. The ATIS (Air Travel Information Service) task initiated by the DARPA spoken language systems program in late 1980's has spurred different research sites to take part in the development of understanding systems. The ATIS project focused on the database access task which allowed users to inquire for flight information such as flight schedules, air fares, ground transportation at a specific airport, etc. Since different systems were developed using the same relational database originating from the Official Airline Guide (OAG), it was possible to compare system performances based on the correct extracted information using a set of prescribed training and test data [47]. Systems were evaluated regularly according to three benchmarks: SPREC (speech recognition performance), NL (natural language understanding of written transcriptions of the spoken sentences) and SLS (spoken language understanding).

The simplest approach for natural language understanding is to perform keyword spotting on the speech recognizer's output. This technique works quite well for small applications. However, it is difficult to scale up to large tasks. When the application domain is more complex, systems require sophisticated natural language analysis to achieve understanding. Previous approaches to this problem include the use of heuristics to map a parsed query into an interpretation. The "parse" may be the output of a grammar-based parser [51], a stochastic concept decoder [45], probabilistic recursive transition networks [35] or semantic interpretation rules [24]. There are also approaches that involve the use of a key-phrase spotting technique [5] for sentence interpretation.

2.1.1 Rule-based Approaches

Most of the rule-based models involve a grammar for meaning interpretation [4, 21, 51, 61]. The grammar usually hand-designed by domain and linguistic experts can handle syntactic and semantic patterns by attempting to cover all the legitimate ways people could ask within a domain. For instance, the grammar is written as a set of context-free rules with constraints in the TINA parser [51], it is converted to a network structure where each node represents a syntactic or semantic category. The utterance can then be parsed into a tree structure where non-terminals are mapped into a semantic frame for interpretation. Here the interpretation adopted depends primarily on an evaluation among the parse alternatives, and sometimes the heuristics may not identify the best interpretation if task knowledge were to be considered. As a result, Ward et al. [61] have proposed a "beam of interpretations" approach, where multiple interpretations from multiple parses are used. In order to handle disfluent spontaneous speech that is often agrammatical, a robust matcher which combines using heuristics is also adopted to analyze parsable phrases and clauses [21, 50].

Since the grammar usually includes the domain-specific constraints, it is difficult to write a set of generalized rules for all domain-independent applications. Due to the extensive handcrafting on the grammar design, portability of a natural language component to another application domain is an expensive task. However, when the rule-based model is compared with the statistical model (more details below) which requires a large annotated training corpus, there might be an advantage in using hand-designed rules for a new application whenever annotated training data is unavailable.

2.1.2 Stochastic Approaches

Stochastic approaches have been explored in order to incorporate automatic learning methods into natural language understanding. Examples include the AT&T CHRONUS [45], LIMSI-CNRS [37], BBN-HUM [35] and the CHANEL systems [24] for the ARPA-ATIS task. In a stochastic-based model, the relationships between the semantic labels (concepts) and corresponding word sequence are automatically learned from a large annotated training corpus. These relationships are used to decode the word sequence into a set of semantic labels for understanding. There exist different techniques for modeling the stochastic-based understanding system. They are illustrated as follows:

2.1.2.1 Hidden Markov Models (HMM)

Hidden Markov Models (HMMs) were used in the AT&T CHRONUS [45] and the LIMSI-CNRS systems [37] in which the words are modeled as observations and the concepts are the hidden states. The model attempts to find the sequence of words W and the concept sequence C that maximize $P(W, C|A)$, where A is the acoustic evidence. This is equivalent to finding the most likely sequence of states in the conceptual HMM, given the word sequence. The conceptual decoder then carries out a linear sequential mapping between sentence segments and concepts. This approach views understanding as decoding of the concepts "hidden" in an utterance, and portability has been demonstrated by migration from the ATIS (Air Travel Information Service) to the French MASK (Multimodal-Multimedia Automated Service Kiosk) task [36].

2.1.2.2 Hidden Understanding Models (HUM)

The hidden understanding model (BBN-HUM) system [35] is based on an approach similar to the Hidden Markov Models. It aims to determine meaning directly from a sequence of words. Probabilistic transition networks are adopted in which meaning is represented by a tree structure while subconcepts are nested within other concepts. Let M be the meaning of an utterance, and W be the sequence of words that convey the meaning. By Bayes' rule, the problem is formulated as:

$$P(M|W) = P(W|M)\frac{P(M)}{P(W)} \tag{2.1}$$

Since $P(W)$ is fixed, maximizing $P(M|W)$ is equivalent to maximize the product $P(W|M)P(M)$. $P(W|M)$ is a semantic language model that employs tree-structure meaning representation: concepts are represented as nodes in the tree, with sub-concepts represented as children. On the other hand, $P(M)$ is a lexical realization model which specifies the transition probabilities between words (i.e. bigram language model). As a result, understanding a sequence of words requires finding a meaning M such that the probability $P(W|M)P(M)$ is maximized. It is done by finding the maximum probability path through the network made up of the two combined models (semantic language network and lexical realization network).

2.1.2.3 Semantic Classification Trees (SCTs)

Semantic interpretation rules have also been used for natural language understanding, where the rules are learnt automatically by decision trees known as Semantic Classification Trees (SCTs) [24]. Each main attribute found in the database (e.g. main attribute $fare.fare\_id$ in the air-travel database) is represented by a SCT that is a binary decision tree with a question associated with each node. Questions are about a sentence pattern consisting of a word sequence or an expression containing words with gaps. Therefore, the model is robust to spontaneous speech. Each node of a SCT has two successors: YES and NO for denoting the presence or absence of the corresponding concept, each of which leads to a different successor node. As a result, understanding the word sequence is simply formulated as a classification problem performed by the SCTs.

The stochastic models can be easily adapted to a new domain if an annotated training corpus is available. However, in the domain of natural language understanding, data collection and annotation is a difficult and time-consuming task. As a result, the corpora with limited data cannot be applied to stochastic models easily.

In this work, we also adopt a statistical approach, except that we use Belief Networks to capture the domain-specific knowledge for natural language understanding. Annotated training data is required for training probabilities. In addition, we have developed a set of guidelines to reflect the "degree of belief" for the Belief Networks so as to ameliorate reliance on the annotated corpora.

2.1.3 Phrase-Spotting Approaches

In this approach, concept-based phrases are directly related to semantic interpretation for understanding. The phrase-level constraint provides wider coverage and it increases robustness to the ill-formed spontaneous utterance. The phrase-spotting model can be achieved by a progressive search strategy such as the A* search in [22]. There has also been an approach that involved the automatic extraction of phrases in a restricted domain. A set of grammar fragments [1], including both syntactically and semantically organized phrases was obtained by clustering of the phrases based on their similarity. In the AT&T *"How May I Help you?"* call-routing service [14], these grammar fragments are associated with various call types. The observed fragments in an utterance are used to determine the most likely call type by maximizing the aposteriori probability for each call type or passing the fragments through a classifier. Alternatively, the call-type classification is performed by a vector-based information retrieval technique [5] where each utterance is represented by a vector of concepts $\vec{D}$. The problem is formulated as a topic identification or document classification problem, and for every input query the system outputs a single identified topic. The corresponding call type is determined by finding the closest degree of similarity between the input utterance and various documents (call types).

2.2 Handling Recognition Errors in Spoken Queries

One of the difficult tasks for spoken language understanding is to handle the imperfect output from the speech recognizer. It is important for the systems to provide graceful recovery from errors. Most of the systems make use of the confidence scores for the N-best hypothesized outputs. The systems are then able to identify and reject the utterances that are likely to be mis-recognized. Information on the semantic interpretation from natural language components can also provide an indication of possible mis-recognition if there is a parse failure on the sentence hypothesis [41].

Spoken language systems which depend on utterance-level confidence scores for detecting mis-recognized words cannot provide optimal rejections, as the overall confidence scores cannot reflect the correctness of a specific word. Therefore, most of the recent speech recognizers have the capability to output word-level confidence scores for each word of the hypotheses in the N-best list [15]. The word with confidence score lower than a certain threshold is likely to be mis-recognized and the systems can take different appropriate actions such as confirmation, rejection, etc. Some systems even integrate the individual word confidence score to generate slot-based (semantic) confidence scores [6]. By combining the prior knowledge of the task with slot-based confidence scores, the slot with low confidence scores which violates the domain-specific constraints will be rejected automatically.

Other than using the acoustic confidence measure to detect the potential errors in the recognizer's hypotheses, some approaches make use of semantic and discourse information [26, 41, 59]. The annotated training hypotheses (label correct or incorrect with respect to the reference transcription) together with the corresponding features (e.g. acoustic confidence scores, semantic information, parse status, etc) are used to train a classifier. Hence the classifier (usually a decision tree) is able to generate an overall confidence likelihood score that can be used to make accept / reject decision for each hypothesized sentence.

In this work, we also attempt to detect the mis-recognized words from spoken queries to improve the understanding accuracy. As acoustic information is not readily available in our data set, we will make use of domain-specific constraints to detect potential errors in the spoken input queries.

2.3 Spoken Dialog Systems

Spoken dialog systems can interact with users to retrieve information so as to complete a goal-oriented task for a restricted domain. The speech recognizer transforms speech to text, the natural language component analyzes the meaning of the user input, the dialog manager determines the appropriate action, the response / retrieved information delivered to the users is expressed in natural language by language generation and the text-to-speech synthesizer. Figure 2.1 depicts the basic components for a spoken dialog system [63].

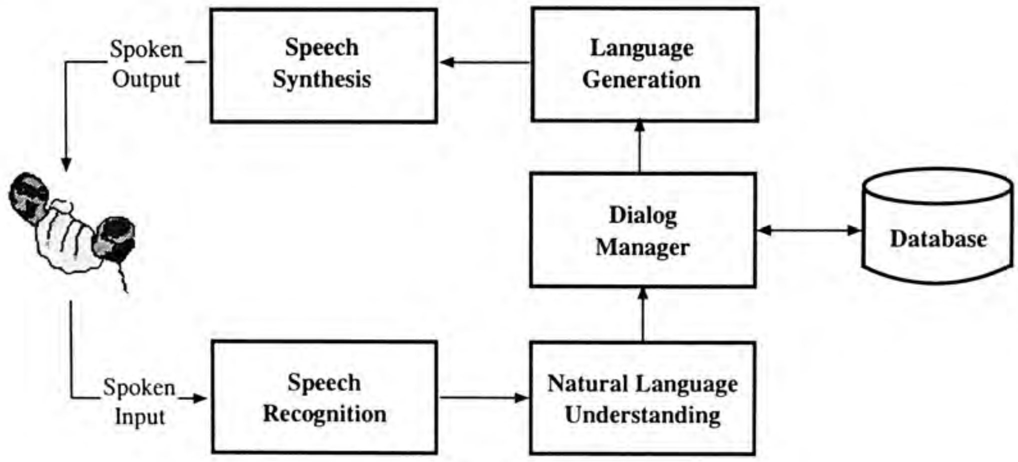

Figure 2.1: Basic components for a spoken dialog system.

There are many spoken dialog systems designed to support mixed-initiative dialog interaction in a multitude of application domains. Examples include air travel (e.g. ATIS, MERCURY, PEGASUS) [47, 52, 65], train schedules (e.g. RAILTEL, MASK, ARISE) [3, 25, 55], electronic automobile classifieds (e.g. WHEELS) [31], weather (e.g. JUPITER) [64], restaurant guide (e.g. BeRP) [21], ferry timetable (e.g. WAXHOLM) [4] and call-routing service [5]. The languages concerned include English and a number of European languages. A few systems have also been developed for Mandarin Chinese [19, 60] and for Cantonese Chinese [32, 34].

Dialog management is the key component in spoken-language systems; it guides the users to complete their goals. The dialog strategies of these systems usually range from system-initiative to user-initiative. The system-initiative dialog model provides directed questions and guides the users through an interaction. This model is relatively straightforward to design because it is usually modeled as a tree structure. Each node represents a system prompt and it branches out with a limited number of user responses. As the system-initiative system usually acquires one attribute in each dialog turn, it can usually attain a higher task completion rate. However, the system-initiative dialog tends to be inflexible, and the user is bound by many constraints throughout the interaction. Conversely, the natural language-based user-initiative dialog model allows for unconstrained input from users. However, this model usually has lower task completion rates relative to the system-initiative model, often because the user's request falls beyond the system's competence level. Among the various dialog strategies, the *mixed-initiative* model is deemed most desirable, since both the user and the system can influence the dialog flow over the course of interaction. This provides greater flexibility than the system-initiative model or user-initiative model.

In the following sections, we will describe several general approaches to the dialog modeling, including finite-state networks (FSN) [23, 62], form-based approaches [12, 43] and sequential decision approaches [27, 44]. Recent research has explored the use of machine learning techniques [13, 28] to automatically determine an appropriate dialog strategy.

2.3.1 Finite-State Networks

One of the simplest approaches to handle human-computer interaction is based on finite-state networks [23, 62]. In a finite-state network, each state represents a particular dialog state, which illustrates the amount of information acquired in the dialog thus far. The transition arc connected with each state represents the corresponding action to be carried out. The transition arc is usually accompanied by a probability value to indicate which action is the most likely for each state, and the probabilities can be either hand-designed by a domain expert or learnt from an annotation corpus. The dialog manager can then determine the appropriate action according to the user's input query.

The finite-state systems are relatively easy to design, as the conversation flow is conducted by traversing one of the pre-defined paths in the network. This dialog model is suitable for domains with a limited number of domain-specific concepts where the interaction is well defined and can be structured as a tree, with one question at each branch of the tree. However, in a more complex domain with a large number of domain-specific concepts, the dialog space may be quite large, and it then becomes difficult and inefficient to associate the entire space of dialog states with corresponding actions and transition arcs.

2.3.2 The Form-based Approaches

Many current dialog systems adopt a form-based approach which handles the dialog status by filling an electronic form (or E-form) [2, 7, 9, 12, 43, 46]. The E-form consists a set of attribute-value slots, representing the constraints acquired for database access. The user's utterance, once it has been converted into the semantic frame via the natural language processing, will subsequently be mapped to the canonical format of the E-form. For each attribute, it is indicated whether the value should be mandatory, forbiddenly or optionally supplied so that the corresponding query type is fully specified and consistent. The system will reason the current status of the E-form based on the hand-designed rules or control table [2] and determine the appropriate response. Context-inheritance is carried out by integrating current dialog context in the E-form. Heuristic rules are often needed to resolve the ambiguity when additional constraints are filled in the slots. When the system acquires all the necessary attribute values for a certain query type, the E-form is ready for database retrieval.

The E-form paradigm provides more flexibility than the finite-state approach, and it can pursue a mixed-initiative dialog model. While the system elicits attribute-value pairs for the users, they do not need to comply with the system's prompt. They can specify their constraints and requirements in any order and fill more than one slot in a single turn. However, the E-form approach is limited to a single-goal approach. It is difficult to handle a complex task with multiple goals.

2.3.3 Sequential Decision Approaches

Some of the mixed-initiative dialog systems have extended the finite-state paradigm to a sequential decision model strategy [27, 44, 52]. The model is described with a set of dialog states, actions and strategy. Dialog states are state variables which represent the amount of information available at a certain point in the dialog. Actions correspond to the interactions of the system with the environment (e.g. users, database) at different dialog states. Strategy is the mapping of the dialog states with the corresponding actions. Figure 2.2 illustrates the algorithm of the AT&T dialog system, which adopts a sequential decision model [29].

$S_t = S_1$
while $S_t \neq S_F$ {
    $A_t \quad = NextAction(S_t)$
    Invoke A_t
    $O_t \quad =$ environment response to A_t
    $S_{i+1} = NextState(S_t, A_t, O_t)$
    $t \quad = t + 1$
}

where
    S_t, A_t denote the state and action at turn t.
    S_1 and S_f stand for initial state and final state.
    Function $NextAction$ determines the next action A_t.
    Function $NextState$ updates the state variable.

Figure 2.2: A dialog described as a sequential process.

The strategy for determining the action for the current dialog state can be implemented in different ways. The AT&T AMICA framework [27] adopts an approach in which the strategy is represented as a recursive transition network. While the arcs of the network illustrate the conditions on the dialog state (e.g. inconsistent, missing mandatory information), the nodes represent different actions (e.g. verification, getting information). As a result, the dialog manager is required to analyse the current dialog state and then refer to the control state in the recursive transition network for determining the appropriate action. For the Mercury Flight Reservation System [52] and the other GALAXY domains [64, 65], the dialog strategy is controlled by a dialog control table. A set of ordered rules are designed which represent the conditions of the dialog state with the corresponding actions. Hence the system consults the dialog control table at each dialog turn and fires the action when a specified condition is met.

Spoken dialog systems using a sequential decision process model usually require a mechanism (dialog strategy) for mapping an appropriate action to each dialog state. The dialog strategy is usually hand-designed by domain experts. It is difficult to explore all the possible situations the dialog system might encounter and portability of the dialog model to a new domain requires the whole design process to be started from scratch. In the next section, we will describe the recent research effort that explores the use of machine learning techniques to automatically determine the optimal dialog strategy.

2.3.4 Machine Learning Approaches

In order to reduce the handcraft design of a spoken dialog system, there are previous efforts that explore the use of machine learning techniques to automatically determine the optimal dialog strategy [13, 28, 30, 58]. A dialog system is described as a sequential decision process (Section 2.3.3) that consists of state space, action set and strategy. While the dialog is formulated as an optimization problem, an optimal strategy can be obtained by reinforcement learning.

Reinforcement learning [56], which is commonly adopted in artificial intelligence problems, involves the agent (i.e. decision maker) learning from interaction with an external environment to achieve a goal. At each time step, the agent receives the environment's state s_t, and then the agent selects and executes an action a_t. The agent's action causes a change to a new state s_{t+1} and the agent receives a reward, $r+1$ from the environment. Figure 2.3 illustrates the agent-environment interaction in the reinforcement learning framework.

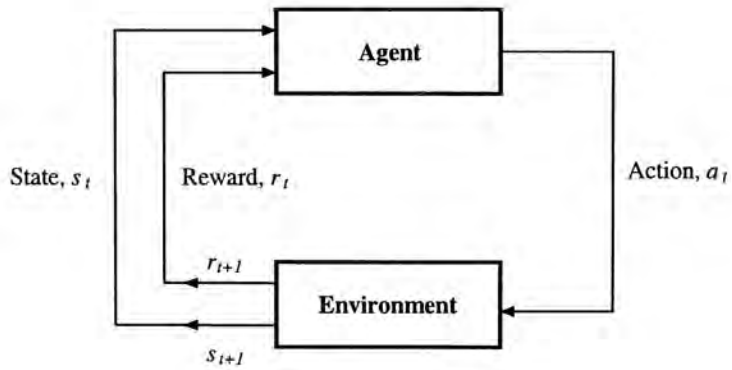

Figure 2.3: The framework of reinforcement learning.

For each interaction, the agent will map the corresponding state to probabilities of selecting each possible action. This mapping is called the agent's strategy / policy and denoted as π_t. Reinforcement learning specifies how the agent changes its strategy so as to maximize the *total amount* of reward it receives over the *long term.*

To apply reinforcement learning in finding an optimal strategy for a dialog system, a reward function is designed. For example in [28], the reward measure is defined as a cost function: a linear combination of the costs such as the duration of the dialog, number of errors, etc. Since the reward function only indicates the immediate reward at a specific state-action pair, a state-action value function $Q(s_t, a_t)$ is required to specify the *total amount* of reward accumulated over the *future* starting from that state. The optimal state-action value $Q^*(s_t, a_t)$ is the expected sum of costs incurred from state s_t with action a_t and following the optimal strategy π^* until the final state is reached. This value can be estimated using reinforcement learning in which the accumulated reward will be successively improved at each iteration. The best dialog strategy is then obtained by selecting the action with the optimal value function at each dialog state.

The methodology of using reinforcement learning in dialog design can automate the development process of a dialog system [28]. Based on a large training corpus, it can explore many possible dialog strategies and keep adapting to user behavior and learning from interaction. The only limitation is that it may be difficult to choose a reward measure or represent the dialog in terms of state space and a set of actions.

Our BN-based dialog model is similar to the machine learing approach. The BNs are trained from the training corpus and the mapping of the user query to the corresponding system response is also performed by the backward inference of the BNs automatically. Since we have developed a set of guidelines for hand-assigning the probabilities for the BNs, our framework can still be applied to a domain where there is a lack of training data.

2.4 Belief Networks

2.4.1 Introduction

Reasoning with uncertain knowledge and beliefs has been an important research issue in the field of artificial intelligence for many years. Several methodologies have been proposed, including certainty factors, fuzzy logic, Dempster-Shafer theory, and probability theory [38]. The probabilistic approach is by far the most popular among all these alternatives. Belief Networks [42] (also known as Bayesian Networks, Causal Networks) are commonly adopted to model the causal structure of a non-deterministic process and apply probability theory for reasoning about incomplete knowledge.

The Belief Network is a directed graph consisting of nodes and directed arcs. While the node represents a random variable with a set of mutually exclusive states, the directed arc models the causal influences between the linked variables. For each variable v with parents / ascendants $w_1, \ldots, w_n$, a conditional probability table $P(v|w_1, \ldots, w_n)$ is defined. Obviously, if v has no parents, this table reduces to the prior probability $P(v)$.

The advantage of a network representation is that it allows the causal dependencies among the variables to be interpreted easily. Consider a typical example from [42], an apartment is installed with an alarm system and the alarm can be set off by two things, an earthquake or a burglary. The likelihood of the earthquake causing the alarm to go off is small compared to a burglary in progress. A radio announcement would be strong evidence supporting an earthquake. Another way to find out about the state of the alarm is through a call from the neighbor. Since a burglary is probably independent of an earthquake, the occurrence of a radio announcement about an earthquake reduces the likelihood of a burglary. We can represent the causal relationships between the events in a directed graph as shown in Figure 2.4. Each node in this network represents a Boolean variable. The value becomes true if a certain event takes place. The diagram expresses certain probabilistic relationships between the events. If there is an observation of a certain event, the probability of each event in the network will be updated.

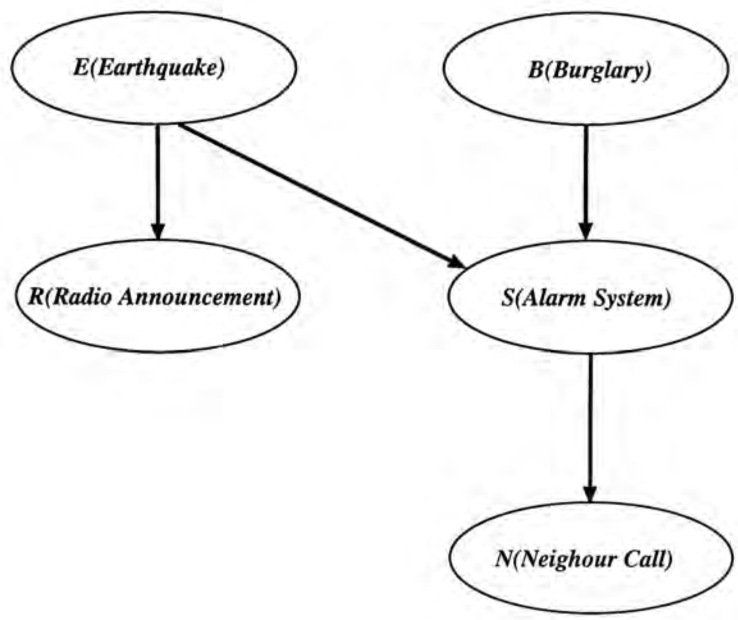

Figure 2.4: A Belief Network depicting the Earthquake-Burglary example.

Conditional independencies among variables in the Belief Network can reduce the amount of computation in finding out the joint probability. The chain rule of probability theory illustrates that the joint probability for the network in Figure 2.4 can be derived from conditional probabilities:

$$P(N, S, R, E, B) = P(N|S, R, E, B)P(S|R, E, B)P(R|E, B)P(E|B)P(B) \tag{2.2}$$

We can simplify the above equation due to conditional independence assumptions for the variables. For example, N (neighbor call) is dependent only on S (alarm system), so $P(N|S, R, E, B) = P(N|S)$. Finally, we have:

$$P(N, S, R, E, B) = P(N|S)P(S|E, B)P(R|E)P(E)P(B) \tag{2.3}$$

In general, given nodes $X = x_1, \ldots, x_n$ for a Belief Network, the joint probability function for any Belief networks can be represented as:

$$P(X) = \prod_{i=1}^{n} P(x_i|\, parents(x_i)) \tag{2.4}$$

2.4.2 Bayesian Inference

Suppose we observe a certain value for one or more of the variables in the Belief Network, these variables will be instantiated and the beliefs (probabilities) of the variables in the network will be revised. Hence inference involves updating the probabilities of each node in a Belief Network from the instantiated nodes. There are different algorithms for Bayesian inferencing. Details can be found in [38, 42]. A brief description is provided here for a simple structure of a Belief Network. For a network with very complex structure, one of the efficient methods for updating probabilities is the cluster-tree method [20] that transforms the complex network into a set of clusters. The updating of the probabilities in Belief networks is quite complicated; details can be found in Appendix C.

Consider the Belief Network in Figure 2.5. The pre-defined structure of the BN leads to computation equivalent to Naïve Bayesian inference. It consists of a hypothesis node H and three evidence nodes E_1, E_2 and E_3. Table 2.1 shows the probability distribution of each variable.

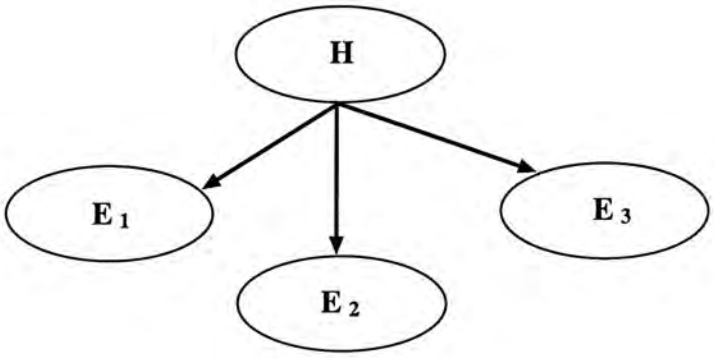

Figure 2.5: A Belief Network with pre-defined structure.

	$P(H)$
$H = 1$	0.45
$H = 0$	0.55

	$P(E_1 \mid H)$	
	$H = 1$	$H = 0$
$E_1 = 1$	0.8	0.6
$E_1 = 0$	0.2	0.4

	$P(E_2 \mid H)$	
	$H = 1$	$H = 0$
$E_2 = 1$	0.5	0.7
$E_2 = 0$	0.5	0.3

	$P(E_3 \mid H)$	
	$H = 1$	$H = 0$
$E_3 = 1$	0.1	0.4
$E_3 = 0$	0.9	0.6

Table 2.1: Probability distribution of each node of the Belief Network in Figure 2.5.

Now if we have observed the evidence nodes with the values of $E_1 = 1$, $E_2 = 0$ and $E_3 = 1$, we want to infer whether the hypothesis node H is true ($H = 1$) for the given evidence nodes. The observed values from the evidence nodes can give an updated probability for the value of H by Bayesian inferencing. In the other words, $P(h|e_1, e_2, e_3)$ is computed (the lowercase symbols stand for the particular values of the corresponding variable). By Bayes' rule, we have:

$$P(h|e_1, e_2, e_3) = \frac{P(h, e_1, e_2, e_3)}{P(e_1, e_2, e_3)} \tag{2.5}$$

Since the value of h is mutually exclusive and exhaustive, the denominator of Equation 2.5 can be transformed to:

$$P(h|e_1, e_2, e_3) = \frac{P(h, e_1, e_2, e_3)}{\sum_{h'} P(h', e_1, e_2, e_3)} \tag{2.6}$$

By Equation 2.4 as described in Section 2.4.1, the joint probability for the numerator and denominator of Equation 2.6 can be factored as follows:

$$P(h|e_1, e_2, e_3) = \frac{P(e_1|h)P(e_2|h)P(e_3|h)P(h)}{\sum_{h'} P(e_1|h')P(e_2|h')P(e_3|h')P(h')} \tag{2.7}$$

Then, Equation 2.7 can be computed by inserting the values for the variables:

$$\begin{aligned}
& P(h = 1|e_1 = 1, e_2 = 0, e_3 = 1) \\
= \; & \frac{P(e_1 = 1|h = 1)P(e_2 = 0|h = 1)P(e_3 = 1|h)P(h = 1)}{\sum_{h'=0,1} P(e_1 = 1|h')P(e_2 = 0|h')P(e_3 = 1|h')P(h')} \\
= \; & \frac{0.018}{0.018 + 0.0396} \\
= \; & 0.3125
\end{aligned}$$

While the prior probability of the hypothesis H is 0.45, the inferred probability of H, given the other variables, is 0.3125.

2.4.3 Applications of the Belief Networks

Applications of the Belief Networks mainly involve dynamic reasoning under uncertainty. Either handcrafted or automatically learned BNs are embedded in the system for inferencing. The major application area has been problem diagnosis [42], which computes the likelihood of alternate diseases in patients or disorders in machines. For example, the Belief Network framework in [54] automates the troubleshooting process by modeling the relationships between various types of printing system problems and their causes.

Belief Networks have also been explored as a representation for common sense reasoning. For instance, they are adopted to make inference about the goals of car drivers in navigating in traffic in which the aim is to induce the plan of driving action from the partial observation of the traffic or highway status [48].

The Lumière [17] project from the Microsoft research group also makes use of Belief Networks in creating software that can automatically and intelligently interact with software users. When the users want to get assistance from online help, they always attempt to communicate their goal with common words and phrases instead of jargon. Hence the Belief Networks, which model the relationships between words in a user's query and the corresponding informational goal, can reason about the goal from the observed actions and queries [16]. This project eventually resulted in the "Office Assistant" in the Microsoft application, which allows users to request assistance by composing natural language queries. Another project from the Microsoft research group called the Bayesian Receptionist focuses on modeling the users' goals typically handled by a receptionist at the front desk [18]. The Belief Networks developed in different levels of task hierarchy can decompose the user's goal into several sub-goals for reasoning. Other than the linguistic features of the user query, the model also makes use of visual finding and gesture to infer the goals of the speakers.

2.5 Chapter Summary

In this chapter, we have described the background information of this thesis. Previous approaches on understanding natural language queries are reviewed. We have also presented the general approaches for handling the recognition errors in the spoken queries so as to improve the understanding accuracies. Besides, several approaches adopted in current spoken language systems are described. Finally, we have given a brief introduction on the use of Belief Networks for reasoning uncertainty. A number of current applications using Belief Networks are also examined.

Chapter 3

Belief Networks for Natural Language Understanding

In this chapter, we explore the use of machine learning techniques in natural language understanding. We adopt a statistical approach, which uses Belief Networks (BNs) to model the domain-specific constraints in a restricted domain. The BNs are intended to capture the causal relationships between the key semantic concepts and the domain-specific goal, and infer the underlying intention of a spoken utterance for natural language understanding. However, in a real spoken language system, the presence of mis-recognized words often causes the system to misunderstand the user query and results in an inappropriate response. As a result, we have devised a methodology which extends our BN framework to handle noisy concepts in the spoken queries. BNs are used to detect and reject mis-recognized concepts automatically and hence infer the user's informational goal(s) correctly.

3.1 The ATIS Domain

We have chosen the ATIS (Air Travel Information Service) domain [47] to investigate the feasibility of using the Belief Networks (BNs) for natural language understanding. ATIS is a common task which was adopted in the DARPA (Advanced Research Projects Agency) Speech and Language Program in the USA. The database is based on the Official Airline Guide (OAG) for airlines operating in North America and Canada.

Our experiments are conducted using the ATIS-3 Class A sentences, with disjoint training and test sets of 1564, 448 (1993 test), and 444 (1994 test) transcribed utterances respectively. A class A sentence is a context-independent query whose interpretation is independent of the dialog context.

ATIS-3 Class A	**1993 Training**	**1993 Test**	**1994 Test**
# Transcribed Utterances	1,564	448	444

Table 3.1: Breakdown of the ATIS-3 Class A sentences.

Each utterance (or query) is accompanied with a corresponding SQL query for retrieving the relevant information. An example Class A query is shown below. Notice that the SQL query is a simplified one. The two original SQL queries which are for maximum and minimum answer can be found in Appendix A.

Query : *"show me the united airlines flights from denver to baltimore leaving on june fourteenth"*

SQL : select *flight_id* from *flight* where *airline_name* = *"united airlines"* and *origin* = *"denver"* and *destination* = *"baltimore"* and *month* = *"june"* and *day* = *"fourteen"*

Thus we derived the informational goal for each utterance from the main attribute label of its SQL query. In our training set, we counted a total of 32 query types (or informational goals, e.g. flight identification, fare identification, etc). Inspection of the training utterances reveals that out of 32 goals, only 11 of them are instantiated 10 times or more. These 11 goals cover over 95% of the training set, 93% of the 1993 test set, and 92% of the 1994 test set. As a result, we treat the queries whose goals are not covered by these 11 goals as out-of-domain (OOD). Besides, we also found 24 training utterances with more than one informational goal. Examples of single goal, multiple goal and OOD queries are shown in Table 3.2.

	Single Goal Query
Query :	*"chicago to san francisco on continental"*
Goal :	Flight_ID
	Multiple Goal Query
Query :	*"give me the least expensive first class round trip ticket on u s air from cleveland to miami"*
Goal :	Flight_ID, Fare_ID
	OOD Query
Query :	*"show me the cities that midwest express serves"*
Goal :	City_Code (OOD, as City_Code is outside the set of 11 goals)

Table 3.2: Examples of single / multiple goal and OOD queries.

3.2 Problem Formulation

We need to infer an appropriate goal for a query, out of the finite set of goals (N) in a restricted domain. One may formulate the problem as N binary decisions, or a single N-ary decision. We have chosen the former approach to facilitate the identification of cases with multiple goals, as well as the rejection of cases with previously unseen, OOD goals. On the other hand, the latter single N-ary decision approach has no rejection capability and can only classify the query to be one of the N goals. We have implemented the N-ary decision approach using decision trees for benchmarking purpose.

We use a semantic tagger to transform the input query into a sequence of semantic concepts. These form the input to our BNs for inferring the query's informational goal(s). Our approach utilizes multiple BNs — each a distinct classifier for making the binary decision regarding a unique goal. A BN outputs the confidence level for its decision regarding an input query, in terms of the *aposteriori* probability. With the use of a probability threshold, the BN outputs for a particular goal may be quantized into a binary decision. Hence, queries for which all BNs vote negative are rejected as OOD.

3.3 Semantic Tagging

Semantic tagging abstracts the words in a query into a set of semantic concepts. While the main attribute label(s) in the SQL query is adopted as the informational goal(s), the remaining attribute labels are identified as key semantic concepts for the ATIS domain, and serve as a reference when we design our semantic tags for labeling an input transcription. We have included semantic concepts for database access, as well as others that play a syntactic role for natural language understanding. Therefore, we have a total of 60 hand-designed semantic tags, where both syntactic (e.g. <PREPOSITION>, <SUPERLATIVE>) and semantic concepts (e.g. <AIRLINE_NAME>, <DAY>,

<FLIGHT_NUMBER>) are included. Previous work has also devised a semi-automatic procedure for discovering such semantic categories from unannotated corpora [53]. Both the hand-designed and semi-automatically generated semantic tags (also known as grammar) are listed in Appendix B.

Training utterances are automatically tagged, using a heuristic procedure. This identifies the semantic concepts in the utterance transcriptions. The following shows an example of an utterance transcription and its corresponding tags:

Query : *"what are the dinner flights from indianapolis to san diego on wednesday may twelfth"*

Tags : <WHAT> <DUMMY> <MEAL_DESCRIPTION> <FLIGHT> <FROM> <CITY_ORIGIN> <TO> <CITY_DESTINATION> <PREP> <DAY_NAME> <MONTH> <DAY>

Since our semantic tagger can segment a given utterance into individual semantic concepts, it can handle spontaneous spoken queries which are filled with pauses or partial words. Henceforth each training query is represented by its annotated goal and a sequence of semantic concepts. These are used to train our BNs, as described in the following section.

3.4 Belief Networks Development

The Belief Network (BN) is a probabilistic causal network, and for our implementation, we adopt a pre-defined structure as depicted in Figure 3.1. This is equivalent to a Naïve Bayes formulation and it models the causal relationships between the concepts and the goal. Concepts within the query are assumed to be independent of one another. Directed arrows are drawn from cause to effect. Hence they show the statistical dependencies between the concepts and goal, which are represented by conditional probabilities of the concepts given the goal. To further simplify our model, each concept node receives a binary input for the concepts occurrence (either present or absent).

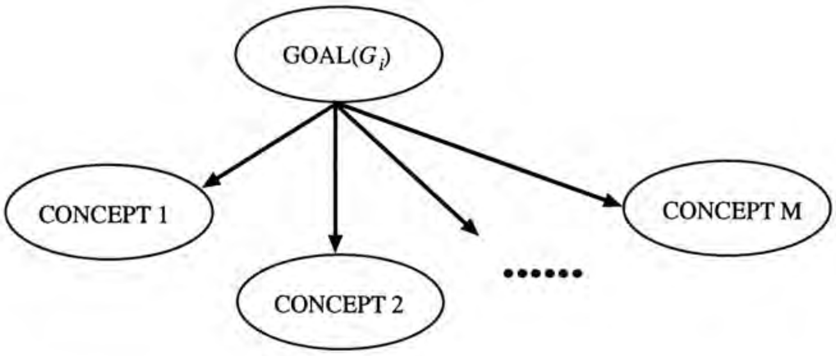

Figure 3.1: The pre-defined structure of a BN.

3.4.1 Concept Selection

For a given goal G_i, and its instantiations in the training set, we record the semantic concepts that are indicative of G_i. The recorded set is limited to M or below in size, in order to constrain computation during training. We compare the use of two measures to select the concepts (C) with strongest dependency on G_i:

(i.) Mutual Information [8], which measures the degree of co-occurrence of concept (C_j) and the corresponding goal (G_i) (*where* i=*1,2* ... N *and* j=*1,2* ... M).

$$MI(C_j, G_i) = P(C_j, G_i) \log \frac{P(C_j, G_i)}{P(C_j)P(G_i)} \tag{3.1}$$

(ii.) Information Gain [10], which is also known as expected mutual information, considers both the presence and absence of the concept and the corresponding goal.

$$IG(C_j, G_i) = \sum_{c=0,1} \sum_{g=0,1} P(C_j = c, G_i = g) \log \frac{P(C_j = c, G_i = g)}{P(C_j = c)P(G_i = g)} \quad (3.2)$$

Based on these measures, the top M semantic concepts will be selected as the feature set for the ith goal. Hence, each goal may have a different set of selected concepts.[1]

3.4.2 Bayesian Inferencing

We develop one BN for each informational goal. The probabilistic information on the goal node, i.e. $P(G_i)$ and the dependency of each concept on the corresponding goal, i.e. $P(C_j|G_i)$ are obtained by tallying the counts from the training data.

Given N trained BNs, we can identify the concepts in the user query by semantic tagging, and employ the BNs for inferring the user's informational goal(s). Each BN with pre-defined topology can apply Bayes' Theorem as shown in Equation 3.3 to infer the likelihood of the corresponding informational goal given the observed concepts $\vec{C}$.

$$P(G_i = 1|\vec{C}) = \frac{P(\vec{C}|G_i = 1)P(G_i = 1)}{P(\vec{C})} \quad (3.3)$$

3.4.3 Thresholding

Each BN outputs its confidence level for the case that the input query is conveying its corresponding goal. Choosing a probability threshold allows for

[1] Since we select a set of concepts for each goal, we will only assume the concepts are conditionally independent of each other.

quantization of this confidence level into a binary decision. It may be reasonable to set the probability threshold at 0.5, since $P(G|\vec{C}) + P(\overline{G}|\vec{C}) = 1$. Alternatively the threshold can be chosen such that the performance on goal inference is maximized. Related performance measures include *recall* (R), the percentage of queries correctly inferred by the BN for G_i out of all the G_i queries; and *precision* (P), the percentage of queries correctly inferred by the BN for G_i out of all the inferred G_i queries. We combine both into a single score by optimizing with the F-measure as shown in Equation 3.4 [57]: (β=1 in our experiments to treat precision and recall with equal importance).

$$F = \frac{(1+\beta^2)RP}{\beta R + P} \tag{3.4}$$

3.4.4 Goal Identification

We develop one BN per informational goal in the training corpus and we have a total of N BNs to represent a finite set of goals in the restricted domain. Each BN outputs the confidence level for its decision regarding an input query, in terms of the aposteriori probability. One decision scheme is to adopt the goal with maximum aposteriori probability for the input query, but it cannot classify the multiple goals and OOD queries. With the use of a probability threshold, the BN outputs for a particular goal may be quantized into a binary decision. The decisions across all the BNs are then combined to identify the output goal of an input query. Therefore, we can revert to an alternative scheme: the query is labeled with the goals for which the BNs vote positive, which achieves multiple goal identification; in the case where all BNs vote negative, the input query is rejected as OOD.

3.5 Experiments on Natural Language Understanding

Our experiments are based on the ATIS-3 Class A (context-independent) sentences. As mentioned previously, only 11 goals (out of 32) are instantiated 10 times or more. These 11 goals cover 95% of the training set. Consequently, we have 11 (N=11) BNs in total to avoid using sparsely trained networks. The remaining goals and utterances are treated as OOD.

3.5.1 Comparison between Mutual Information and Information Gain

For each of the 11 BNs, we select (M=20) concepts with the strongest dependency on the goal. *Mutual Information* (MI) and *Information Gain* (IG) are compared as the dependency measure. Only the selected goals in the training query are considered during classification, which maximizes the aposteriori probability in Equation 3.3.

Since IG considers both the presence and absence of concepts for goal classification, it can extract 20 concepts for all of our goals. MI considers only the cases when a concept is present, and extracts fewer than 20 concepts for a number of goals. Therefore when MI is used we normalize the aposteriori probability prior to goal classification, by padding with a multiplicative constant of 0.5. For example, MI selects 16 concepts for the BN of Ground_Transport. Therefore, four extra concepts with a probability distribution of $P(C_i|G) = 0.5$ and $P(C_i|\overline{G}) = 0.5$ will be padded to the network for normalization.

We adopt the goal with maximum aposteriori probability for the input query. Comparisons between the use of two different measures for concept selection (MI vs. IG) are tabulated in Table 3.3. The results are based on the goal classification accuracies on the training set.

Concept Selection Measure	**Performance (Training)**
Mutual Information (MI)	85.42% (1336/1564)
Information Gain (IG)	93.67% (1465/1564)

Table 3.3: Comparison between two different measures for concept selection (MI vs. IG)

It is observed that IG performs better than MI in concept selection for goal classification. This implies that the absence of certain concepts may be indicative of the informational goal under some situations. To illustrate with an example, consider a query from our training set:

Query : *"may I have a listing of flight numbers from columbus ohio to minneapolis minnesota on monday"*
Tags : <DUMMY> <HAVE> <DUMMY> <PREP> <FLIGHT_NUM> <FROM> <CITY_ORIGIN> <STATE_NAME> <TO> <CITY_DESTINATION> <STATE_NAME> <PREP> <DAY_NAME>
Goal : Flight_Number

According to the set of concepts selected by MI for the goal Flight_ID, all the query's semantic tags are indicative of the goal. The set of concepts selected by IG was similar, but it is also augmented by the *absence* of <FLIGHT_NUM>. The occurrence of <FLIGHT_NUM> in the input query lowers the aposteriori probability for the goal Flight_ID, which is eventually outweighed by the goal Flight_Number.

3.5.2 Varying the Input Dimensionality

A series of experiments were conducted in which we varied the BN input dimensionality, which is equivalent to the number of stored concepts per goal. Variation covered the range from 15 concepts to the full set of 60 concepts. The goal identification accuracies for the training set, 1993 test and 1994 test sets are tabulated in Figure 3.2.

Performance accuracies in the plot are normalized based on the full size of the training / test sets. Since our decision scheme is based on the maximum aposteriori probability among 11 BNs, queries which do not belong to the 11 goals are counted as errors. As observed in Figure 3.2, training accuracies increase with input dimensionality, while testing accuracies tend to decrease beyond 20 concepts per goal, possibly due to overfitting of the training data. This suggests that 20 concepts per goal is a suitable parameter setting. Performance with different normalization schemes — over the entire set, versus normalizing only over the relevant queries that belong to the 11 goals, are shown in Table 3.4.

Normalization	Training	1993 Test	1994 Test
Over All Queries	93.7% (1465/1564)	87.9% (394/448)	86.7% (385/444)
Queries of 11 Goals	97.1% (1465/1509)	95.4% (394/413)	94.6% (385/407)

Table 3.4: Goal classification accuracies computed using different normalization schemes.

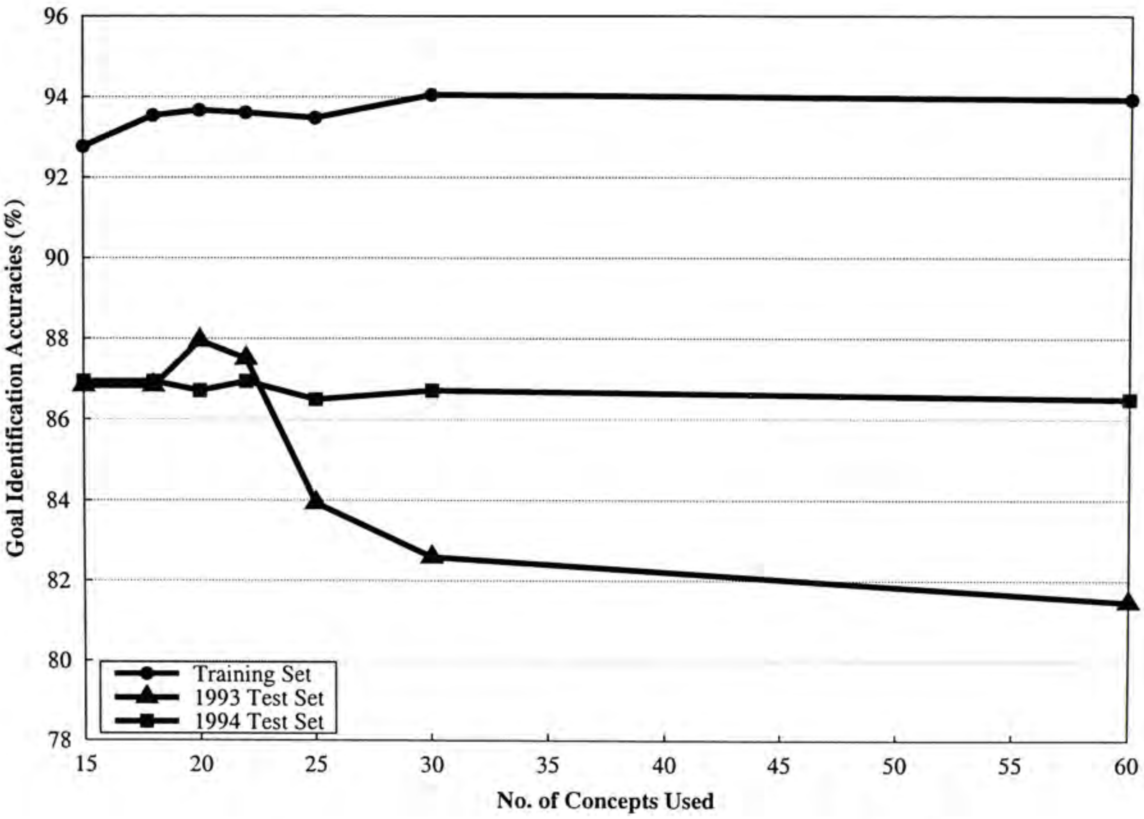

Figure 3.2: Goal identification accuracies for different BN input dimensionalitie schemes.

3.5.3 Multiple Goals and Rejection

Thresholding enables the BN to make a binary decision about its goal. For a given query, we can look across all BNs to see if more than one network has voted positive (the case of multiple goals), or if all networks have voted negative (the case of unseen goal). A conversational system, which identifies an information-seeking query to have multiple informational goals, may provide additional relevant information in the response. Alternatively if the query is identified with an unseen goal, it may be rejected as an OOD.

We can either set the probability threshold at 0.5 ($\theta_{0.5}$), or at a value which maximizes the F-measure (θ_f) as described in Section 3.4.3. The results obtained with the threshold of 0.5 and the values which maximize the F-measure are shown in Table 3.5. Their comparison is based on the performance of the two test sets — correct single goal classification, number of rejected queries, correct rejection, number of multiple goal queries, correct multiple goal classification and overall correctly handled queries. Table 3.5 suggests that the threshold should be set by the F-measure, rather than at 0.5. This increases the rejection rate, but also improves the rejection accuracy. Moreover, it drastically reduces the number of queries identified to have multiple goals, while maintaining the same correct identification rate. Correctly handled cases include queries with correct goal classification, as well as those with correct rejection. Comparison with our results without using a probability threshold (Table 3.5, bottom row) suggests that there is a slight performance advantage if an appropriate probability threshold is used.

	1993 Test		1994 Test	
Threshold	$\theta_{0.5}$	θ_f	$\theta_{0.5}$	θ_f
Single Goal Correct	382/405	381/405	372/401	373/401
# Rejection	19	39	30	35
Rejection Correct	8/35	23/35	12/37	13/37
# Multiple Goals	119	51	69	22
Multiple Goals Correct	5/8	5/8	4/6	4/6
Total Correctly Handled	88.2% (395/448)	91.3% (409/448)	87.4% (388/444)	87.8% (390/444)
Total Correctly Handled (No Threshold)	87.9% (394/448)		86.7% (385/444)	

Table 3.5: Comparing the use of different probability thresholds — the use of 0.5 vs. other values which maximize the F-measure. The values in the denominators illustrated the total number of the corresponding queries.

3.5.4 Comparing Grammars

In prior work at our laboratory, a statistical approach [53] was implemented to capture semantic categories from unannotated corpus so as to get rid of daunting grammar writing by domain experts. When we apply this semi-automatic procedure on the ATIS domain, it results in the discovery of 36 semantic categories (non-terminals), in both semantic and syntactic dimensions. Table 3.6 tabulates the size of our hand-designed grammar (G_H) as well as the semi-automatically generated grammar (G_{SA}). Semantic categories of both grammars are listed in Appendix B. The smaller size of the G_{SA} can be explained by the occurrence of sparse semantic concepts, which cannot be discovered during the automatic grammar induction process.

Grammar	Non-terminals	Terminals
Semi-automatically Generated (G_{SA})	36	446
Hand-designed (G_H)	60	483

Table 3.6: The size of semi-automatically generated (G_{SA}) and hand-designed grammars (G_H).

We have tested the goal identification performance using the G_{SA}. Comparative results on goal identification accuracies using the G_H and G_{SA} (with probability threshold of 0.5) are illustrated in Table 3.7. This suggests that the G_{SA} gives promising results and has only small degradation in goal identification accuracies.

ATIS-3	1993 Test		1994 Test	
Grammar Used	G_H	G_{SA}	G_H	G_{SA}
Single Goal Correct	382/405	364/405	372/401	370/401
Rejection Correct	8/35	8/35	12/37	11/37
Multiple Goals Correct	5/8	5/8	4/6	4/6
Total Correctly Handled	**88.2%**	**84.2%**	**87.4%**	**86.7%**

Table 3.7: Comparative results on goal identification accuracies based on a hand-designed (G_H) and a semi-automatically generated grammar (G_{SA}) respectively.

3.6 Benchmark with Decision Trees

As mentioned previously, to benchmark our approach we have implemented an alternative approach using decision trees [49]. We adopt *C4.5* decision tree, which learns production rules from the training set and generates the tree automatically. It also involves heuristic methods for simplifying the decision tree (pruned tree), with the aim of producing more comprehensible structure. A single decision tree is developed to make the 11-way decision on goal classification. The full set of 60 semantic concepts is used as input, and attribute selection for tree branching is based on the information-theoretic criterion. Assume that the training set T contains G_i classes (i.e. goals) and $|T|$ denotes the number of cases (i.e. queries) in T. We can use Equation 3.5 to measure the average amount of information needed to identify the classes in training set T. (This quantity is also known as the entropy of T.)

$$info(T) = -\sum_{i=1}^{N} \frac{freq(G_i)}{|T|} \log_2 \frac{freq(G_i)}{|T|} \tag{3.5}$$

If we partition the training set T on the attribute X (i.e. a semantic concept) into subsets: T_1, T_2 .., T_n, and $|T_i|$ denotes the number of cases in T_i, the expected information requirement can be found as the weighted sum over the subsets, as in Equation 3.6.

$$info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times info(T_i) \tag{3.6}$$

Information gain by branching on attribute X can be computed from Equation 3.7, and the attribute with the greatest information gained is selected as the root of the tree (sub-tree).

$$Gain(X) = info(T) - info_x(T) \tag{3.7}$$

Part of the decision tree representing the ATIS domain in classifying the 11 goals is shown in Figure 3.3. Each node has two successors for denoting the presence or absence of the corresponding concept. The decision tree hence classifies the user query by starting at the root of the tree and moving through until a leaf (goal) is encountered (without confidence levels in the output). Since the decision tree has no rejection capability, OOD queries are counted as errors. Performance was 90.0% (403/448) for the 1993 test set, and 88.7% (394/444) for the 1994 test set. On the other hand, our N-binary approach using BNs with probability threshold that maximizes the F-measure can facilitate the identification of cases of single, multiple goal and OOD queries. Comparative results are illustrated in Table 3.8, which suggests that both approaches deliver comparable performance for our task.

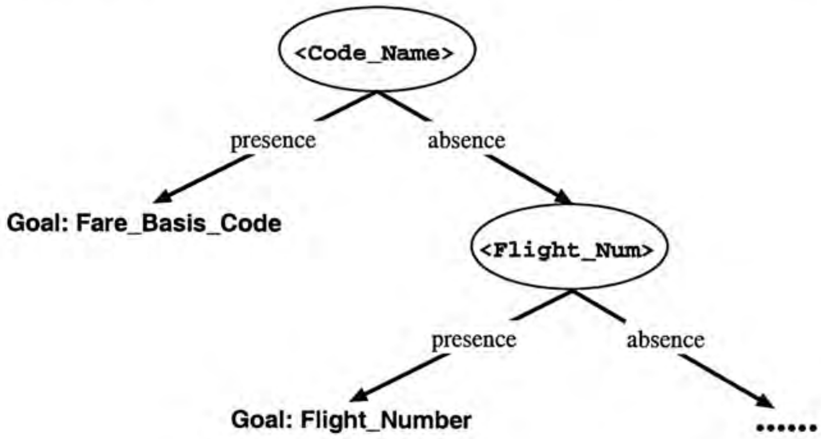

Figure 3.3: Part of the decision tree for the ATIS task.

Approach	1993 Test Set	1994 Test Set
BNs (N-binary)	91.3% (409/448)	87.8% (390/444)
Decision Tree (N-ary)	90.0% (403/448)	88.7% (394/444)

Table 3.8: Goal identification accuracies obtained using N-binary (BN) and N-ary approach (decision tree) respectively.

3.7 Performance on Natural Language Understanding

We have also evaluated our outputs in terms of their extracted semantic categories [37], where the evaluation metric is the sentence error rate. The evaluation procedures are illustrated on Figure 3.4. Each semantic case frame generated is compared against the attribute labels from the corresponding SQL query. A sentence is considered correct only if the inferred goal and extracted concepts in the generated semantic frame agree with those in the reference semantic frame (derived from the SQL in the ATIS corpora). The sentence error rates for the ATIS-3 1993 test set and 1994 test set are 9.8% and 13.7% as shown in Table 3.9. When we compare our results with the NL (natural language understanding of written transcriptions of the spoken sentences) results from the 10 ATIS evaluation sites [39] [40] as shown in Table 3.10, our performances fall within a reasonable range.

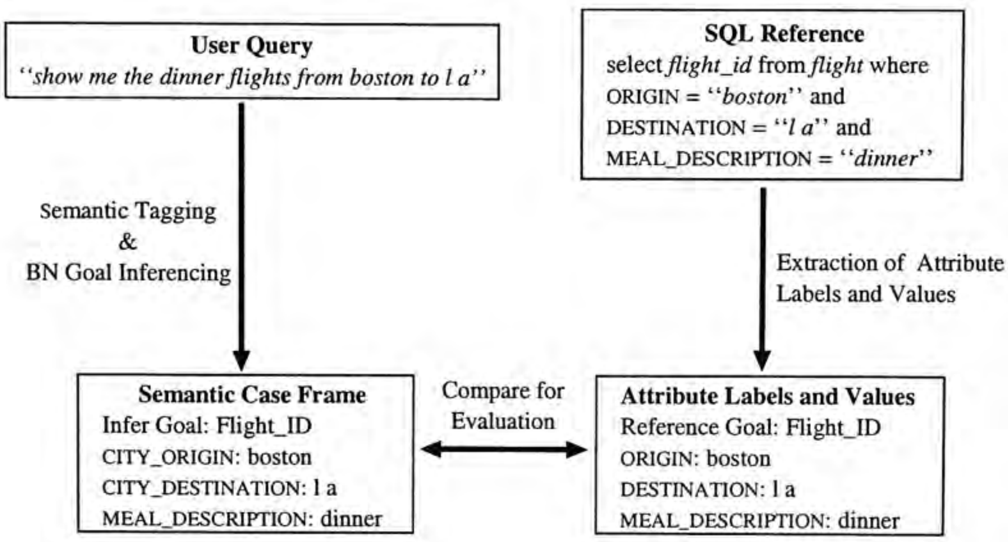

Figure 3.4: Procedures for semantic case frame evaluation.

ATIS-3 Class A	1993 Test Set	1994 Test Set
Sentence Error Rate	9.8%	13.7%

Table 3.9: The sentence error rates of Class A queries of ATIS-3 1993 and 1994 test sets respectively.

ATIS-3 Class A	1993 Test Set	1994 Test Set
Sentence Error Rate	6.0 – 28.6%	3.8 – 30.6%
Median Value	7.0%	12.4%

Table 3.10: Benchmark NL results from the 10 ATIS evaluation sites.

3.8 Handling Speech Recognition Errors in Spoken Queries

Speech recognition greatly affects the natural language understanding accuracies, as the presence of mis-recognized words always results in the inevitable mismatch with the user's expectation. Recognition errors may substitute a single word for another, replace a single word with multiple smaller words or erroneously concatenate multiple words to substitute for a longer word. All of these errors are stumbling blocks for understanding. To help alleviate this problem, we have extended the BN framework from natural language understanding to the detection of mis-recognized words. The learnable topology of our BNs is enhanced to capture the causal relationships between the concepts and the goal as well as among the concepts. The input query can then be verified against the domain-specific constraints by using backward inference, so that spurious concepts which are caused by mis-recognition can be detected and rejected.

3.8.1 Corpus Preparation

Our experiments are based on the training and 1993 test sets of the ATIS-3 Class A queries. In order to simulate the corrupted queries, i.e. with the presence of mis-recognized words, we will randomly convert a word unigram / bigram / trigram to their confusable counterparts based on the algorithm below:

Step 1 : Extract all the unigrams, bigrams and trigrams from the training set and find their corresponding phonetic pronunciation.

Step 2 : Generate a confusion matrix for all the extracted words or phrases.

Step 3 : Use a seed to randomly pick up a unigram / bigram / trigram within a user query and convert to its corresponding confusion counterparts with a phonetic distance of 1 or 2.[2]

Figure 3.5: Algorithm for simulating corrupted queries.

Some examples from ATIS domain includes:

Original Query : *cleveland to miami on wednesday arriving before four p m*

Corrupted Query : *cleveland to miami on* ***to*** *wednesday arriving before* ***fare*** *p m*

Original Query : *give me the flights and fares for a trip to cleveland from miami on wednesday*

Corrupted Query : *give me the flights* ***land for*** *a trip to cleveland from miami on wednesday*

[2] Phonetic distance is the count of the difference between the the two sequences of phonetic symbols.

3.8.2 Enhanced Belief Network Topology

The pre-defined BN topology incorporates the simplifying assumption that all concepts are dependent only on the goal, but are independent of one another. The topology can be enhanced by learning the inter-concept dependencies from training data according to the Minimum Description Length (MDL) principle [33]. The inclusion of inter-concept dependencies brought performance improvements in goal identification on ATIS-3 Class A queries by 3.5%. The resultant topology is illustrated in Figure 3.6. Notice that it captures not only the causal dependencies between the informational goals and the corresponding concepts, but it also shows the relations between the concepts (the dotted arrows). Hence, we have 11 BNs (with learnt topology) to represent each of the informational goals in the ATIS domain. In order to further improve the overall performance — single, multiple goals and OOD identification, we adopt a probability threshold which maximizes the F-measure (θ_f) as described in Section 3.4.3.

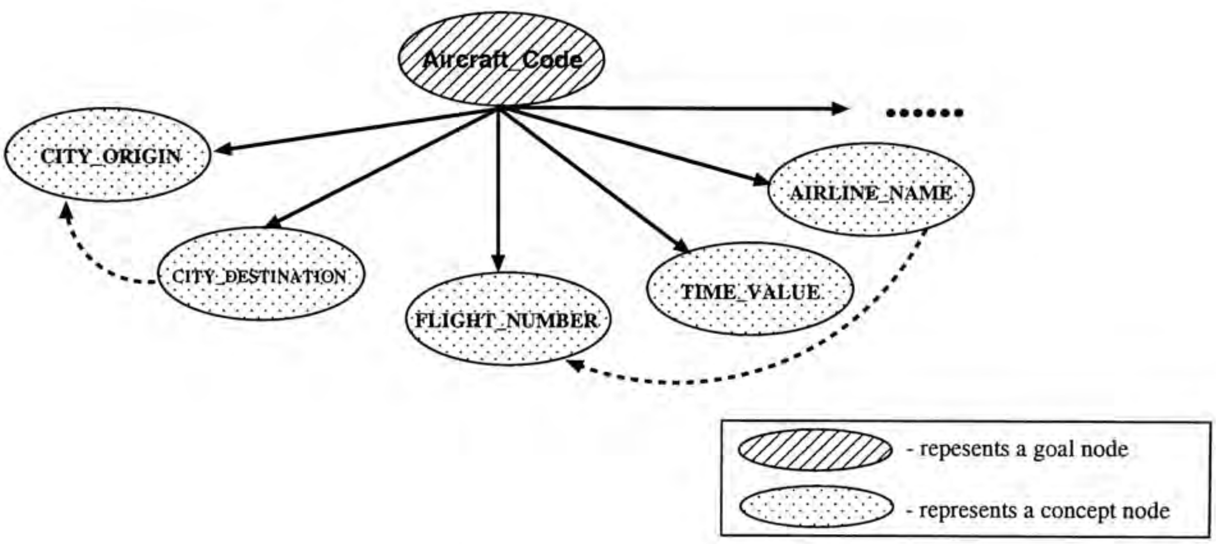

Figure 3.6: Topology of the BN for the informational goal Aircraft_Code.

3.8.3 BNs for Handling Speech Recognition Errors

The algorithm for detecting the mis-recognized concepts is summarized in Figure 3.7. The approach mainly involves: identification of informational goal, detection and rejection of spurious concepts.

For each noisy query

1 For each BN (i):

2 Informational goal inference:
Output a confidence level and make a binary decision by the threshold determined by the F-measure (θ_f).

3 Detection of spurious concepts by backward inference:
- i. Instantiate the goal node $P^*(G_i)$ to 1 or 0 to infer backwards for $P^*(C_j)$.[3]
- ii. If C_j is present in the query and $P^*(C_j)$ from backward inference is lower than the threshold θ,[4] C_j is deemed spurious.

4 Infer the query again with the spurious concepts removed.

5 Compare the inference results among 11 BNs. Trust the BN with maximum updated probability $P^*(G_i)$ and reject the spurious concepts according to its decision.

6 Use the updated concepts, and feed into the 11 BNs again to obtain the informational goal again. Each trained BN (11 BNs) will make a binary decision by θ_f regarding the presence or absence of its corresponding informational goal. Decision across all BNs results in the identification of single goal, multiple goals or OOD query.

Figure 3.7: Algorithm for handling noisy queries.

[3] The asterisk (*) denotes an updated probability.

[4] We can simply choose threshold at 0.5 since $P(C_j) + P(\overline{C_j}) = 1$. Experiments on varying the threshold value θ are discussed later.

3.8.3.1 Informational Goal Inferencing

Each BN will output a confidence level regarding the presence or absence of its informational goal. Probability propagation within the learnt BN topology is more complex and the Bayes' Theorem as described in Equation 3.3 cannot be applied for the calculation of the aposteriori probability $P^*(G_i|\vec{C})$ directly. According to the learnt topology as shown in Figure 3.6, the network is divided into sub-networks: {Aircraft_Code, CITY_ORIGIN, CITY_DESTINATION}, {Aircraft_Code, FLIGHT_NUMBER, AIRLINE_NAME}, {Aircraft_Code, TIME_VALUE}, etc. Updated joint probabilities $P^*(G_i, \vec{C})$ are iteratively computed according to the Equation 3.8 by each sub-network. The updated probability $P^*(G_i)$ is hence computed[5] by the marginalization of the updated joint probability $P^*(G_i, \vec{C})$. Notice that the probability associated with the asterisk (*) means its value has been updated by the instantiation of its / other nodes. $P^*(G_i)$ is then compared to the threshold θ_f (determined by the F-measure) to make the binary decision for the presence or absence of the goal.

$$P^*(G_i, \vec{C}) = P(G_i|\vec{C})P^*(\vec{C}) \rightarrow P^*(G_i, \vec{C}) = \frac{P(G_i, \vec{C})}{P(\vec{C})}P^*(\vec{C}) \qquad (3.8)$$

where $P^*(\vec{C})$ is instantiated to the presence or absence of the concepts.

$P(G_i, \vec{C})$ *is the joint probability obtained from training set.*

$P^*(G_i, \vec{C})$ *is the updated joint probability.*

[5] Detailed calculation of $P^*(G_i)$ is illustrated in Appendix C.

3.8.3.2 Detecting Mis-Recognized Concepts

Each BN will then perform backward inference to test the network's confidence for each input concept. Backward inference involves probability propagation within the BN. The goal node of the corresponding BN is instantiated to either 1 or 0 (this value is dependent on the binary decision of the BN). The updated goal probability $P^*(G_i)$ then propagates to update the joint probability of $P(C, G_i)$ for each sub-network by Equation 3.9:

$$P^*(\vec{C}, G_i) = P(\vec{C}|G_i)P^*(G_i) \tag{3.9}$$

where $P^*(G_i)$ *is updated and instantiated to 1 or 0.*

$P(\vec{C}|G_i)$ *is the conditional probability obtained from training data.*

$P^*(\vec{C}, G_i)$ *is the updated joint probability.*

Thereafter we can obtain $P^*(C_j)$ by marginalization.[6] By comparing this value with the threshold θ, we can determine whether the concept should be present or absent in the query. The value of θ can be pre-set as 0.5 or determined by maximizing the overall performance of the training set described in a later section. We can hence reject the spurious concepts that may be caused by mis-recognition if the concepts' occurrences do not match with the binary decision from the backward inference.

[6] Detailed calculation of $P^*(C_j)$ from backward inference is shown in Appendix C.

3.8.3.3 Rejecting Mis-Recognized Concepts

The detected spurious concepts are then rejected by the corresponding BN and each BN will perform goal inference again. The BN with the maximum updated probability $P^*(G)$ is accepted and we will update the concepts according to the decision from this BN. Then we will feed the updated query to the 11 BNs again for goal inference, the decisions across all BNs are combined to identify the informational goal of the input query. We labeled the query with a goal if the corresponding BN votes positive with the maximum updated probability $P^*(G)$. Alternatively, we may label the query with all goals for which the BNs vote positive. In the case where all BNs vote negative, the input query is rejected as OOD.

To provide an example, consider a Flight_ID query "*cleveland to miami on wednesday arriving before four p m*". We have simulated recognition errors within the query, and the corrupted query becomes: "*cleveland to miami on* ***to*** *wednesday arriving before* ***fare*** *p m*". Our previous BN framework infers this query goal as Fare_ID wrongly. Table 3.11 shows the processes for rejecting the mis-recognized words and inferring the informational goal correctly.

Query : *"cleveland to miami on* ***to*** *wednesday arriving before* ***fare*** *p m"*
Tags : <CITY_ORIGIN> <TO> <CITY_DESTINATION> <PREP> <PREP> <DAY_NAME> <TO> <PRE_TIME> <FARE> <TIME_UNIT>

Step 1: Informational goal inference by each BN.

Each BN outputs binary decision by using threshold θ_f:

$P(Goal = \text{Aircraft\_Code} \mid Query) = 0.0000 \rightarrow$ **Goal Absent (θ_f for this goal is 0.83)**
$P(Goal = \text{Fare\_ID} \mid Query) = 0.5035 \rightarrow$ **Goal Present (θ_f for this goal is 0.30)**
$P(Goal = \text{Flight\_ID} \mid Query) = 0.4902 \rightarrow$ **Goal Present (θ_f for this goal is 0.37)**
...

Step 2: Detection of spurious concepts by backward inference of each BN.

e.g. For the BN of Flight_ID: (below shows the probabilities obtained from backward inference using 0.5 as threshold θ. Notice that <FARE> is spurious because its presence violates the binary decision from backward inference.)

Concept$_j$ (C_j)	$P^*(C_j)$	Binary Decision	Actual Occurrence
CITY_ORIGIN	0.9548	Present	Present
CITY_DESTINATION	0.8757	Present	Present
FARE	**0.0234**	**Absent**	**Present**
CODE_NAME	0.0000	Absent	Absent

Step 3: Each BN performs goal inference again with the corresponding spurious concepts removed.

$P(Goal = \text{Flight\_ID} \mid Query) = 0.9462$ (with <FARE> removed)
$P(Goal = \text{Fare\_ID} \mid Query) = 0.6585$ (with <PRE_TIME> removed)
...
$P(Goal = \text{Aircraft\_Code} \mid Query) = 0.0000$ (no concept is removed)

Step 4: Reject the spurious concepts according to the BN with maximum aposteriori probability.

Since the BN of Flight_ID has maximum aposteriori probability, we will reject <FARE> from the input query.

Step 5: Use the updated query to infer the goal again.

$P(Goal = \text{Flight\_ID} \mid Updated\ Query) = 0.9462$
$P(Goal = \text{Fare\_ID} \mid Updated\ Query) = 0.0023$
...
$P(Goal = \text{Aircraft\_Code} \mid Updated\ Query) = 0.0000$
$\Longrightarrow$ The inferred goal is Flight_ID

Table 3.11: Procedures for recovery from speech recognition errors.

3.8.4 Experiments on Handling Speech Recognition Errors

3.8.4.1 Varying the threshold value θ

Since some of the concepts are sparsely trained, if our threshold θ for rejecting the spurious concepts is too high, some concepts will be wrongly rejected. To illustrate with an example, consider the query:

Query	: *i'd like a one way ticket from milwaukee to orlando either wednesday evening or thursday morning*
Corrupted Query	: *i'd like a one way ticket from milwaukee orlando either wednesday evening or thursday morning* (deletion of <TO> between the two cities)
Tags	: <ONE_WAY> <FROM> <CITY_ORIGIN> <CITY_DESTINATION> <DAY_NAME> <PERIOD> <DAY_NAME> <PERIOD>
Inferred Goal	: Flight_ID (Correct)

Our BN for Flight_ID votes positive for this query. It then performs backward inference and the results in Table 3.12 using 0.5 as threshold indicated that the concepts <DAY_NAME> and <PERIOD> are spurious. Since such of these concepts only act as additional specification for database access, they are optional in the query.

Hence we have conducted a series of experiments in which the value for the threshold θ is varied from 0.1 to 0.5.[7] The inferred goal(s) are considered correct only if they agree with those in the reference SQL. Queries whose goals are not covered by our 11 BNs are treated as OOD, and are considered

[7] The maximum value of the threshold is 0.5 since $P(C) + P(\overline{C}) = 1$.

Concept$_j$ (C_j) (Part of concepts)	$P^*(C_j)$	Binary Decision for C_j	Actual Occurrence for C_j
CITY_ORIGIN	0.9548	Present	Present
CITY_DESTINATION	0.8757	Present	Present
PERIOD	**0.2719**	**Absent**	**Present**
DAY_NAME	**0.3333**	**Absent**	**Present**
FARE	0.0234	Absent	Absent
CODE_NAME	0.0000	Absent	Absent

Table 3.12: Updated probabilities obtained from backward inference (BN of Flight_ID) using 0.5 as threshold for the corrupted query *"i'd like a one way ticket from milwaukee orlando either wednesday evening or thursday mroning"*.

to be identified correctly if there are classified as such. The goal identification accuracies for the noisy training set are shown in Figure 3.8. The curve in Figure 3.8 shows the goal identification accuracies when the spurious words are rejected based on our algorithm using various threshold values, while the straight line shows the goal identification accuracy (81.5%) when the spurious words are not rejected (i.e. no threshold used). By optimizing on overall goal identification accuracy based on the training set, it is suggested that the threshold θ for rejecting the spurious concepts should be set to 0.15.

3.8.4.2 Results

We adopt a suitable threshold ($\theta = 0.15$) to reject the spurious concepts in our algorithm. We have tested the corrupted ATIS-3 1993 test set which is generated randomly based on the algorithm described in Section 3.8.1. Table 3.13 shows the breakdown of the corrupted queries in the 1993 test set. For

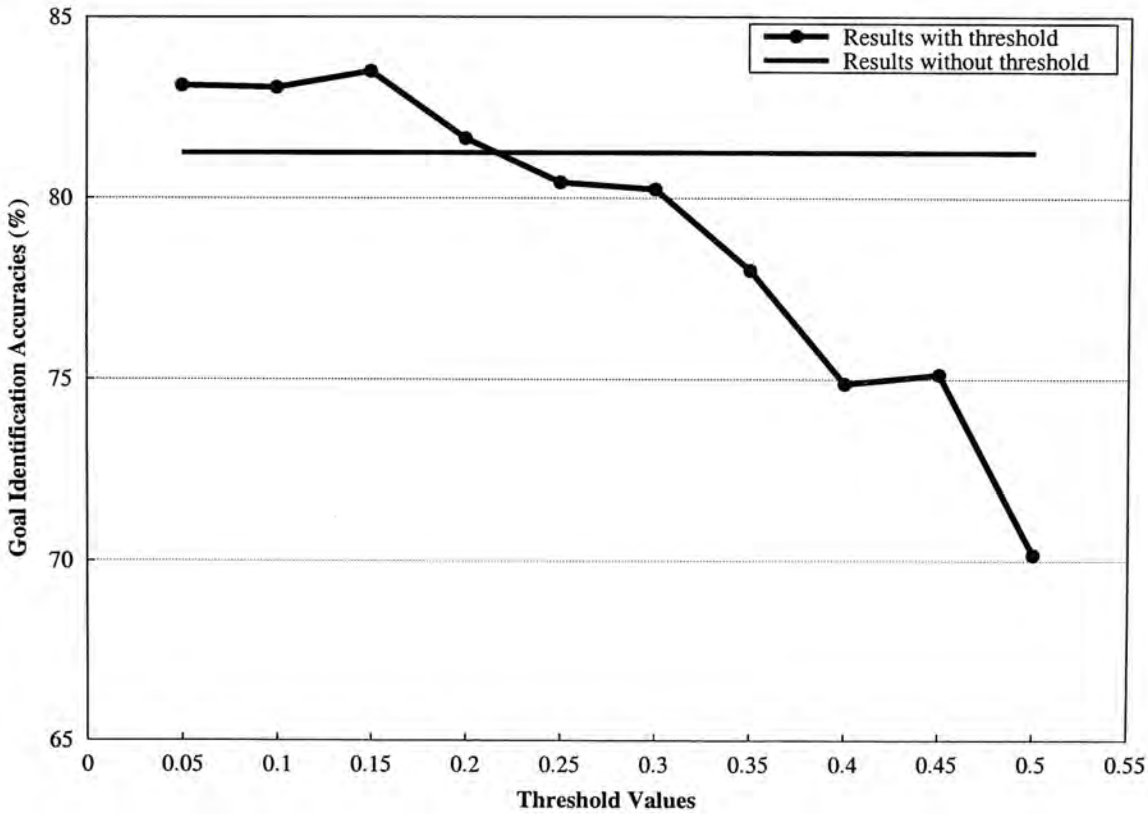

Figure 3.8: Goal identification accuracies of the corrupted training set when threshold θ ranges from 0.05 to 0.5.

the total of 448 test queries, only 262 have been corrupted. The rest of the queries are identical to the original (they are referred to clean queries). We show the comparative results on the original set and the corrupted set in Table 3.14. The set of corrupted queries has degraded from 91.3% to 82.8%. By rejecting the spurious words, the result has been improved by 2% (from 82.8% to 85.0%).

Clean Queries	186
Corrupted Queries	262
Total Queries	448

Table 3.13: Breakdown of the ATIS-3 1993 corrupted test set.

ATIS-3 1993 Test Set	**Original Set**	**Corrupted Set**	**Corrupted Set (reject the spurious concepts)**
Single Goal Correct	381	347	357
Rejection Correct	23	22	22
Multiple Goal Correct	5	2	2
Total Correctly Handled	409/448 (91.3%)	371/448 (82.8%)	381/448 (85.0%)

Table 3.14: Overall goal identification accuracies using the rejection threshold of 0.15 for the ATIS-3 1993 original and corrupted test sets.

3.8.5 Significance Testing

From the corrupted ATIS-3 1993 test set (448 queries in total), 371 (82.8%) queries were identified correctly without the rejection of spurious words; and 381 (85.0%) queries were identified correctly with the rejection of spurious words. We conducted a significance test on the performance difference as follow:

The null hypothesis (H_0) and alternate hypothesis (H_1) are:

$$H_0 : d = p_1 - p_2 = 0$$
$$H_1 : d = p_1 - p_2 < 0$$

where α *is* 0.05 (*significance level*), *hence* $Z_{0.05}$ *is* 0.8289.

p_1 *is proportion of correctly identified queries without the rejection of spurious words.*

p_2 *is proportion of correctly identified queries with the rejection of spurious words.*

The test statistic is:

$$Z_0 = \frac{p_1 - p_2}{\sqrt{\hat{p}(1-\hat{p})\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}} \tag{3.10}$$

where $p_1 = \dfrac{371}{448},\ p_2 = \dfrac{381}{448}$

$n_1 = n_2 = 448$

$\hat{p} = \dfrac{x_1 + x_2}{n_1 + n_2} = \dfrac{371 + 381}{448 + 448} = 0.8393$

We reject H_0 if $Z_0 < -Z_{0.05}$.

Since $Z_0 = -0.9096 < -Z_{0.05}$, we conclude that the rejection of the spurious words which may be due to speech recognition error gives better goal identification accuracies, and the performance difference is statistically significant.

3.8.6 Error Analysis

We found that out of the total 262 corrupted queries in the ATIS-3 1993 corrupted test set (Table 3.13), the goal identification for 211 queries have not been changed. This can be explained that the mis-recognized words do not affect the meaning for understanding (Table 3.15, Example 1). For the rest of the corrupted (51) queries in which the goal identification has been changed, 38 of them (out of 51) which are inferred wrongly with the presence of the mis-recognized words (they are inferred correctly when they are not corrupted). By rejecting the spurious words, 8 queries are recovered and inferred correctly (Table 3.15, Example 2). There are 13 queries (out of 51) which are inferred correctly with the presence of mis-recognized words. They are inferred wrongly when they are not corrupted (Table 3.15, Example 3). For the rest of the 186 clean queries, there are 3 cases of false rejection which lead to an incorrect goal identification (Table 3.15, Example 4).

Example 1 : Mis-recognition does not affect the goal inference.	
Original Query	: *yes i'd like to find a flight from memphis to tacoma stopping in los angeles*
Inferred Goal	: Flight_ID (Correct)
Corrupted Query	: *yes i'd like* ***two friends*** *flight from memphis to tacoma stopping in los angeles*
Inferred Goal	: Flight_ID (Correct)
Example 2 : Mis-recognition causes the incorrect goal inference but can be recovered by our algorithm.	
Original Query	: *cleveland to miami on wednesday arriving before four p m*
Inferred Goal	: Flight_ID (Correct)
Corrupted Query	: *cleveland to miami on* ***to*** *wednesday arriving before* ***fare*** *p m*
Inferred Goal	: Fare_ID (Wrong)
New Inferred Goal (with rejection algorithm)	: Flight_ID (Correct, <FARE> is rejected)
Example 3 : Inferred goal is corrected for the corrupted query, it is inferred wrongly in the original query.	
Original Query	: *what is the seating capacity on the aircraft m eight zero*
Inferred Goal	: Aircraft_Code (Wrong, the goal should be Aircraft_Capacity which is OOD)
Corrupted Query	: *what is the seating capacity on the air* ***cost*** *m eight zero*
Inferred Goal (with or without rejection algorithm)	: OOD (Correct)
Example 4 : False rejection for the clean query.	
Original Query	: *flight number from houston to dallas*
Inferred Goal	: Flight_Number (Correct)
Clean Query	: *flight number from houston to dallas*
Inferred Goal (with rejection algorithm)	: Flight_ID (Wrong, since <Flight_Number> has been rejected by backward inference)

Table 3.15: Example queries and corresponding inferred goals generated by the BN framework with capability of rejection of mis-recognized concepts.

3.9 Chapter Summary

This chapter describes our initial attempt in applying Belief Networks for the identification of informational goals in information-seeking queries. The BNs can model the causal relations between the query's semantic concepts and the underlying informational goal(s). By formulating our N-way classification problem as N binary classifications, we are able to (i) identify queries with multiple informational goals, and (ii) reject queries whose goals are outside of the prescribed knowledge domain, without significant loss in goal classification performance. Our experiments also found IG and the F-measure to be favorable, for their respective tasks of feature selection and probability thresholding in binary classifications. We also have extended the use of BNs from natural language understanding to the detection of misrecognized concepts in the spoken queries. In order to capture the domain-speccific constraints with the inter-concept dependencies, we adopt the Minimum Description Length (MDL) principle to automatically learn the least complex topologies for the BNs. User input can then be verified against the domain-specific constraints by using backward inference. Therefore, spurious concepts which are caused by speech recognition errors can be detected and rejected. By rejecting the spurious concepts within the spoken query automatically, an improvement in goal identification accuracy was realized, from 82.8% to 85.0%. The improvement was statistically significant.

Chapter 4

Belief Networks for Mixed-Initiative Dialog Modeling

We have demonstrated the feasibility of using Belief Networks (BNs) for natural language understanding in the previous chapter. However, in spoken dialog systems, most of the users do not specify all the required information attributes within a query, and they usually engage with the system for several dialog turns in order to achieve their goals. Among the various dialog strategies of spoken dialog systems, the *mixed-initiative* model is deemed most desirable, since *both* the user and the system can influence the dialog flow over the course of interaction. This provides greater flexibility than the system-initiative model or user-initiative model. It is possible to build effective mixed-initiative interactions by *handcrafting* flexible transitions between the system-initiative and user-initiative models. While handcrafting can produce a sophisticated dialog flow, the task is expensive, and may become intractable with increasingly complex domains. In this chapter, we propose to use Belief Networks to automatically govern the model transitions for *mixed-initiative* interactions. We have extended the BN framework, previously used for natural language understanding, for the current task of mixed-initiative dialog modeling within the context of the CU FOREX system.

4.1 The CU FOREX Domain

4.1.1 Domain-Specific Constraints

We have chosen to investigate the feasibility of using BNs for dialog modeling, within the context of the CU FOREX system [34]. This is a bilingual (English and Cantonese) conversational hotline that supports inquiries about foreign exchange information. It supports inquiries regarding the exchange rates between a currency pair, as well as the interest rates for various time durations for a currency. The domain is relatively simple, and is characterized by two query types (or informational goals - Exchange Rates and Interest Rates); and five domain-specific concepts (a CURRENCY PAIR, TIME DURATION, EXCHANGE RATE and INTEREST RATE). For database retrieval, there are two constraints: An inquiry regarding exchange rates mandates that the currencies to be bought and sold be specified. An inquiry regarding interest rates mandates that a currency and a time duration be specified.

4.1.2 Two Interaction Modalities

The CU FOREX has been made available for experimentation by the public since August 1999. Since then, we have received a few hundred calls per month on average. It currently supports two interaction modalities, based on SpeechWorks 4.0 [34]. The Directed Dialog (DD), designed for novice users, guides the user through a session and elucidates what may be said at various stages of the interaction. Table 4.1 shows a directed dialog (DD) interaction between the system and the user. Notice that at every dialog turn, the system guides the user to provide a specific information attribute. The Natural Language Shortcut (NLS) is designed for expert users who want to expedite the inquiry session by uttering a full query, thus traversing the entire session within one interaction. Hence the query may carry multiple attributes for database retrieval, as opposed to a single attribute per utterance in the directed dialog case. Example dialog for Natural Language Shortcut interaction is shown in Table 4.2.

4.2 The Belief Networks

We adopt the use of BNs in mixed-initiative dialog modeling. It involves the processes of (i) inferring the informational goal of a user's query, as well as (ii) verifying the input query against domain-specific constraints. To identify the user's informational goal, we have devised a framework that utilizes BNs, as described in Chapter 3. A brief description is provided as follows.

System	:	"*Welcome to* CU FOREX. *Which language would you prefer, Cantonese or English?*"
User	:	"*English.*"
System	:	"*Would you like exchange rates or interest rates?*"
User	:	"*Interest rates.*"
System	:	"*Which currency are you interested in?*"
User	:	"*Hong Kong dollar.*"
System	:	"*Would you like deposit durations for twenty-four hours, one month ... or one year?*"
User	:	"*One month.*"
System	:	"*The quotes we provided are for reference only. The interest rate of the Hong Kong dollar for one month is XXX.*"

Table 4.1: Directed dialog (DD) interaction between the system and the user, for an interest rates inquiry.

System	:	"*Welcome to* CU FOREX. *Which language would you prefer, Cantonese or English?*"
User	:	"*English.*"
System	:	"*What kind of currency information are you interested in?*"
User	:	"*I'd like to know the exchange rates between the US dollar and the Hong Kong dollar please.*"
System	:	"*The quotes we provided are for reference only. Exchange rate: The US dollar to the Hong Kong dollar, the buying rate is XXX, the selling rate is XXX.*"

Table 4.2: Natural language shortcut (NLS) interaction between the system and the user, for an exchange rates inquiry.

4.2.1 Informational Goal Inference

A BN is trained for each domain-specific informational goal. In this domain, there are two informational goals - Exchange Rates and Interest Rates. Hence we developed two BNs, one for each goal, using the natural language queries we have collected.[1] Each BN receives as input all of the five domain-specific concepts: CURRENCY_1, CURRENCY_2, DURATION, EXCHANGE_RATE and INTEREST_RATE.[2] We have also enhanced the pre-defined topology by means of automatic learning using the Minimum Description Length (MDL) principle [33]. The resulting topology is illustrated in Figure 4.1. Notice that it captures not only the causal dependencies between the information goal and the corresponding concepts, but it also shows the dependencies between the concepts (i.e. the dotted arrow).

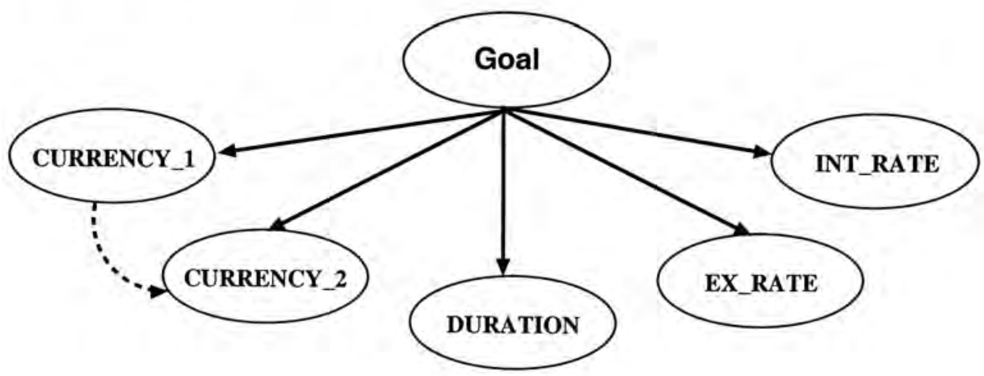

Figure 4.1: The enhanced topology of our BNs in the CU FOREX domain.

A trained BN is then used to make a binary decision based on the concepts present in the input query, regarding the presence or absence of the goal G_i. As mentioned before, in order to compute the likelihood of the goal given the

[1] We have transcribed around 500 natural language queries for training the BNs.

[2] Since our domain is relatively simple, we did not select the input concepts for the BN using the Information Gain criterion.

present concepts, the learnt network should first be divided into several sub-networks: {Goal, CURRENCY_1, CURRENCY_2}, {Goal, DURATION}, {Goal, EX_RATE} and {Goal, INT_RATE}. The updated joint probabilities $P^*(G_i, \vec{C})$ are then iteratively computed according to Equation 3.8 by each sub-network, and the updated probability $P^*(G_i)$ can be obtained by the marginalization of the updated joint probability. $P^*(G_i)$ is then compared to a threshold to make the binary decision — θ may be set to 0.5 ($\theta_{0.5}$), or an optimized value for each goal by F-measure (θ_f). Details for computing the updated probability $P^*(G_i)$ by Bayesian inferencing can be found in Appendix C.

The decisions across all the BNs are combined to identify the output goal of an input query. We can adopt a goal if the corresponding BN votes positive with the highest updated probability $P^*(G_i)$. Alternatively, we may label the query with all the goals for which the BNs vote positive. Should all BNs vote negative, the input query is rejected as OOD. Consider an example query: "*Can I have the exchange rate of the yen please*". Typical values of the updated probabilities that are obtained from goal inference of the two BNs are shown in Table 4.3. These values are compared with a pre-set threshold $\theta_{0.5}$ for making the binary decision and the input is hence classified as an Exchange Rates query. For the other example query: "*tell me about stock quotes*". The two BNs both vote negative for the query as depicted in Table 4.3 and hence the input is classified as OOD.

Query: "*Can I have the exchange rate of the Yen please?*"
BN for Exchange Rates: $P^*(Goal =$ Exchange Rates$) = 0.823 \rightarrow$ goal present
BN for Interest Rates: $P^*(Goal =$ Interest Rates$) = 0.256 \rightarrow$ goal absent
Hence, the inferred goal is Exchange Rates.

Query: "*Tell me about stock quotes.*"
BN for Exchange Rates: $P^*(Goal =$ Exchange Rates$) = 0.14 \rightarrow$ goal absent
BN for Interest Rates: $P^*(Goal =$ Interest Rates$) = 0.13 \rightarrow$ goal absent
Hence, the user's query is considered as OOD.

Table 4.3: Typical values of the updated probabilities obtained from goal inference using BNs in the CU FOREX domain.

4.2.2 Detection of Missing / Spurious Concepts

Having inferred the informational goal of the query, the corresponding goal node is instantiated, and we perform a backward inference to test the network's confidence in each input concept. When the goal node is instantiated, the joint probability of $P(\vec{C}, G_i)$ will be updated for each sub-network as in Equation 3.9.

By marginalization of the updated joint probability, we can get $P^*(C_j)$. This probability is compared with the threshold $\theta = 0.5$ to determine whether the concept should be present or absent. Details for obtaining the updated probability $P^*(C_j)$ by backward inferencing can be found in Appendix C.

$$P^*(C_j) = \begin{cases} \geq \theta & \rightarrow \quad C_j \text{ should be } \textit{present} \text{ in the given } G_i \text{ query} \\ < \theta & \rightarrow \quad C_j \text{ should be } \textit{absent} \text{ in the given } G_i \text{ query} \end{cases}$$

Backward inference *verifies* the validity of the input query against domain-specific constraints. In this way, we can test for cases of *spurious* and *missing* concepts,[3] and generate the appropriate system responses.

As an example, consider an interest rates query "*Can I have the interest rates of the Yen for one month please?*". We instantiated the goal node of the BN (for Interest Rates) to 1, and performed backward inference for each input concept C_j to obtain $P^*(C_j)$. The associated probabilities and binary decisions are shown in Table 4.4. The corresponding binary decisions obtained for each concept (using $\theta = 0.5$) agrees with the actual occurrences in the input query. The semantic frame is thus ready to be processed for database retrieval.

Concept$_j$ (C_j)	$P^*(C_j)$	**Binary Decision for** C_j	**Actual Occurrence for** C_j
CURRENCY_1	0.910	Present	Present
CURRENCY_2	0.006	Absent	Absent
DURATION	0.770	Present	Present
EX_RATE	0.011	Absent	Absent
INT_RATE	0.867	Present	Present

Table 4.4: The binary decision for each concept from backward inference (BN of Interest Rates) agrees with their actual occurrence.

[3] These may be due to speech recognition errors in an integrated spoken dialog system.

However, in situations where the binary decision for each concept *disagree* with its actual occurrence, further processing is necessary. The following shows two cases:

- **Case 1: Missing Concepts**

  If the binary decision for *concept*$_j$ is positive but it is absent in the input query, a missing concept is detected. The dialog model is designed such that the system will prompt for the missing concept. Table 4.5 illustrates the associated probabilities and binary decisions from backward inference for an interest rates query "*Can I have the interest rate of the Yen?*". Backward inference produces $P^*(C_j)$ and thresholding gives the binary decision regarding the presence or absence of C_j. When we compare the binary decision from backward inference and the actual occurrences of C_j in the input query, we detect that the concept <DURATION> is missing. This invokes the dialog model to generate the prompt, "*How long would you like to deposit?*".

Concept$_j$ (C_j)	$P^*(C_j)$	**Binary Decision for C_j**	**Actual Occurrence for C_j**
CURRENCY_1	0.910	Present	Present
CURRENCY_2	0.006	Absent	Absent
DURATION	**0.770**	**Present**	**Absent**
EX_RATE	0.011	Absent	Absent
INT_RATE	0.867	Present	Present

Table 4.5: The binary decision of the concept <DURATION> from backward inference (BN of Interest Rates) does not agree with the actual occurrence. Hence <DURATION> is deemed missing and the dialog model prompts for it.

- **Case 2: Spurious concepts**

  Should a spurious concept be detected, i.e. the presence of $concept_j$ violates the binary decision, the system would automatically ask the user for clarification. For example, consider the query "*Can I have the interest rate of the Lira against the Yen*", the inferred goal is Exchange Rates (where $P^*(Goal = \text{Exchange Rates}) = 0.79$). The corresponding probabilities and binary decisions from backward inferencing are shown in Table 4.6. It can be seen that the concept <INT_RATE> is spurious. The BN for Exchange Rates indicates that this concept should not be present. Hence, the system will follow the inferred goal (Exchange Rates) to generate the clarification response: "*Are you referring to the exchange rate between the Lira and the Yen?*".

$Concept_j$ (C_j)	$P^*(C_j)$	**Binary Decision for C_j**	**Actual Occurrence for C_j**
CURRENCY_1	0.910	Present	Present
CURRENCY_2	0.920	Present	Present
DURATION	0.017	Absent	Absent
EX_RATE	0.840	Present	Absent
INT_RATE	**0.023**	**Absent**	**Present**

Table 4.6: The binary decision of the concept <INT_RATE> from backward inference (BN of Exchange Rates) conflicts with the actual occurrence. Hence <INT_RATE> is deemed spurious and the dialog model issues a clarification response.

4.3 Integrating Two Interaction Modalities

We have developed a dialog system using BNs. Automatic detection of missing / spurious concepts drives the mixed-initiative dialog model and hence integrates the two interactions modalities (DD, NLS). The example dialogs in Table 4.7 show that the BN-based dialog model can handle pure DD or NLS as well as mixed-initiative interactions. It can prompt for missing concepts, ask for clarification from the user when spurious concepts are detected, and reject OOD input.

Dialogs 1 and 2 in Table 4.7 show that the BN-based dialog model can produce pure DD or NLS interactions identical to the those of previous dialog models illustrated in Tables 4.1 and 4.2. In the system-initiative DD interaction, the BN-based dialog model detects the missing concepts and prompts the user until all concept occurrences agree with the domain-specific constraints. In the user-initiative NLS interaction, the concepts agree with the domain-specific constraints and hence the system is able to produce a response directly.

The BN-based dialog model can also freely transit between the DD and NLS interactions. Users are no longer constrained to input one information attribute per dialog turn. In addition, the system is capable of prompting for missing information as shown in Dialog 3, or clarifying confusion due to spurious concepts in Dialog 4.

The system is also able to provide a mixed-initiative interaction with either novice or expert users in a single spoken interface in Dialog 5, as well as reject OOD input automatically as shown in Dialog 6.

Dialog 1:	**Pure DD, asking for exchange rates information.**
System :	"What kind of currency information are you interested in?"
User :	"*Exchange rates.*"
System :	"Which currency would you like to sell?"
User :	"*Hong Kong dollar.*"
System :	"Which currency would you like to buy?"
User :	"*US dollar.*"
System :	"The exchange rate of the Hong Kong dollar against the US dollar is XXX."
Dialog 2:	**Pure NLS interaction, asking for interest rates information.**
System :	"What kind of currency information are you interested in?"
User :	"*I would like the interest rate of the Yen for one year.*"
System :	"The interest rate of the Yen for one year is XXX."
Dialog 3:	**Prompting for missing information.**
System :	"What kind of currency information are you interested in?"
User :	"*I would like the interest rate of the Yen.*" (NLS interaction)
System :	"How long would you like to deposit?"
User :	"*Two months.*" (DD interaction)
System :	"The interest rate of the Yen for two months is XXX."
Dialog 4:	**Asking for clarification upon the detection of spurious concepts.**
System :	"What kind of currency information are you interested in?"
User :	"*Interest rate of the Hong Kong dollar against the Yen.*"
System :	"Are you referring to the exchange rate of the Hong Kong dollar against the Yen?"
User :	"*Yes please.*"
System :	"The exchange rate of the Hong Kong dollar against the Yen is XXX."
Dialog 5:	**Handling mixed-initiative interaction.**
System :	"What kind of currency information are you interested in?"
User :	"*Exchange rate.*" (DD interaction)
System :	"Which currency would you like to sell?"
User :	"*Hong Kong dollar against the US dollar.*" (NLS interaction)
System :	"The exchange rate of the Hong Kong dollar against the US dollar is XXX."
Dialog 6:	**Handling OOD query.**
System :	"What kind of currency information are you interested in?"
User :	"*Tell me about stock quotes.*" (OOD query)
System :	"Sorry we don't have the information you want. Please input service, exchange rates or interest rates."

Table 4.7: Example dialogs produced by the BN-based dialog model.

4.4 Incorporating Out-of-Vocabulary Words

When users interact with a dialog system, they usually do not know the system's capability and they may therefore inquire for information that is outside the scope of the system's competence. Hence it is important for a dialog system to detect the out-of-vocabulary (OOV) words and guide the users with proper instructions. In the CU FOREX system, we observed that 3.6% of users have asked about unknown currencies which resulted in incorrect responses from the system. In this task, we aim to incorporate the capability of OOV detection in the BN-based dialog model in the CU FOREX domain.

4.4.1 Natural Language Queries

Detection of missing concepts from the user query may imply that the user has input some unseen vocabulary items which lead to parse failures. For example, for the query illustrated in Table 4.8 "*What is the interest rate of Finland Markka for three months?*", since "*Finland Markka*" lies outside of our grammar's terminal categories, it is tagged as <OOV>. The BN dialog model detects a missing concept of <CURRENCY> from backward inference. Instead of prompting the user for missing concepts, the model is designed to examine the <OOV> tag first.

We make use of the contextual information to identify a possible tag for the <OOV>. Based on the training data of the natural language spoken queries, we have designed a few contextual rules as illustrated in Table 4.9 to transform the <OOV> to an unseen currency (<OOV_CURRENCY>).

System :	"*What kind of currency information are you interested in?*"
User :	"*What is the interest rate of* ***Finland Markka*** *for three months?*" <WHAT> <DUMMY> <INT_RATE> <PREP> <OOV> <PREP> <DURATION>
System :	[Detection of a missing concept <CURRENCY>]

Table 4.8: Occurrence of OOV words in a natural language query.

Rule	Occurrence of the tag order	Then
1.	<INT_RATE> <PREP> <OOV> <PREP> <DURATION>	<OOV>→ <OOV_CURRENCY>
2.	<INT_RATE> <PREP> <OOV>	
3.	<EX_RATE> <PREP> <OOV> <AGAINST> <CURRENCY>	

Table 4.9: Some of the contextual rules for transforming the <OOV> tag to <OOV_CURRENCY>.

Our example query in Table 4.8 matches Rule 1, in which the <OOV> is preceded by bigram tags <INT_RATE> <PREP> and succeeded by bigram tags <PREP> <DURATION>. Therefore the <OOV> tag is induced as <OOV_CURRENCY>.

4.4.2 Directed Queries

We can use the contextual information in the natural language queries to identify a possible tag for the <OOV>. However, for directed queries whereby the user usually responds to the system with a specific attribute, we can only determine a possible tag for the <OOV> based on the dialog flow. An example illustrated in Table 4.10 shows that the occurrence of the <OOV> in a directed dialog interaction will be induced as <OOV_CURRENCY> based on the dialog context.

System	:	"*What kind of currency information are you interested in?*"
User	:	"*Can I have the exchange rate please?*"
System	:	"*Please tell me the currency you want to sell.*"
User	:	"***Finland Markka***" [Directed dialog response] <OOV> will be induced as <OOV_CURRENCY>

Table 4.10: Based on the dialog context, we can determine the <OOV> to be an unseen currency <OOV_CURRENCY> .

After the system has determined the <OOV> as an <OOV_CURRENCY>, it will ask the user for clarification, which results in 2 cases:

- **Case 1: The assumption of the OOV words as a currency is correct**

  If the user confirms the unseen words <OOV_CURRENCY> as a currency, our system will automatically add the new currency to the lexicon. A correct response will then be provided for the user. An example dialog is shown in Table 4.11.

- **Case 2: The assumption of the OOV words as a currency is wrong**

  Sometimes, the occurrence of OOV words may be due to speech recognition errors. As a result, if the BN-based dialog model induces the <OOV_CURRENCY> wrongly as a currency, the system will then prompt the user again for the missing concepts. An example dialog is shown in Table 4.12.

System	:	"*What kind of currency information are you interested in?*"
User	:	"*What is the interest rate of* ***Finland Markka*** *for three months?*"
System	:	"*The currency you are referring is Finland Markka, is that correct?*"
User	:	"*Yes.*"
System	:	"*The interest rate of Finland Markka for three months is XXX.*" *Or* "*Sorry, we do not have the currency information you want.*"

Table 4.11: The user confirms the OOV word as a currency.

System :	"*What kind of currency information are you interested in?*"
User :	"*Can I have the exchange rate please?*
System :	"*Please tell me the currency you want to sell?*
User :	"***E U.***" (E U lies outside of our lexicon and it is tagged as <OOV>. It becomes <OOV_CURRENCY> based on the dialog flow.)
System :	"***The currency you are referring is E U, is that correct?***"
User :	"*No.*"
System :	"***Please tell me the currency you want to sell?***"
User :	"***E C U.***"

Table 4.12: The OOV words may be caused by a speech recognition error, in which case the system will prompt the user for the missing currency again.

4.5 Evaluation of the BN-based Dialog Model

Our evaluation is based on 550 dialog sessions collected using the CU FOREX system during the period between November and December 1999. Approximately 17% were rejected manually, as the users were clearly attempting to break the system. Of the remaining queries, 285 sessions were obtained from the DD while 170 sessions were obtained from NLS hotlines. The failure point recovery rates of the DD and NLS models are shown in Table 4.13. Failures in the DD model are mainly caused by queries with multiple information attributes, OOV or OOD input. Failures in the NLS model are due to missing, spurious or OOV concepts or OOD queries. Detailed statistics are tabulated in Table 4.14. In comparison, the BN-based mixed-initiative dialog model can automatically reject OOD input, and successfully handle *all* the dialogs. Table 4.15 shows the typical causes of failure in the original DD and NLS models, in which the BN-based dialog model can handle all the cases well and offer a continuation option at the failure point.

	Failure Point Recovery Rate	
	CU FOREX	BN-based Dialog Model
DD (total: 285 sessions)	85%	100%
NLS (total: 170 sessions)	63%	100%

Table 4.13: Failure point recovery rates of the original CU FOREX system with Directed Dialog (DD) and Natural Language Shortcut (NLS) interactions, in comparison with failure point recovery rates from the BN-based dialog model.

	Failures for the CU FOREX Dialog Model			
	Multiple Attributes	Missing Concepts	Spurious Concepts	OOV/ OOD
DD	11%	—	—	4%
NLS	—	30%	4%	3%

Table 4.14: Causes of failure for the CU FOREX dialog model. Percentages refer to the proportion of the evaluated queries (285 DD and 170 NLS).

Dialog 1: DD Interaction	
System	: *"Would you like exchange rates or interest rates?"*
User	: *"Exchange rates."*
System	: *"Which currency you would like to sell?"*
User	: *"Yen to Hong Kong dollars please."*
System (CU FOREX)	: Failed. (Multiple information attributes in a DD interaction.)
System (BN-based Dialog Model)	: *"The exchange rate between the Yen and the Hong Kong dollar is XXX."* (Concepts pass the domain constraints.)
Dialog 2: NLS Interaction	
System	: *"What kind of currency information are you interested in?"*
User	: *"Tell me about interest rates."*
System (CU FOREX)	: Failed. (Missing concepts: <CURRENCY_1> and <DURATION>.)
System (BN-based Dialog Model)	: *"Please tell me the currency you are interested in."* (Prompt for the missing concepts.)

Table 4.15: Comparison of the DD / NLS interactions between the original CU FOREX system and the BN-based dialog model.

4.6 Chapter Summary

This chapter describes our first attempt in applying Belief Networks for dialog modeling in a simple foreign exchange domain. The topologies of the BNs are designed to capture domain-specific constraints. The presence / absence of concepts are used to infer the presence / absence of each goal, by means of the BN. Automatic detection of missing / spurious concepts is achieved by backward inference using the BN corresponding to the inferred goal. This detection procedure hence invokes the dialog model to prompt for missing information, and ask for clarification. Besides, our dialog mode is enhanced to incorporate the OOV currency. In the next chapter, we will investigate the domain portability issue as well as testing our BN-based dialog model in a more complex domain.

Chapter 5

Scalability and Portability of Belief Network-based Dialog Model

In this chapter, we will explore the scalability and portability of our proposed Belief Network (BN)-based dialog model across application domains. In dialog modeling, the BNs are used to automatically govern the transitions between a system-initiative and a user-initiative dialog model, in order to produce mixed-initiative interactions. We have already demonstrated its feasibility in the simple CU FOREX (foreign exchange) domain, which is characterized by two query types and five domain-specific concepts. In this work, we will migrate our dialog model from the simple domain of foreign exchange to a more complex domain of air travel information service (ATIS).

5.1 Migration to the ATIS Domain

We will test the BN-based dialog model in a more complex domain. We have chosen the ATIS (Air Travel Information Service) domain due to data availability. ATIS is characterized by 11 query types and 60 hand-designed semantic tags. Hence we need to scale up our BN-based dialog model for the ATIS application. We have used the ATIS-3 Class A (context-independent) queries in our BN-based framework for natural language understanding as described in Chapter 3. However, in the current task of dialog modeling, both the ATIS-3 Class A and Class D (context-dependent) queries are used. Class A queries are self-contained while interpretation of the Class D queries is dependent on the discourse context.

The disjoint training and test sets consist of 2820, 773 (1993 test), and 732 (1994 test) transcribed utterances, with the breakdown of Class A and D queries tabulated in Table 5.1. Each utterance is accompanied with its corresponding SQL query for retrieving the relevant information. Some examples are illustrated in Table 5.2. Notice that the SQL query of the Class D utterance is associated with additional semantic concepts from the earlier discourse.

ATIS-3	Training	1993 Test	1994 Test
Class A Queries	1564	448	444
Class D Queries	1256	325	288
Total	**2820**	**773**	**732**

Table 5.1: Breakdown of ATIS-3 Class A and D queries.

Query (Class A)	: *"find me a flight that flies from memphis to tacoma"*
SQL	: select *flight_id* from *flight* where *origin* = *"memphis"* and *destination* = *"tacoma"*
Query (Class D)	: *"which of these flights stop in los angeles"*
SQL	: select *flight_id* from *flight* where *origin* = *"memphis"* and *destination* = *"tacoma"* and *stopover* = *"los angeles"*
Query (Class D)	: *"does that flight serve dinner"*
SQL	: select *flight_id* from *flight* where *origin* = *"memphis"* and *destination* = *"tacoma"* and *stopover* = *"los angeles"* and *meal_description* = *"dinner"*

Table 5.2: Sample ATIS Class A and Class D queries. They are in sequential order and belong to the same dialog session.

5.2 Scalability of the BN-based Dialog Model

5.2.1 Informational Goal Inference

There are a total of 60 hand-designed semantic concepts in the ATIS domain. In order to constrain computation time for goal inference, we have limited the number of semantic concepts (M) that are indicative of each goal G_i. The parameter M (=20) has been selected using the Information Gain criterion as described in Section 3.5.1 to optimize on overall goal identification accuracy with reference to the Class A training utterances.

We have also refined the pre-defined BN topology using the Minimum Description Length (MDL) principle to model concept dependencies. An example of the BN for Aircraft_Code is shown in Figure 5.1.

Each BN has a classification-based network topology — there are M (= 20)

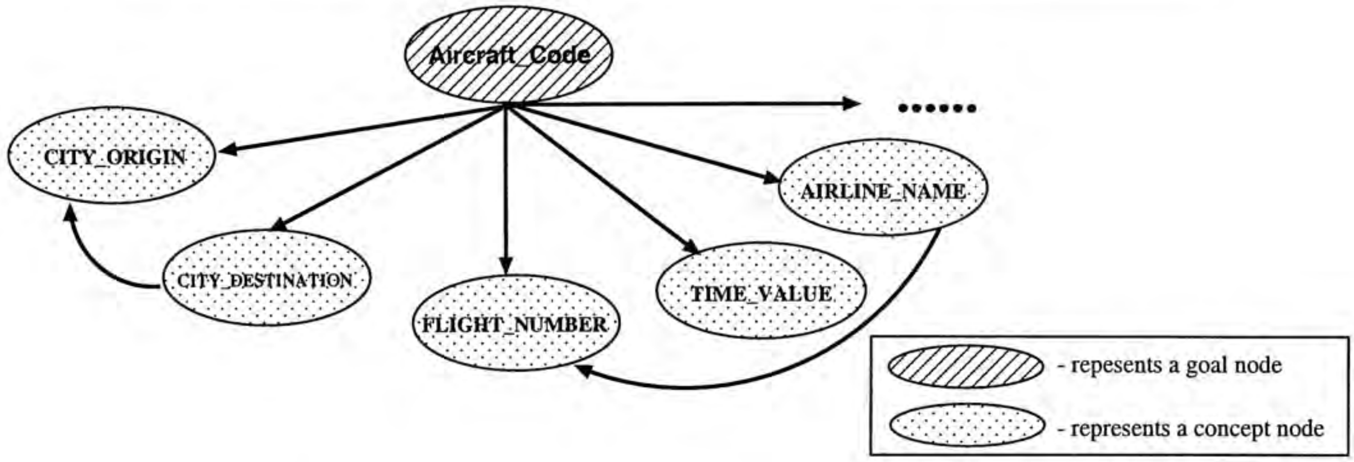

Figure 5.1: Topology of the BN for the informational goal Aircraft_Code.

input concept nodes (e.g. AIRLINE_NAME, FLIGHT_NUMBER, etc.) and a single output node. To avoid the use of sparsely trained BNs, we have developed 11 BNs to capture the domain-specific constraints for each informational goal using Class A training data. The remaining goals are treated as OOD.

A trained BN is used to infer the presence / absence of its corresponding informational goal, based on the input concepts. The updated joint probabilities are iteratively computed by each sub-network, and the updated probability $P^*(G_i)$ is computed by the marginalization of the updated joint probability $P^*(G_i, \vec{C})$. Thresholding $P^*(G_i)$ with the threshold of 0.5 ($\theta_{0.5}$) determines the presence or absence of the goal G_i.

The binary decisions across all BNs are combined to identify the informational goal of the input query. We may label the query to a goal based on the maximum aposteriori rule. Alternatively, we may label the query with all the goals for which the BNs vote positive. Should all BNs vote negative, the input query is rejected as OOD.

5.2.2 Detection of Missing / Spurious Concepts

Having inferred the informational goal of the query, the corresponding goal node is instantiated, and we perform backward inference to obtain $P^*(C_j)$. Comparing this probability with the threshold ($\theta = 0.5$) decides whether the concept C_j should be present or absent according to domain-specific constraints. The binary decision is compared with the actual occurrence of the concept C_j in the input query for detecting missing or spurious concepts. These are, in turn, used to drive the dialog model. However, as we migrated from the CU FOREX domain to the ATIS domain, we discovered that this methodology often produces several missing or spurious concepts for an input query. For example, consider the query:

Query	: "*what type of aircraft is used in american airlines flight number seventeen twenty three*"
Concepts	: <WHAT> <TYPE> <AIRCRAFT> <AIRLINE_NAME> <FLIGHT_NUMBER>
Inferred Goal	: Aircraft_Code (Correct)

Our BN for Aircraft_Code performed backward inference and the updated probabilities $P^*(C_j)$ obtained are shown in Table 5.3. Our detection algorithm labels the concepts <CITY_ORIGIN> and <CITY_DESTINATION> to be missing, and <FLIGHT_NUMBER> to be spurious. One reason is because in the training data, most queries with the goal Aircraft_Code provided the city pair instead of the flight number, but both serve equally well as an additional specification for database access. If our dialog model followed through with these detected missing and spurious concepts, it would prompt the user for the city of origin, then the city of destination; and then clarify that the flight number is spurious. Such a dialog model has too much redundancy, and fails to realise that the attribute pair (AIRLINE_LINE, FLIGHT_NUMBER) provides an equivalent amount of constraints as the attribute pair (CITY_ORIGIN, CITY_DESTINATION) for retrieving an Aircraft_Code query. As a result, we defined two thresholds for backward inferencing, as follows:

$$P^*(C_j) = \begin{cases} \geq \theta_{upper} & \rightarrow \quad C_j \text{ should be } \textit{present} \text{ in the given } G_i \text{ query} \\ < \theta_{upper} \text{ and } \geq \theta_{lower} & \rightarrow \quad C_j \text{ is } \textit{optional} \text{ in the given } G_i \text{ query} \\ < \theta_{lower} & \rightarrow \quad C_j \text{ should be } \textit{absent} \text{ in the given } G_i \text{ query} \end{cases}$$

Concept$_j$ (C_j) **(Subset)**	$P^*(C_j)$	**Binary Decision for** C_j	**Actual Occurrence for** C_j
AIRCRAFT	1.000	Present	Present
AIRLINE_NAME	0.538	Present	Present
CITY_ORIGIN	0.645	Present	Absent
CITY_DESTINATION	0.615	Present	Absent
TIME_VALUE	0.201	Absent	Absent
FLIGHT_NUMBER	0.420	Absent	Present

Table 5.3: Updated probabilities obtained from backward inferencing (BN of Aircraft_Code) using 0.5 as threshold for the query *"what type of aircraft is used in american airlines flight number seventeen twenty three"*.

Hence concepts whose probabilities (from backward inference) score between θ_{upper} and θ_{lower} will not take effect in response generation (i.e. prompting / clarification). Concepts whose scores exceed θ_{upper}, if they also correspond to an SQL attribute, will be prompted if missing; concepts whose scores scant θ_{lower}, if they correspond to an SQL attribute, will be clarified if spurious. By minimizing number of dialog turns interacting with the users in the training data, we have empirically adopted 0.7 and 0.2 for θ_{upper} and θ_{lower} respectively. The double threshold scheme enables the dialog model to prompt for missing concepts that are truly needed, and clarify for spurious concepts that may confuse the query's interpretation.

5.2.3 Context Inheritance

The ATIS corpus contain both Class A and Class D queries. While the semantics of the Class A queries are self-contained, those of the Class D queries are context-dependent. Interpretation of the Class D queries requires referencing discourse context from previous dialog turns. Consequently, we have enhanced our BN-based dialog model with the capability of context inheritance for handling ATIS queries.

In our initial approach to context inheritance, the current query inherits all the semantic concepts from the previous query (of the same dialog session) prior to goal inference. However, we found that this scheme is too aggressive, and the extra concepts affected the goal identification performance. An example is shown in Table 5.4.

System	:	*"what kind of flight information are you interested in"*
User	:	*"find me a flight from cincinnati to westchester county arriving next saturday before six p m"*
(Concepts)	:	<CITY_ORIGIN> <CITY_DESTINATION> <DAY_NAME> <TIME>
System	:	*Goal Inference:* Flight_ID (Correct)
User	:	*"tell me the airports in new york city area"*
(Concepts)	:	<CITY_ORIGIN> <CITY_DESTINATION> <DAY_NAME> <TIME> <AIRPORT> <CITY_NAME> (All concepts from previous query are inherited)
System	:	*Goal Inference:* Flight_ID (Wrong, the correct goal is Airport_Code, the additional concepts inherited from previous query result in the wrong identification of the informational goal.)

Table 5.4: Additional inherited concepts affect the goal identification performance.

As a result, context inheritance will be invoked for goal inference *only if* the query was previously (prior to context inheritance) classified as OOD. Otherwise, the original inferred goal of the query is maintained. For example, our previously illustrated query in Table 5.4 *"tell me the airports in new york city area"*, is inferred as Airport_Code correctly prior to the context inheritance. Table 5.5 illustrates another example in which the system retrieves the concepts from the discourse context and infers the goal again when the query is first classified as OOD.

Besides, context inheritance serves to fill in the concepts detected missing from the original query. An example ATIS dialog is shown in Table 5.6. The Class D Fare_ID query (*"for this flight how much would a first class fare cost"*) mandates the <CITY_ORIGIN> and <CITY_DESTINATION>. Hence the system will automatically retrieve these concepts from the discourse context.

System	:	*"what kind of flight information are you interested in"*
User	:	*"i'd like to fly from miami to chicago on american airlines"* (Class A)
(Concepts)	:	<FLIGHT> <FROM> <CITY_ORIGIN> <TO> <CITY_DESTINATION> <PREP> <AIRLINE_NAME>
System	:	*Goal Inference:* Flight_ID (Correct)
User	:	*"which ones arrive around five p m"* (Class D)
(Concepts)	:	<TO> <TIME_VALUE> (From current query) <FLIGHT> <FROM> <CITY_ORIGIN> <TO> <CITY_DESTINATION> <PREP> <AIRLINE_NAME> (Inherited from previous query)
System	:	*Goal Inference:* Flight_ID (Correct. The query is first classified as OOD, we try to infer the goal again (becomes Flight_ID) after the inheritance of discourse context.)

Table 5.5: Examples of ATIS dialogs produced by the BN-based dialog model. The system invokes goal inference again when the query is first classified as OOD.

System :	*"what kind of flight information are you interested in?"*
User :	*"please list all the flights from chicago to kansas city on june seventeenth."* (Class A)
System :	*Goal Inference:* Flight_ID (Correct, the decisions from backward inference agree with actual occurrence of the concepts)
User :	*"for this flight how much would a first class fare cost."* (Class D)
System :	*Goal Inference:* Fare_ID (Correct) (Backward inference results as shown below using double threshold scheme indicating that the concepts <CITY_ORIGIN> and <CITY_DESTINATION> are missing. These concepts are automatically retrieved from the discourse context.)

$Concept_j$ (C_j) **(subset)**	$P^*(C_j)$	**Decision for** C_j	**Actual Occurrence for** C_j
AIRPORT_NAME	0.0000	Absent	Absent
CITY_ORIGIN	0.9629	Present	Absent
CITY_DESTINATION	0.9629	Present	Absent
CLASS_NAME	0.2716	Optional	Present
FARE	0.8765	Present	Present

Table 5.6: Our BN-based dialog model is enhanced with the capability of inheritance for the missing concepts.

We inherit discourse context for all the Class D queries.[1] This inheritance scheme may be to aggressive under certain situations. Based on the training data, we have designed three context "refresh rules" to "undo" context inheritance for several query types. The rules and the examples (the queries are in sequential order within the same dialog session) are described below:

- **Rule 1.** Disinherit previous context for Class D Airline_Name queries. If the goal of the Class D query is Airline_Name, it is usually asking about the full name of a specific airline code, (a typical example is "*what does d l mean*", where *d l* stands for Delta Airlines.) This query type is usually a context-independent query and only the key concept <AIRLINE_CODE> is required and the rest of the keys from previous discourse are redundant and will not be inherited. Table 5.7 shows an example of "undoing" the context for the Class D Airline_Name query.

Query 1	:	"*from milwaukee to atlanta before ten a m daily*" (Class A)
Inferred Goal	:	Flight_ID
Case Frame	:	CITY_ORIGIN = '*milwaukee*' CITY_DESTINATION = '*atlanta*' DEPARTURE_TIME = '*ten a m*'
Query 2	:	"*what does y x*[2]*mean*" (Class D)
Inferred Goal	:	Airline_Name (This query is asking for the full name for the airline code '*y x*', hence the concepts from previous query, e.g. CITY_NAME, DEPARTURE_TIME are not inherited.)
Case Frame	:	AIRLINE_CODE = '*y x*'

Table 5.7: Previous context will not be inherited for the Class D Airline_Name queries (Query 2 in this table).

[1] We assume we know which queries are Class A and D respectively.
[2] *y x* stands for Midwest Express Airlines.

- **Rule 2.** Class D Airline_Code queries disinherit the concepts <AIRLINE_NAME> or <AIRLINE_CODE> from the previous discourse.

  If the goal of the Class D query is Airline_Code, it is obviously asking about an airline for a specific flight (the typical example is "*which airlines have first class flights today*"), hence the concepts <AIRLINE_NAME> or <AIRLINE_CODE> from the previous discourse will not be inherited.

  A sequence of example queries is illustrated in Table 5.8.

Query 1	:	"*show me all flights from san jose to phoenix*" (Class A)
Inferred Goal	:	Flight_ID
Case Frame	:	CITY_ORIGIN = '*san jose*' CITY_DESTINATION = '*phoenix*'
Query 2	:	"*what airline is h p*[3]" (Class D)
Inferred Goal	:	Airline_Name
Case Frame	:	AIRLINE_CODE = '*h p*'
Query 3	:	"*what airline has flight four four six*" (Class D)
Inferred Goal	:	Airline_Code (This query is asking for an airline, so AIRLINE_CODE = '*h p*' from Query 2 is not inherited.)
Case Frame	:	CITY_ORIGIN = '*san jose*' CITY_DESTINATION = '*phoenix*' FLIGHT_NUMBER = 446

Table 5.8: Concepts of <AIRLINE_CODE> or <AIRLINE_NAME> will not be inherited for the Class D Airline_Code query (Query 3 in this table).

[3] *h p* stands for America West Airlines.

- **Rule 3.** Class D Flight_ID or Fare_ID queries disinherit the concept <CODE_NAME>.

  If the goal of the Class D query is Flight_ID or Fare_ID, that is, the user is asking for a flight or a fare, the concept <CODE_NAME> (which refers to the fare code or restriction code) will not be inherited. Table 5.9 illustrates an example of this rule.

Query 1	:	"*one way ticket from milwaukee to orlando either wednesday evening or thursday morning*" (Class A)
Inferred Goal	:	Flight_ID
Case Frame	:	CITY_ORIGIN = '*milwaukee*' CITY_DESTINATION = '*orlando*' PERIOD = '*wednesday evening*' PERIOD = '*thursday evening*'
Query 2	:	"*what does fare code y mean*" (Class A)
Inferred Goal	:	Fare_Basis_Code
Case Frame	:	CODE_NAME = '*y*'
Query 3	:	"*now show me all flights leaving before eight a m*" (Class D)
Inferred Goal	:	Flight_ID (This query is asking for flight information and CODE_NAME = '*y*' from Query 2 which refers to fare code is not inherited.)
Case Frame	:	CITY_ORIGIN = '*milwaukee*' CITY_DESTINATION = '*orlando*' DEPARTURE_TIME = 800

Table 5.9: Concept <CODE_NAME> is irrelevant for the Flight_ID or Fare_ID queries and is not inherited (Query 3 in this table).

5.3 Portability of the BN-based Dialog Model

In addition to scalability, this work also includes a preliminary examination of the portability of the BN-based framework across different application domains. Migration to a new application often implies the lack of domain-specific data to train the BN probabilities. Under such circumstances, the BN probabilities can be hand-assigned to reflect the "degree of belief" of the knowledge domain expert. We have designed some general principles for probability assignment, as will be presented in the following subsections. Since the ATIS corpus only provides reference semantic frames but not mixed-initiative interactions, our evaluation for portability focuses on the ability of the ported BNs to correctly identify goals and concepts in the user's query. We assume that BN probabilities that achieve good performance in goal and concept identification should have captured domain-specific constraints well. Such constraints are critical for automatic detection of missing or spurious concepts in order to drive mixed-initiative dialog modeling.

5.3.1 General Principles for Probability Assignment

Under the condition that there is little or no training data, we do not have a statistical basis for selecting the relevant concepts for each BN. (Recall that previously we have used the Information Gain criterion for this purpose). Hence we begin by identifying the concepts that are directly relevant to each goal according to human judgement. Doing so for all the 11 goals in ATIS extracts a set of 23 concepts in total. Among these, 13 are semantic concepts that correspond to SQL attributes for database access, e.g. <AIRPORT_NAME>, <AIRLINE_NAME>, <TRANSPORT_TYPE>;

and the remaining ones are keyword-based concepts, e.g. <AIRCRAFT>, <FARE>,<FROM>).[4] For the sake of simplicity, we assumed independence among concepts in the BN (i.e. adopt the pre-defined topology), and develop 11 BNs with 23 concepts each. We then hand-assigned the four probabilities for each BN, namely $P(C_j|G_i)$, $P(\overline{C_j}|G_i)$, $P(C_j|\overline{G_i})$, $P(\overline{C_j}|\overline{G_i})$. We avoid assigning the probabilities of 1 or 0 since they are not supportive of probabilistic inference. In the following we describe the general principles for assigning $P(C_j|G_i)$ and $P(C_j|\overline{G_i})$. The remaining $P(\overline{C_j}|G_i)$ and $P(\overline{C_j}|\overline{G_i})$ can be derived by the complement of the former two probabilities.

5.3.1.1 Probability Assignment for $P(C_j|G_i)$

Table 5.10 displays the guidelines by which we assign values to the probabilities $P(C_j|G_i)$. The assignment is based on human judgement of possible occurrence frequency of a concept C_j in queries of goal G_i.

- **Case 1. C_j must occur given G_i**

  If we identify a concept C_j to be mandatory for a query of goal G_i, we will hand-assign a high probability roughly from 0.95 to 0.99 for $P(C_j|G_i)$. For example, this applies to the concept <FARE> (corresponding to the words e.g. *fare*, *price*, etc.) which must occur in a Fare_ID query. (e.g. "*what is the first class **fare** from detroit to las vegas*" and "*show me the first class and coach **price***").

[4] The 23 semantic concepts for each handcrafted BN are listed in Appendix D.

- **Case 2. C_j often occurs given G_i**

  If the concept often occurs with the G_i query, then we will lower the assigned value of $P(C_j|G_i)$ to the range 0.7 to 0.8. For example, a Fare_ID query often occurs with the concepts <CITY_ORIGIN> and <CITY_DESTINATION>.

- **Case 3. C_j may occur given G_i**

  This applies to the concepts that act as additional constraints for database access. Examples are <TIME_VALUE>, <DAY_NAME>, <PERIOD> specified in the user query, the assigned values for $P(C_j|G_i)$ range between 0.4 and 0.6.

- **Case 4. C_j seldom occurs given G_i**

  This is the case where the occurrence of concept C_j in queries with goal G_i is infrequent. Examples include the concept <STOPS> which specifies a request for a nonstop flight is not usually associated with the Fare_ID queries, and so the assigned value for $P(C_j|G_i)$ ranges from 0.2 to 0.3.

- **Case 5. C_j never occurs given G_i**

  If the presence of concept C_j usually implies absence of goal G_i, then the probability of $P(C_j|G_i)$ is set to low values between 0.01 and 0.1. Examples include the concept <FLIGHT_NUMBER> in the Flight_ID query, i.e. the presence of <FLIGHT_NUMBER> in the input query implies that the goal Flight_ID is unlikely, because the aposteriori probability for the BN of Flight_ID is lowered.

Condition	**Probability of $P(C_j \vert G_i)$**
1. C_j must occur given G_i	0.95 – 0.99
2. C_j often occurs given G_i	0.7 – 0.8
3. C_j may occur given G_i	0.4 – 0.6
4. C_j seldom occurs given G_i	0.2 – 0.3
5. C_j never occurs given G_i	0.01 – 0.1

Table 5.10: Conditions for assigning the probability $P(C_j|G_i)$.

5.3.1.2 Probability Assignment for $P(C_j|\overline{G_i})$

Table 5.11 displays the guidelines by which we assign values to the probabilities $P(C_j|\overline{G_i})$). The assignment is based on human judgement of the possible occurrence frequency of a concept C_j in queries of goals other than G_i.

- **Case 1. C_j always occurs for goals other than G_i**

  Consider the relationship between the concept <CITY_ORIGIN> and the goal Aircraft_Code. Since <CITY_ORIGIN> always occurs in other informational goals, (e.g. Flight_ID, Fare_ID, etc.), we assign $P(C\text{<CITY\_ORIGIN>} = 1|\ G\text{<Aircraft\_Code>} = 0)$ in the range from 0.7 to 0.9.

- **Case 2. C_j sometimes occurs for goals other than G_i**

  Consider the relationship between the concept <CLASS> and the goal Aircraft_Code. Since <CLASS> sometimes occurs in the informational goals other than Aircraft_Code, and acts as an additional constraint for database access, we assign $P(C\text{<CLASS>} = 1|G\text{<Aircraft\_Code>} = 0)$ in the range from 0.2 to 0.5.

- **Case 3. C_j seldom occurs for goals other than G_i**

  This applies to the concepts that are strongly dependent on a specific goal and hence seldom appear for other goals. For example, the concept <TRANSPORT> usually accompanies the goal Ground_Transport only. Hence $P(C$<TRANSPORT> $= 1|$ G<Ground_Transport> $= 0)$ is set close to 0.

Condition	Probability of $P(C_j\|\overline{G_i})$
1. C_j always occurs for goals other than G_i	0.7 – 0.9
2. C_j sometimes occurs for goals other than G_i	0.2 – 0.5
3. C_j seldom occurs for goals other than G_i	0.01 – 0.1

Table 5.11: Conditions for assigning the probability $P(C_j|\overline{G_i})$.

5.3.2 Performance of the BN-based Dialog Model with Hand-Assigned Probabilities

BNs with hand-assigned probabilities achieved a goal identification accuracy of 80.9% for the ATIS-3 1993 test set (Class A and D queries included). This compares to 84.6% when they have been automatically trained on the training data. We identify the queries with a single goal based on the maximum aposteriori rule. On the other hand, the queries whose goals are not covered by our 11 BNs are treated as OOD. Queries are considered to be identified correctly if there are classified as such. The availability of training data for the BNs enhances performance in goal identification. An example ATIS dialog generated by the BN-based dialog model with hand-assigned probability is shown in Table 5.12.

System	:	*"what kind of flight information are you interested in"*
User	:	*"find me a flight that flies from memphis to tacoma"* (Class A)
(Concepts)	:	<FLIGHT> <FROM> <CITY_ORIGIN> <TO> <CITY_DESTINATION>
System	:	*Goal Inference:* Flight_ID (Correct)
User	:	*"which of these flights stop in los angeles"* (Class D)
(Concepts)	:	<FLIGHT> <STOP> <CITY_STOPOVER>
System	:	*Goal Inference:* Flight_ID (Correct, results from backward inference indicate that the concepts <CITY_ORIGIN> and <CITY_DESTINATION> are missing. These concepts are automatically retrieved from the previous discourse.)

Table 5.12: Examples of ATIS dialogs produced by the BN-based dialog model with hand-assigned probabilities.

Besides, we have compared the hand-assigned probabilities with the trained probabilities based on natural language understanding, where the evaluation metric is the sentence error rate. A sentence is considered correct only if the inferred goal and extracted concepts in the generated semantic frame agree with those in the reference semantic frame (derived from the SQL query in the ATIS corpus). The goal identification accuracies and the sentence error rates for the ATIS-3 1993 test set are summarized in Table 5.13. When we compare our results with the NL understanding results from the 10 ATIS evaluation sites [39] shown in Table 5.14, our performance falls within a reasonable range.

	Class	BNs (Handcrafted)	BNs (Trained)
Goal ID Accuracy	A (448)	90.2%	91.7%
	D (325)	68.3%	74.8%
	A + D	80.9%	84.6%
Sentence Error Rate	A (448)	12.1%	9.2%
	D (325)	40.9%	33.9%
	A + D	24.2%	19.5%

Table 5.13: Goal identification accuracies and sentence error rates of Class A and D queries of ATIS-3 1993 test data for the hand-assigned probabilities and automatically trained probabilities respectively.

Class	Sentence Error Rate	Median Value
A (448)	6.0 - 28.6%	12.8%
D (325)	13.8 - 63.1%	29.1%
A+D (773)	9.3 - 43.1%	19.5%

Table 5.14: Benchmark NL results from the 10 ATIS evaluation sites.

5.3.3 Error Analysis

The query is considered correct in our evaluation if the inferred goal together with the extracted concepts agree with all the attributes in the reference SQL query. Although we have the context "refresh rules" to "undo" context inheritance for several query types, we have observed that our strategy for context inheritance still leads to insertion errors in the extracted concepts in the generated semantic frame. This is illustrated in the example in Table 5.15.

Query 1:	"*list flights from oakland to salt lake city before six a m thursday morning*" (Class A) (Our system generates a correct semantic frame.)	
Query 2:	"*list delta flights before six a m*" (Class D) (Our system generates a correct semantic frame.)	
Query 3:	"*list all flights from twelve oh one a m until six a m*" (Class D) (Semantic frame is shown below)	
	Case Frame	**SQL Reference**
Goal:	Flight_ID	Flight_ID
Concepts:	CITY_ORIGIN: *oakland* CITY_DESTINATION: *salt lake city* DEPARTURE_TIME: *twelve oh one a m until six a m* AIRLINE_NAME: *delta (a concept insertion error)*	ORIGIN: *oakland* DESTINATION: *salt lake city* DEPARTURE_TIME: ≥ 1 & ≤ 600

Table 5.15: The semantic frame of Query 3 indicates that our context inheritance strategy may be too aggressive, resulting in a concept insertion error in the generated semantic frame.

Further inspection of the high sentence error rate of the Class D context-dependent queries (33.9% for the trained BN) reveals that the BNs which are trained from the Class A queries cannot model the Class D queries well. The degradation of goal identification performance from 91.7% (Class A) to 74.8% (Class D) is not surprising. For example, most of the Class A training queries with the goal Fare_ID are associated with the concepts of <FARE>, <CITY_ORIGIN> and <CITY_DESTINATION>, a typical example being *"what is the price of flights from indianapolis to memphis"*. However, for the Class D Fare_ID query *"what is the cost of the flight on u s air"* (Table 5.16, second query), although the city of origin and destination are inherited for goal inference, the presence of the concept <AIRLINE_NAME>, which is sparse in the Class A Fare_ID queries, will lower the probability of occurrence of the goal Fare_ID. This query is eventually outweighed by the goal Flight_ID, which is thus inferred wrongly.

Query (Class A)	: *"show flights from cleveland to miami that arrive before four p m"*
Concepts	: <FLIGHT> <FROM> <CITY_ORGIN> <TO> <CITY_DESTINATION> <TO> <PREP> <TIME_VALUE>
Inferred Goal	: Flight_ID (Correct)

Query (Class D)	: *"what is the cost of the flight on u s air"*
Concepts	: <WHAT> <FARE> <FLIGHT> <AIRLINE_NAME>
Inferred Goal	: Flight_ID (Wrong, the correct goal is Fare_ID)

Table 5.16: This table provides an example where the Class D query is inferred wrongly because the BN trained from the Class A queries cannot model the Class D queries well.

5.4 Enhancements for Discourse Query Understanding

From the previous section, we have identified two problems in our BN-based dialog model. First, our framework is not robust on the Class D context-dependent queries. Since the probabilities of the BNs are tailor-made for Class A queries only, it is not surprising that there is a degradation on the goal identification accuracies from Class A (91.7%) to Class D (74.8%) queries. Second, our model cannot capture some of the equivalence classes. Our experiments show that in the complex ATIS domain with more semantic concepts, some concepts are sparsely trained. Some concepts need to be present, others should be absent, but still others should be optional. We previously proposed the two levels of thresholding (0.7 and 0.2) in the backward inferencing to decide if a concept be present, optional or absent in the query. However, this prototype needed further improvement as it is infeasible to determine the two new thresholds when we migrate to other domains. It is deemed desirable to use a *single* pre-set threshold ($\theta = 0.5$) for backward inference in all the application domains. In order to tackle these two problems, we propose two measures as follows:

5.4.1 Combining Trained and Handcrafted Probabilities

Since the BNs were trained on Class A queries only, they cannot fully capture the concept dependencies in the Class D queries. By analyzing the errors in the Class D training queries, we attempt to modify the BN probabilities.

For instance, a number of Class D Fare_ID queries that contain the concept <AIRLINE_NAME> were handled incorrectly by the BNs. Investigation shows that:

$$P(C\text{<AIRLINE\_NAME>} = 1|G\text{<Fare\_ID>} = 1) = 0.19$$

The value is low because there are few occurrences of this concept together with the goal in the Class A training queries. With reference to the guidelines presented in Table 5.10, the concept <AIRLINE_NAME> may occur for Fare_ID queries. Hence we revised the probability $P(C_j|G_i)$ from 0.19 to 0.4. By incorporating some hand-assigned probabilities to the trained BNs, we hope that our framework is able to capture the concept dependencies for Class D queries.

5.4.2 Handcrafted Topology for BNs

The BN topologies learnt from training data may not be able to capture some of the equivalence classes due to the lack of data. As we have discussed in Section 5.2.2, the learnt topology of the BN for Aircraft_Code only captures the city of origin and destination for the specification for database access. Therefore, the Aircraft_Code query (in Table 5.17) specifying the airline and the flight number will obtain several missing / spurious concepts when the single threshold scheme for backward inference is used. A pair of cities will be prompted for missing while the flight number is clarified as spurious. However, the flight number together with the airline serve equally well for database access. There is a causal dependence between the equivalence classes, which is an OR relationship:

(CITY_ORIGIN and CITY_DESTINATION) or (FLIGHT_NUMBER and AIRLINE_NAME)

The occurrence of the second attribute pair (FLIGHT_NUMBER and AIRLINE_NAME) is relatively sparse in the training corpus, and the topology of our BN is unable to reflect the equivalence classes. The learnt topology of the original BN for Aircraft_Code in Figure 5.1 shows that there is no linkage between the two attribute pairs, an indication that they are independent with each other.

Query	: *"what type of aircraft is used in american airlines flight number seventeen twenty three"*
Concepts	: <WHAT> <TYPE> <AIRCRAFT> <AIRLINE_NAME> <FLIGHT_NUMBER>
Inferred Goal	: Aircraft_Code (Correct)

Table 5.17: An Aircraft_Code query that specifies the airline together with the flight number.

We attempt to refine the BN topologies by adding linkage(s) that are obvious but have not been learnt due to training data sparseness. For example, for the Aircraft_Code BN, we notice that there should be dependence between the two attribute pairs: (<CITY_ORIGIN>, <CITY_DESTINATION>) and (<AIRLINE_NAME>, <FLIGHT_NUMBER>) and we add a link between them. The refined topology is depicted in Figure 5.2. From the new topology, the occurrence of the concept <FLIGHT_NUMBER> not only depends on <AIRLINE_NAME>, but also the pair of cities.

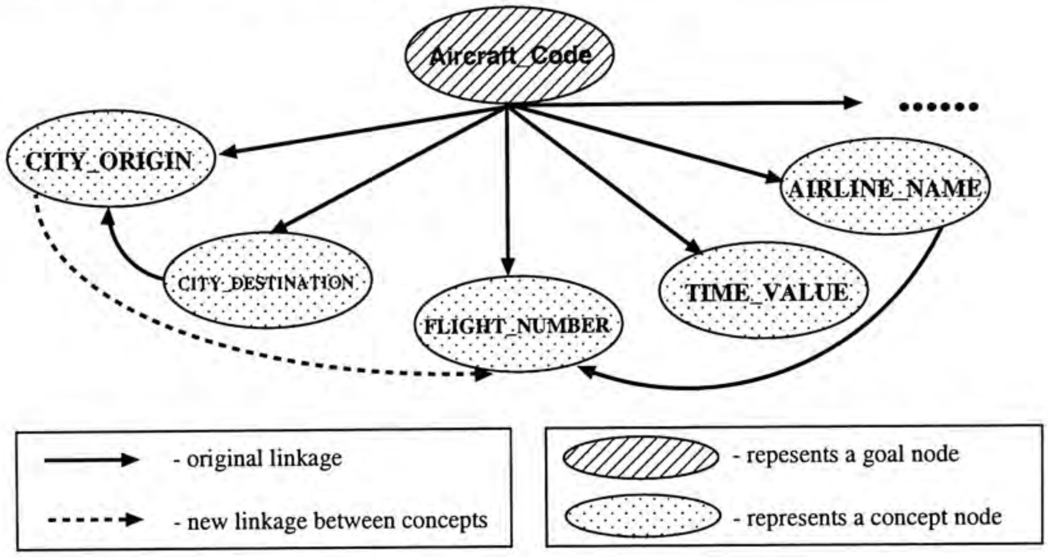

Figure 5.2: The new topology of the BN for the informational goal Aircraft_Code.

5.4.2.1 New Mechanism for Backward Inference

The probabilities of the BN with the new topology can be obtained from the training data. Then the BN is ready for goal inference and detection of the missing / spurious concepts by backward inference. Using the example Aircraft_Code query in Table 5.17, our BN for Aircraft_Code performed backward inference. The original scheme for backward inference simply involves the instantiation of the corresponding goal node. However, in order to consider the dependence between the concepts, we adopt a new scheme for backward inference as shown in Figure 5.3. The new backward inference results using a single threshold ($\theta = 0.5$) are tabulated in Table 5.18.

The results indicate that the query passes the domain-specific constraints and the query is ready for database retrieval. Notice that our previous learnt BN using the original backward inference scheme with a single threshold indicated that the concepts <CITY_ORIGIN> and <CITY_DESTINATION>

For the BN of inferred goal G_i with concepts C_j :

1 Instantiate the goal node to 1

2 For each concept (C_k) of goal G_i (where $C_k \neq C_j$)

Identify its parent(s) C_k (i.e. there is a link going from C_k to C_j).
Instantiate the node for C_k node according to its occurrence in the user's query.

3 Obtain the updated probability for $P^*(C_j)$

Figure 5.3: New mechanism for backward inference.

Concept$_j$ (C_j) **(subset)**	$P^*(C_j)$	**Binary Decision for** C_j	**Actual Occurrence for** C_j
AIRCRAFT	1.000	Present	Present
AIRLINE_NAME	0.583	Present	Present
CITY_ORIGIN	0.000	Absent	Absent
CITY_DESTINATION	0.000	Absent	Absent
TIME_VALUE	0.201	Absent	Absent
FLIGHT_NUMBER	0.600	Present	Present

Table 5.18: Updated probabilities obtained from using new mechanism of backward inferencing (BN of Aircraft_Code) using 0.5 as threshold for the query "*what type of aircraft is used in american airlines flight number seventeen twenty three*".

are missing, while <FLIGHT_NUMBER> is spurious for this query (refer to Table 5.3).

To further illustrate the behavior of the hand-refined BN topology, consider another Aircraft_Code query which specifies the city pair as shown in Table 5.19. Our BN for Aircraft_Code performed backward inference using the new mechanism and the results using the single threshold scheme are also tabulated in the same table. It is found that the occurrence of the attribute pair (<CITY_ORIGIN>, <CITY_DESTINATION>) will automatically lower the probabilities of the <AIRLINE_NAME> and <FLIGHT_NUMBER>. Our new BN for Aircraft_Code can thus capture more sophisticated concept dependence by incorporating handcrafted linkage.

However, some of the optional concepts which are sparse in the training data are still classified as spurious when we use a single threshold in the backward inference. From the previous example query with the backward inference results shown in Table 5.19, the concept <TIME_VALUE>, which is an additional specification for database access, is classified as spurious. In order to clarify the truly spurious concepts, we will identify a set of "optional" concepts for each goal described in the following section.

Query	: *"display types of aircraft departing from cleveland to dallas before noon"*
Concepts	: <TYPE> <AIRCRAFT> <FROM> <CITY_ORIGIN> <TO> <CITY_DESTINATION> <PREP> <TIME_VALUE>
Inferred Goal	: Aircraft_Code (Correct)

Concept$_j$ (C_j) **(subset)**	$P^*(C_j)$	**Binary Decision for** C_j	**Actual Occurrence for** C_j
AIRCRAFT	1.000	Present	Present
AIRLINE_NAME	0.000	Absent	Absent
CITY_ORIGIN	1.000	Present	Present
CITY_DESTINATION	1.000	Present	Present
TIME_VALUE	0.201	Absent	Present
FLIGHT_NUMBER	0.000	Absent	Absent

Table 5.19: This table shows an Aircraft_Code query specified with a city pair and the updated probabilities obtained from the new mechanism of backward inferencing with 0.5 as the threshold for this query.

5.4.2.2 Identification of the Optional Concepts

We have a total of 60 hand-designed semantic concepts in the ATIS domain. In order to identify a set of optional semantic concepts for each goal, the frequency of each concept associated with the corresponding goal is obtained from the training set. Concepts whose frequency exceeds 0 and scant the threshold θ_{G_i} will be classified as optional, where the threshold θ_{G_i} is half of the number of the G_i queries, i.e. $\theta_{G_i} = \left\lceil \frac{Freq(Gi)}{2} \right\rceil$.

For example, the frequency of the semantic concepts for the goal Aircraft_Code are listed in Table 5.20. Since the training queries for Aircraft_Code were instantiated 13 times, the threshold θ_{G_i} is 7. We can then determine the optional concepts for the BN of Aircraft_Code (shaded part in Table 5.20). As a result, optional concepts will not be considered spurious during backward inference, and the BN-baased dialog model will not generate a clarification response for the concepts.

5.4.3 Performance of the Enhanced BN-based Dialog Model

We have modified the probabilities of the trained BN so that our framework can model the Class D queries as well. Besides, in order to capture the equivalence classes that are sparsely trained, we will handcraft the topology for the BN (insert linkage by hand). By adopting the new mechanism for the backward inference scheme and identifying a set of optional concepts associated with each goal, we can use a single pre-set threshold ($\theta = 0.5$) to identify the presence or absence of the concepts. Robustness of our BN-based dialog model is hence achieved, and the performance of our BN-based

Concepts	Frequency
<AIRCRAFT>	13
<PREP>	10
<TO>	8
<FROM>	8
<CITY_ORIGIN>	8
<CITY_DESTINATION>	8
<AIRLINE_NAME>	7
<KIND>	6
<FLIGHT_NUMBER>	4
<WHAT>	5
<SUPERLATIVE>	2
<TIME_VALUE>	2
<MONTH>	2
<FLIGHT>	2
<DAY>	2
<PRE_TIME>	1
<PERIOD>	1
<DAY_NAME>	1

Table 5.20: Frequency of concepts associated with Aircraft_Code from the training set. The shaded concepts whose frequency is below the threshold ($\theta_G = 7$) are identified as optional concepts.

dialog model is improved. The goal identification accuracies and the sentence error rates for the ATIS-3 1993 test set are summarized in Table 5.21. When we compare the new results (with the enhancement of incorporating the handcrafted probabilities into the trained BN, using the handcrafted topology with the new backward inference scheme) to the one using the trained BNs only (for the convenience of reading, the fourth column of Table 5.13 is duplicated to be the rightmost column of Table 5.21), the goal identification accuracy for the Class D queries is increased by 8.6% (absolute) while the sentence error rate is decreased by 7.0% (absolute).

	Class	BNs (Enhanced)	BNs (Original)
Goal ID Accuracy	A (448)	91.7%	91.7%
	D (325)	83.4%	74.8%
	A + D	88.2%	84.6%
Sentence Error Rate	A (448)	9.2%	9.2%
	D (325)	26.8%	33.9%
	A + D	16.6%	19.5%

Table 5.21: Goal identification accuracies and sentence error rates for the ATIS-3 1993 test data (especially the Class D queries) are improved when we have incorporated into our framework hand-assigned probabilities and a handcrafted topology.

5.5 Chapter Summary

This chapter describes the scalability and portability of the BN-based dialog model as we migrate from a simpler foreign exchange domain (CU FOREX) to the relatively more complex air travel domain (ATIS). The adapted processes include: (i) automatic selection of specified concepts in the user's query, for the purpose of informational goal inference; (ii) automatic detection of missing / spurious concepts based on backward inference using the BN. Two levels of thresholding are adopted in backward inference to decide which concepts are present, absent or optional in the given query. The dialog model can thus identify the truly missing / spurious concepts. We have also enhanced our dialog model with the capability of discourse context inheritance.

To ease portability across domains, which often implies the lack of training data for the new domain, we have developed a set of principles for hand-assigning BN probabilities, based on the "degree of belief" in the relationships between concepts and goals. Since the ATIS data only provides reference semantic interpretations, we are constrained to evaluate the proposed BN framework based on goal identification performance and sentence error rates respectively. It is found that application of our model to the ATIS data gave promising results. However, our BNs which are trained from the Class A queries, cannot model the Class D queries well. The performance of our dialog model is further improved if the hand-assigned probabilities are incorporated with the trained BNs. Furthermore, in order to improve the robustness of our BN-based dialog model in which a single pre-set threshold is adopted for backward inference in all application domains, we have incorporated prior knowledge in the development of the BN topology. Linkages between concepts are handcrafted to reflect the equivalence classes. Also, optional concepts for each goal are identified so that they are not deemed as spurious from backward inference.

Chapter 6

Conclusions

6.1 Summary

In this thesis, we have presented a methodology of using Belief Networks (BNs) for natural language understanding and dialog modeling in human-computer conversational systems. The BNs model the causal relationships between the semantic concepts in the user's query and its underlying informational goal. They are used to infer the users' intentions in their information-seeking queries. By formulating our N-way classification problem as N binary classifications, each performed by a BN, our framework has the capability of (i) identifying queries with multiple informational goals, and (ii) rejecting queries whose goals are outside of the prescribed knowledge domain.

The topology of our BNs can also be automatically learned to capture more sophisticated domain-specific constraints so as to improve the understanding accuracy. We have demonstrated that our BN-based framework in natural language understanding is robust against speech recognition errors. The user's input is validated against the domain constraints by using backward inference of the BNs. The spurious concepts which may be caused by speech recognition errors can be rejected automatically. Our BN-based framework can be further enhanced to handle dialog modeling. The detection of missing / spurious concepts from backward inference governs the model transitions for mixed-initiative dialog modeling.

We tested the BN-based dialog model in the context of the CU FOREX domain. The CU FOREX system is a bilingual hotline for real-time foreign exchange information. It presently supports two separate interaction modalities: a direct dialog (DD) interaction, which is system-initiative for novice users; as well as natural language shortcut (NLS), which is user-initiative for expert users. While the BNs are used to infer the informational goals of the user's query, we also attempt to verify the validity of the input query against the domain-specific constraints by using backward inference. The BNs can thus detect missing concepts as well as spurious concepts, invoke the dialog model to prompt for missing information, and ask for clarification. As a result, the BN-based dialog model can integrate the two interaction modalities to achieve mixed-initiative interaction. Furthermore, we have enhanced our dialog model with the capability of incorporating OOV words.

We have also investigated the scalability and portability of the BN-based dialog model as we migrated from the simple foreign exchange domain (CU FOREX) to the relatively more complex air travel domain (ATIS). The complexity of an application domain is characterized in part by the number of in-domain informational goals and concepts. The presence / absence of concepts are used to infer the presence / absence of each goal, by means of the corresponding BN. When a large number of in-domain concepts are available (especially for complex domains), we used an information-theoretic criterion (Information Gain) to automatically select the small set of concepts most indicative of a goal, and do so for every in-domain goal. Automatic detection of missing / spurious concepts is achieved by backward inference using the BN corresponding to the inferred goal. For the simple CU FOREX domain, detection of missing / spurious concepts was based on a single probability threshold. However, scaling up to ATIS (which has many more concepts) shows that some concepts need to be present, others should be absent, but still others should be optional. Hence we need to use two levels of thresholding to decide if a concept should be present, optional or absent in the query. We have also enhanced our BN-based dialog model with the capability of context-inheritance, in order to handle the context-dependent user queries in the ATIS domain. Discourse context is inherited for the Class D queries, and we invoke goal inference again after context inheritance if a query was previously classified as OOD.

As regards portability, migration to a new application domain often implies the lack of domain-specific training data. Hence we have proposed a set of general principles for probability assignment to the BNs, as a reflection of our "degree of belief" in the relationships between concepts and goals. We compared the goal identification performances, as well as sentence error rates between the use of hand-assigned probabilities, and the probabilities trained from the ATIS training set. Results show that the hand-assigned probabilities offer a decent starting performance to ease portability to a new domain. The system performance is improved if user data is available to train the probabilities.

Since our BNs are trained from the Class A context-independent queries, they cannot model the Class D context-dependent queries well. This results in high sentence error rate in the Class D queries. Therefore, we incorporate the trained BN with handcrafted probabilities to model the domain-specific constraints for Class D queries as well. Besides, the learnt topology of the BNs are not able to capture the sparsely trained concept dependencies as well as some optional concepts. Therefore, the two levels of thresholding previously adopted in the ATIS domain determined which concepts should be present, absent or optional in the given query. However, it is desirable to replace the double threshold detection scheme with a single threshold scheme, which to avoid setting ad hoc threshold value. In order to capture the concept dependencies which are lacking in the training queries, we handcrafted the linkages for the BNs. A set of optional concepts for each informational goal is also identified so that the dialog model will not clarify them as spurious. Robustness of our BN-based dialog model is achieved and the system performance has been further improved when we adopted this new approach.

6.2 Contributions

This work is one of the first attempts in applying Belief Networks to the problem of natural language understanding and dialog modeling. Our BN-based framework has a number of innovative features which are contributive to the field. They are listed as follows:

1. BNs are used to model the causal relationships between the user's informational goal and the semantic concepts provided in his / her query. Hence our BN-based framework is capable of natural language understanding. Our approach is capable of *rejecting* out-of-domain queries that lie outside of the system's scope of competence and is *robust* against *speech recognition errors.*

2. The BNs are *automatically trainable* and their topologies are *automatically learnable* from training data. This *reduces handcrafting* involved in the design of conventional natural language understanding systems.

3. We have shown how our BN-based framework can be extended for *mixed-initiative dialog modeling.* In enforcing domain-specific constraints, our framework is capable of automatically detecting missing or spurious concepts. This is in turn used to drive the mixed-initiative dialog model to prompt for missing concepts and clarify spurious concepts.

4. In order to enhance *portability* of our BN-based framework across application domains, we have developed a set of principles for hand-assigning probabilities for the BN. This ameliorates the reliance on the availability of large amounts of annotated training data.

5. We have also included an initial attempt to demonstrate the *scalability* of our BN-based framework in modeling increasingly complex domains, i.e. migrating from the foreign exchange domain to the air travel domain. As the complexity of an application domain increases, the domain-specific constraints may involve mandatory concept sets, optional concept sets and concepts sets that should be absent. We have proposed a possible method to model such phenomena by means of manual enhancements of BN topologies. The approach seems promising but further investigation will be needed.

6.3 Future Work

Possible extensions of this work include:

1. Incorporating the confidence scores of the spoken queries from the speech recognizer in the BN-based dialog model. We can then utilize the scores with the binary decision of the BNs to further detect the errors that may lead to misunderstanding. More appropriate responses such as rejection, implicit / explicit confirmation can also be included. Besides, the dialog model can adapt to be more system-initiative if it detects that the user has difficulty in achieving his / her goal.

2. Modeling the informational goals that are in-domain but not statistically significant. For the ATIS-3 domain, there are a total of 32 query types but we only model 11 goals to avoid sparse data. In order to model all query types, we can inject prior knowledge to reflect "degree of belief" for the network probabilities.

3. Modeling the concept dependencies of the BNs manually. As discussed in Section 5.3, when we port the BN-based dialog model to a new domain with insufficient training data, we can hand-assign probabilities for the BNs to reflect the "degree of belief" and we assume independence among concepts in the BNs for simplicity. However, we can further improve the performance of the BN-based dialog model by inserting linkage(s) between the concepts in the BNs. Since we do not have any training data to estimate the inter-concept dependencies, we can randomly assign values for the BN probabilities and refine their values when we get the real data.

4. Determining context inheritance by BNs. As mentioned in Section 5.2.3, we inherit discourse context for all the Class D queries and adopt some "refresh rules" to undo context inheritance for several query types. It is seem that the exploration of including Class D queries in training the BNs may be useful. The BNs can hence capture the causal relationships between the Class A and Class D queries and determine which attributes should be inherited.

5. Integrating external factors into the BN-based dialog model to drive the dialog flow. Most current E-commerce related applications require authentication of user identity, e.g. by means of speaker verification technology. If the user passes the verification, it will drive the system one way. If not, it will drive the system another way. Our BN-based dialog model is also extensible to include an external engine to affect the dialog flow.

Bibliography

[1] K. Arai, J. Wright, G. Riccardi, and A. Gorin. "Grammar Fragment Acquisition Using Syntactic and Semantic Clustering". In *Proceedings of the 5th International Conference on Spoken Language Processing*, 1998.

[2] R. Belvin, R. Burns, and C. Hein. "Spoken Language Navigation Systems for Drivers". In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[3] S. Bennacef, L. Devillers, S. Rosset, and L. Lamel. "Dialog in the RAILTEL Telephone-Based System". In *Proceedings of the 4th International Conference on Speech and Language Processing*, pages 550–553, October 1996.

[4] M. Blomberg, R. Carlson, K. Elenius, B. Granstrom, J. Gustafson, S. Hunnicutt, R. Lindell, and L. Neovius. "An Experimental Dialogue System: WAXHOLM". In *Proceedings of the 3th European Conference on Speech Communication and Technology*, 1993.

[5] B. Carpenter and J. Chu-Carroll. "Natural Language Call Routing: A Robust, Self-Organizing Approach". In *Proceedings of the 5th International Conference on Spoken Language Processing*, 1998.

[6] E. Chang. "Improving Rejection with Semantic Slot-based Confidence Scores". In *Proceedings of the 6th European Conference on Speech Communication and Technology*, 1999.

[7] J. Chu-Carroll. "Form-based Reasoning for Mixed-Initiative Dialogue

Management in Information-Query Systems". In *Proceedings of the 6th European Conference on Speech Communication and Technology*, 1999.

[8] K. Church and P. Hanks. "Word Association Norms, Mutual Information, and Lexicography". In *Computational Linguistics, Vol. 16, No. 1*, pages 22–29, 1990.

[9] K. Davies, R. Donovan, M. Epstein, M. Franz, A. Ittycheriah, E. E. Jan, J. M. LeRoux, D. Lubensky, C. Neti, M. Padmanabhan, K. Papineni, S. Roukos, A. Sakrajda, J. Sorensen, B. Tydlitat, and T. Ward. "The IBM Conversational Telephony System for Financial Applications". In *Proceedings of the 6th European Conference on Speech Communication and Technology*, 1999.

[10] R. Fano. *Transmission of Information.* MIT Press, 1961.

[11] J. Glass. "Challenges for Spoken Dialogue Systems". In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, December 1999.

[12] D. Goddeau, H. Meng, J. Polifroni, S. Seneff, and S. Busayapongchai. "A Form-Based Dialogue Manager for Spoken Language Applications". In *Proceedings of the 4th International Conference on Spoken Language Processing*, pages 542–545, 1996.

[13] D. Goddeau and J. Pineau. "Fast Reinforcement Learning of Dialog Strategies". In *Proceedings of the International Conference of Acoustics, Speech and Signal Processing*, 2000.

[14] A. Gorin, G. Riccardi, and J. Wright. "How May I Help You?". In *Speech Communication, Vol. 23*, pages 113–127, 1997.

[15] T. Hazen, T. Burianek, J. Polifroni, and S. Seneff. "Integrating Recognition Confidence Scoring with Language Understanding and Dialogue Modeling". In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[16] D. Heckerman and E. Horvitz. "Inferring Informational Goals from Free-Text Queries: A Bayesian Approach". In *Proceedings of the 14th Conference on Uncertainty in Aritificial Intelligence*, pages 230–238, 1998.

[17] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse. "The Lumière Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users". In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 256–265, 1998.

[18] E. Horvitz and T. Paek. "A Computational Architecture for Conversation". In *Proceedings of the 7th International Conference on User Modeling*, pages 201–210, 1999.

[19] C. Huang, P. Wu, X. Zhang, S. Zhao, T. Hung, and B. Xu. "LODESTAR: A Mandarin Spoken Dialogue System for Travel Information Retrieval". In *Proceedings of the 6th European Conference on Speech Communication and Technology*, 1999.

[20] V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, 1996.

[21] D. Jurafsky, C. Wooters, G. Tajchman, J. Segal, A. Stolcke, E. Fosler, and N. Morgan. "The Berkeley Restaurant Project". In *Proceedings of the 3rd International Conference on Spoken Language Processing*, pages 2139–2142, 1994.

[22] T. Kawahara, N. Kitaoka, and S. Doshita. "Concept-based Phrase Spotting Approach For Spontaneous Speech Understanding". In *Proceedings of International Conference of Acoustics, Speech and Signal Processing*, pages 557–560, 1996.

[23] K. Kita, Y. Fukui, M. Nagata, and T. Morimoto. "Automatic Acquisition of Probabilistic Dialogue Models". In *Proceedings of the 4th International Conference on Spoken Language Processing*, pages 196–199, 1996.

[24] R. Kuhn and R. Mori. "The Application of Semantic Classification Trees to Natural Language Understanding". In *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 17, No. 5*, pages 449–460, 1995.

[25] L. Lamel, S. Bennacef, J. Gauvain, H. Dartigues, and J. Temem. "User Evaluation of the MASK Kiosk". In *Proceedings of the 5th International Conference on Spoken Language Processing*, 1998.

[26] I. Langkilde, M. Walker, J. Wright, A. Gorin, and D. Litman. "Automatic Prediction of Problematic Human-Computer Dialogue In 'How May I Help You?'". In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 369–376, December 1999.

[27] E. Levin, S. Narayanan, R. Pieraccini, K. Biatov, E. Bocchieri, G. D. Fabbrizio, W. Eckert, S. Lee, A. Pokrovsky, M. Rahim, P. Ruscitti, and M. Walker. "The AT&T-DARPA Communicator Mixed-Initiative Spoken Dialog System". In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[28] E. Levin, R. Pieraccini, and W. Eckert. "A Stochastic Model of Human-Machine Interaction for Learning Dialogue Strategies". In *IEEE Transactions on Speech and Audio Processing, Vol. 8, Issue 1*, pages 11–23, January 2000.

[29] E. Levin, R. Pieraccini, W. Eckert, P. D. Fabbrizio, and S. Narayanan. "Spoken Language Dialogue: From Theory to Practice". In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, 1999.

[30] D. Litman, S. Singh, M. Kearns, and M. Walker. "NJFun: A Reinforcement Learning Spoken Dialogue System". In *Proceedings of the Satellite Dialogue Workshop, ANLP-NAACL*, April 2000.

[31] H. Meng, S. Busayapongchai, J. Glass, D. Goddeau, L. Hetherington, E. Hurley, C. Pao, J. Polifronri, S. Seneff, and V. Zue. "WHEELS: A Conversational System in the Automobile Classifieds Domain". In

Proceedings of the 4th International Conference on Spoken Language Processing, pages 542–545, 1996.

[32] H. Meng, S. F. Chan, Y. F. Wong, T. Y. Fung, W. C. Tsui, T. H. Lo, C. C. Chan, K. Chen, L. Wang, T. Y. Wu, X. Li, T. Lee, W. N. Choi, Y. W. Wong, P. C. Ching, and H. S. Chi. "ISIS: A Multilingual Spoken Dialog System Developed with CORBA and KQML Agents". In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[33] H. Meng, W. Lam, and K. F. Low. "Learning Belief Networks for Language Understanding". In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, December 1999.

[34] H. Meng, S. Lee, and C. Wai. "CU FOREX: A Bilingual Spoken Dialog System for the Foreign Exchange Domain". In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, 2000.

[35] S. Miller, R. Bobrow, R. Schwartz, and R. Ingria. "Statistical Language Processing Using Hidden Understanding Models". In *Proceedings of the ARPA Human Language Technology Workshop*, pages 278–282, 1994.

[36] W. Minker. "Stochastically-Based Natural Language Understanding Across Tasks and Languages". In *Proceedings of the 5th European Conference on Speech Communication and Technology*, pages 1423–1426, 1997.

[37] W. Minker, S. Bennacef, and J. L. Gauvain. "A Stochastic Case Frame Approach for Natural Language Understanding". In *Proceedings of 4th International Conference on Spoken Language Processing*, 1996.

[38] R. E. Neapolitan. *Probabilistic Reasoning in Expert Systems: Theory and Algorithms.* Wiley, 1988.

[39] D. Pallet, J. Fiscus, W. Fisher, J. Garofolo, B. Lund, and M. Przybocki. "1993 Benchmark Tests for the ARPA Spoken Language Program". In

Proceedings of the Spoken Language Technology Workshop, pages 15–40, 1994.

[40] D. Pallet, J. Fiscus, W. Fisher, J. Garofolo, B. Lund, and M. Przybocki. "1994 Benchmark Tests for the ARPA Spoken Language Program". In *Proceedings of the Spoken Language Technology Workshop*, pages 5–38, 1995.

[41] C. Pao, P. Schmid, and J. Glass. "Confidence Scoring for Speech Understanding Systems". In *Proceedings of the 5th International Conference on Spoken Language Processing*, 1998.

[42] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[43] B. Pellom, W. Ward, and S. Pradhan. "The CU Communicator: An Architecture for Dialogue Systems". In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[44] R. Pieraccini, E. Levin, and W. Eckert. "AMICA: The AT&T Mixed Initiative Conversational Architecture". In *Proceedings of the 5th European Conference on Speech Communication and Technology*, 1997.

[45] R. Pieraccini, E. Tzoukermann, Z. Gorelov, J. Gauvain, E. Levin, C. Lee, and J. Wilpon. "A Speech Understanding System Based on Statistical Representation of Semantics". In *Proceedings of International Conference of Acoustics, Speech and Signal Processing*, pages I.193–196, 1992.

[46] A. Potamianos, E. Ammicht, and H. K. Kuo. "Dialogue Management in the Bell Labs Communicator System". In *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000.

[47] P. Price. "Evaluation of Spoken Language Systems: The ATIS Domain". In *Proceedings of the ARPA Human Language Technology Workshop*, pages 91–95, 1990.

[48] D. Pynadath and M. Wellman. "Accounting for Context in Plan Recognition, with Application to Traffic Monitoring". In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 427–481, 1995.

[49] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufman, 1993.

[50] S. Seneff. "Robust Parsing for Spoken Language Systems". In *Proceedings of International Conference of Acoustics, Speech and Signal Processing*, pages 189–192, 1992.

[51] S. Seneff. "TINA: A Natural Language System for Spoken Language Applications". In *Computational Linguistics, Vol. 18, No. 1*, pages 61–86, 1992.

[52] S. Seneff and J. Polifroni. "Dialogue Management in the Mercury Flight Reservation System". In *Proceedings of the Satellite Dialogue Workshop, ANLP-NAACL*, April 2000.

[53] K. C. Siu and H. Meng. "Semi-automatic Acquistion of Domain-specific Semantic Structures". In *Proceedings of the 6th European Conference on Speech Communication and Technology*, 1999.

[54] C. Skaanning, V. Jensen, U. Kjærulff, L. Parker, P. Pelletier, and L. Rostrup-Jensen. "SACSO - A Bayesian-Network Tool for Automated Diagnosis of Printing Systems". In *Proceedings of the 9th International Workshop on Principles of Diagnosis*, 1998.

[55] J. Sturm, E. D. Os, and L. Boves. "Dialogue Management in the Dutch ARISE Train Timetable Information System". In *Proceedings of the 6th European Conference on Speech Communication and Technology*, 1999.

[56] R. Sutton and A. Barto. *Reinforcement Learning — An Introduction.* MIT Press, 1998.

[57] C. J. van Rijsbergen. *Information Retrieval.* Butterworth, London, 1979.

[58] M. Walker. "An Application of Reinforcement Learning to Dialogue Strategy Selection in a Spoken Dialogue System for Email". In *Journal of Artificial Intelligence Research, Vol 12*, pages 387–416, 2000.

[59] M. Walker, J. Wright, and I. Langkilde. "Using Natural Language Processing and Discourse Features to Identify Understanding Errors in a Spoken Dialogue System". In *Proceedings of the 17th International Conference on Machine Learning*, 2000.

[60] C. Wang, J. Glass, H. Meng, J. Polifroni, S. Seneff, and V. Zue. "YINHE: A Mandarin Chinese Version of the GALAXY System". In *Proceedings of the 5th European Conference on Speech Communication and Technology*, pages 351–354, 1997.

[61] W. Ward and S. Issar. "Recent Improvements in the CMU Spoken Language Understanding System". In *Proceedings of the ARPA Human Language Technology Workshop*, pages 213–216, 1994.

[62] B. L. Zeigler and B. Mazor. "Dialog Design For a Speech-Interactive Automation System". In *Proceedings of the 4th European Conference on Speech Communication and Technology*, pages 113–116, 1995.

[63] V. Zue and J. Glass. "Conversational Interfaces: Advances and Challenges". In *Proceedings of the 5th European Conference on Speech Communication and Technology*, pages KN9–18, 1997.

[64] V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, and L. Hetherington. "JUPITER: A Telephone-Based Conversational Interface for Weather Information". In *IEEE Transactions on Speech and Audio Processing, Vol. 8, No. 1*, January 2000.

[65] V. Zue, S. Seneff, J. Polifroni, M. Phillips, C. Pao, D. Goddeau, J. Glass, and E. Brill. "PEGASUS: A Spoken Language Interface for On-Line Air Travel Planning". In *Speech Communication, Vol. 15*, pages 331–340, 1994.

[0] References without page numbers are those from CD Rom or World Wide Web.

Appendix A

The Two Original SQL Query

```
select distinct '"'||v0.flight_id||'"', '"'||v0.airline_code||'"', '"'||v1.airport_code||'"',
'"'||v2.city_code||'"', '"'||v2.city_name||'"', '"'||v3.airport_code||'"',
'"'||v4.city_code||'"', '"'||v4.city_name||'"', '"'||v5.days_code||'"',
'"'||v6.day_name||'"', v6.year, v6.month_number, v6.day_number from
flight v0, airport_service v1, city v2, airport_service v3, city v4, days v5, date_day v6
where (( v0.airline_code = 'UA' )
and ((( v0.from_airport = any v1.airport_code )
and (( v1.city_code = any v2.city_code )
and ( v2.city_name = 'DENVER' )))
and ((( v0.to_airport = any v3.airport_code )
and (( v3.city_code = any v4.city_code )
and ( v4.city_name = 'BALTIMORE' )))
and (( v0.flight_days = any v5.days_code )
and (( v5.day_name = any v6.day_name )
and ((( v6.year = 1993 )
and ( v6.month_number = 6 ))
and ( v6.day_number = 14 )))))))
```

Figure A.1: SQL query for maximum reference answer.

```
select distinct flight.flight_id from flight where
  ( flight.airline_code = "UA" and
    ( flight.from_airport in
        ( select airport_service.airport_code from airport_service where
          airport_service.city_code in
            ( select city.city_code from city where
              city.city_name = "DENVER"
            )
        ) and
        ( flight.to_airport in
            ( select airport_service.airport_code from
              airport_service where airport_service.city_code in
                ( select city.city_code from city where
                  city.city_name = "BALTIMORE"
                )
            ) and flight.flight_days in
            ( select days.days_code from days where days.day_name in
                ( select date_day.day_name from date_day where
                  date_day.year = 1993 and date_day.month_number = 6 and
                  date_day.day_number = 14
                )
            )
        )
    )
  )
```

Figure A.2: SQL query for minimum reference answer.

Appendix B

The Two Grammars, G_H and G_{SA}

AIRCRAFT: aircraft, plane, aircrafts, planes, airplane, airplanes, aeroplane, aeroplanes
AIRCRAFT_CODE: d ten, seventy three s, seven fifty seven, m eighty, seven thirty three, m eight zero, seventy two s, d nine s, d c tens, d c ten, MANUFACTURE + DIGIT, AIRCRAFT + DIGIT
AIRLINE: airline, airlines
AIRLINE_NAME: american, american airline, american airlines, american flights, air canada, alaska airlines, alaska airline, continental, continental airline, continental airlines, canadian airline, anadian airlines, canadian airlines international, delta, delta airline, delta airlines, tower air, america west, northwest, northwest airline, nationair, t w, united, southwest, southwest air, southwest airlines, midwest express, united airline, united airlines, trans world airlines, trans world airline, a a, a c, a s, c o, c p, d l, f f, h p, n w, n x, t w a, u a, u s, u s air, w n, y x, k w
AIRPORT: airport, airports

continued on next page ...

continued from previous page ...

AIRPORT_NAME: boston airport, love field, dulles, houston intercontinental, kennedy, kennedy airport, john f kennedy, john f kennedy airport, midway, los angeles international, los angeles international airport, los angeles airport, la guardia, la guardia airport, orlando airport, orlando international, general mitchell, general mitchell international, general mitchell international aiport, ontario airport, ontario international, o'hare, saint petersburg airport, san francisco international, san francisco international airport, san francisco airport, salt lake airport, salt lake city airport, toronto international, toronto international airport, lester pearson airport, newark airport, b n a, b o s, b u r, d a l, d f w, e w r, h o u, i a d,i a h, j f k, l a x, m c o, m a, m k e, o r d, p i e, s f o, s l c, c v g, t p a, l g a, b w i, d t w, y y z
BACK: returns, return, returning
CITY: cities, city
CITY_NAME: westchester, westchester county, atlanta, baltimore, boston, burbank, charlotte, chicago, cincinnati, cleveland, columbus, dallas, denver, detroit, fort worth, houston, indianapolis, kansas city, vegas, las vegas, long beach, los angeles, memphis, miami, milwaukee, inneapolis, montreal, nashville, new york, new york's, new york city, newark, oakland, ontario, orlando, philadelphia, phoenix, pittsburgh, salt lake, salt lake city, san diego, san francisco, san jose, seattle, st. louis, saint louis, st. paul, saint paul, st. petersburg, saint petersburg, tacoma, tampa, toronto, washington, l a, philly, canada
CLASS: classes, class
CLASS_NAME: business, business class, first class, coach, economy
CODE: code, codes
CODE_NAME: s, s slash, a p, a p slash, h, f, y, y n, q, q oh, b, q o, s a, a p fifty eight, b h, a p slash fifty seven

continued on next page ...

continued from previous page ...

COMPARSION: less than, more than, equal, equal to, same, same as
CONNECTIONS: connection, connections, combination, combinations, connecting, connecting flights, direct flights, connecting flight
CONNECTIVE: slash, and, or, either, but, also
COST: DIGIT + MONEY_UNIT
DAY: second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, thirteenth, fourteenth, fifteenth, sixteenth, seventeenth, eighteenth, nineteenth, twentieth, twenty first, twenty second, twenty third, twenty fourth, twenty fifth, twenty sixth, twenty seventh, twenty eighth, twenty ninth, thirtieth, thirty first
DAY_NAME: day, days, week, weeks, weekday, weekend, week days, week day, weekdays, monday, tuesday, wednesday, thursday, friday, saturday, sunday, during the week, today, yesterday, tomorrow, tonight, monday's, tuesday's, wednesday's, thuraday's, friday's, saturday's, sunday's, now, mondays, tuesdays, wednesdays, thursdays, fridays, saturdays, sundays
DIGIT: oh, zero, one, two, three, four, five, six, seven, eight, nine, ten, eleven, twelve, thirteen, fourteen, fifteen, sixteen, seventeen, eighteen, nineteen, twenty, thirty, forty, fifty, sixty, seventy, eighty, ninety, hundred, thousand, hundreds, thousands, single, double, ones, twos, threes, fours, fives, sevens, eights, nines, tens, twentys, thirtys, fortys, fiftys, sixtys, seventys, eightys, ninetys
DUMMY: may i, need to, want to, like to, would like to, i would like, i would like to, show me, ineed, i want, i need to, i want to, trying to, try to, the, a, an, please
FARE: fare, costs, cost, price, fares, airfare, airfares, prices, air fare, air fares, flight fare, flight fares, flight price

continued on next page ...

continued from previous page ...

FIRST:	first
FLIGHT:	flight, flights, fly, flies, flying
FLIGHT_DAYS:	everyday, daily
FLIGHT_NUM:	flight number, flight numbers
FLIGHT_NUMBER:	FLIGHT + DIGIT, AIRLINE_NAME + DIGIT
FROM:	from, departing from, depart from, leave from, leaving, start from, starting from, flying from, fly from, flies from, takeoff from, goes from, go from, take off, takes off, taking off, travel from, departs, depart, departure, departing, leave, leaves, leaving from, takeoff, takeoffs, come from, coming from, comes from
HOW:	how much, how many, how far, how long, how about
KIND:	kind, type, types, kinds, sort
MUNUFACTURER:	boeing, mcdonell donglas
MEAL:	meal, meals
MEAL_DESCRIPTION:	dinner, lunch, snack, supper, breakfast, snacks
MEAN:	mean, stand for, meaning, stands for
MODIFIER:	late, early, earliest, earlier, mid, latest, last, later, next, red eye
MONEY_UNIT:	dollar, dollars
MONTH:	january, february, march, april, may, june, july, august, september, october, november, december

continued on next page ...

continued from previous page ...

ONE_WAY: one way
OR: slash, or
PERIOD: morning, afternoon, evening, day, night, midday, mid-day, breakfast time, lunch time, dinner time, lunchtime, lunch time, dinnertime, noontime, noon, mornings, nights, midnight, mid-night
PRE_TIME: before, after, at, around, about, by
PREP: on, in, between, with, of, for, up, out, under, off
RESTRICTION: restriction, restrictions
ROUND_TRIP: round trip, round trip flight, round trip ticket, round trips, and back
SERVE: serve, served, serves, service, serving
STATE_CODE: d c
STATE_NAME: arizona, california, colorado, florida, indiana, michigan, minnesota, missouri, nevada, new jersey, new york, north carolina, ohio, quebec, tennessee, texas, utah, washington
STOPS: nonstops, nonstop, one stop, at least one stop
SUPERLATIVE: cheapest, closest, expensive, highest, lowest, shortest, smallest, minimum, maximum, most, least
TO: be there, into, to, arrive to, arriving to, arrives to, arrived to, landing in, land in, fly to, destination, back to, go to, arrive, arrives, arriving, arrived, landed, land, lands, landing, landings, arrival, reach, reaches, reaching

continued on next page ...

continued from previous page ...

TIME: time, times
TIME_UNIT: a m, p m, o' clock, o'clock, o clock, hour, hours
TIME_VAULE: DIGIT + TIME_UNIT, PRE_TIME + DIGIT
TRANSPORT: transport, transportation, ground transportation, ground transport
TRANSPORT_TYPE: rental car, rent a car, need a car, taxi, limousine, train
VIA: via, by way, stop, stopover, stopovers, stopping, stoping in, stops in, stopover in, stop over in, stopping over in, layover in, laying over in, make a stop, goes through, go through
WHAT: what're, what's, what
WHERE: where, anywhere
WHICH: which
YEAR: nineteen ninety three

Table B.1: The hand-designed grammar (G_H).

AIRCRAFT: aircraft, plane, aircrafts, planes
AIRLINE_NAME: air canada, alaska airline, alaska airlines, america west, american, american airline, american airlines, american flights, canadian airline, canadian airlines, canadian airlines international, continental, delta, delta airline, delta airlines, midwest express, n w, n x, nationair, northwest, northwest airline, southwest, southwest air, southwest airlines, t w, t w a, tower air, trans world airline, trans world airlines, united, united airline, united airlines, a a, a c, a s, u a, u s, u s air, c o, c p, d l, f f, h p, k w, w n, y x
AIRPORT_NAME: boston airport, dulles, general mitchell, general mitchell international, general mitchell international aiport, h o u, houston intercontinental, kennedy airport, la guardia, la guardia airport, lester pearson airport, los angeles airport, los angeles international, los angeles international airport, love field, newark airport, o'hare, ontario airport, ontario international, ord, orlando airport, orlando international, saint petersburg airport, salt lake airport, salt lake city airport, san francisco airport, san francisco international, san francisco international airport, toronto international, toronto international airport, b n a, b o s, b u r, b w i, c v g, d a l, d f w, d t w, e w r, i a d, i a h, j f k, l a x, l g a, m c o, m i a, m k e, o r d, p i e, s f o, s l c, t p a, y y z
CITY_NAME: atlanta, baltimore, boston, burbank, canada, charlotte, chicago, cincinnati, cleveland, columbus, dallas, denver, detroit, fort worth, houston, indianapolis, kansas city, l a, las vegas, long beach, los angeles, memphis, miami, milwaukee, minneapolis, montreal, nashville, new york, new york city, new york's, newark, oakland, ontario, orlando, philadelphia, philly, phoenix, pittsburgh, saint louis, saint paul, saint petersburg, salt lake, salt lake city, san diego, san francisco, san jose, seattle, st. louis, st. paul, st. petersburg, tacoma, tampa, toronto, washington, westchester, westchester county
COST: DIGIT + MONEY_UNIT
CLASS_NAME: business, business class, coach, economy, first class

continued on next page ...

continued from previous page ...

CODE_NAME: s, s slash, a p, a p slash, h, f, y, y n, q, q oh, b, q o, s a, a p fifty eight, b h, a p slash fifty seven
COMPARSION: less than, more than, equal, equal to, same, same as
CONNECTIONS: connecting flights, direct flights, connecting flight
DAY: first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, thirteenth, fourteenth, fifteenth, sixteenth, seventeenth, eighteenth, nineteenth, twentieth, twenty first, twenty second, twenty third, twenty fourth, twenty fifth, twenty sixth, twenty seventh, twentyeighth, twenty ninth, thirtieth, thirty first
DAY_NAME: weekday, weekend, week days, week day, weekdays, monday, tuesday, wednesday, thursday, friday, saturday, sunday, today, yesterday, tomorrow, tonight, monday's, tuesday's, wednesday's, thuraday's, friday's, saturday's, sunday's, now, mondays, tuesdays, wednesdays, thursdays, fridays, saturdays, sundays
DIGIT: oh, zero, one, two, three, four, five, six, seven, eight, nine, ten, eleven, twelve, thirteen, fourteen, fifteen, sixteen, seventeen, eighteen, nineteen, twenty, thirty, forty, fifty, sixty, seventy, eighty, ninety, hundred, thousand, hundreds, thousands, single, double, ones, twos, threes, fours, fives, sevens, eights, nines, tens, twentys, thirtys, fortys, fiftys, sixtys, seventys, eightys, ninetys
FARE: fare, costs, cost, price, fares, airfare, airfares, prices
FLIGHT: flight, flights, fly, flies, flying
FLIGHT_DAYS: daily, everyday
FLIGHT_NUM: flight number, flight numbers
FLIGHT_NUMBER: FLIGHT + DIGIT, AIRLINE_NAME + DIGIT

continued on next page ...

continued from previous page ...

FROM: from, departing from, depart from, leave from, leaving, start from, starting from, flying from, fly from, flies from, takeoff from, goes from, go from, take off, takes off, taking off, travel from, departs, depart, departure, departing, leave, leaves, leaving from, takeoff, takeoffs, come from, coming from, comes from
HOW: how much, how many
KIND: kind, type, types
MEAL_DESCRIPTION: breakfast, dinner, lunch, snack, snacks, supper
MODIFIER: earliest, latest
MONTH: january, february, march, april, may, june, july, august, september, october, november, december
MONEY_UNIT: dollar, dollars
PERIOD: afternoon, breakfast time, day, dinner time, dinnertime, evening, lunch time, lunchtime, mid-day, mid-night, midday, midnight, morning, mornings, night, nights, noon, noontime
PRE_TIME: before, after, at, around, about, by
ROUND_TRIP: round trip, round trip flight, round trip ticket, round trips
STATE_NAME: arizona, california, colorado, florida, indiana, michigan, minnesota, missouri, nevada, new jersey, new york, north carolina, ohio, quebec, tennessee, texas, utah, washington
STOPS: nonstops, nonstop, one stop, at least one stop
SUPRELATIVE: cheapest, closest, expensive, highest, lowest, shortest, smallest, minimum, maximum, most, least

continued on next page ...

continued from previous page ...

TRANSPORT: transport, transportation, ground transportation, ground transport
TRANSPORT_TYPE: rental car, rent a car, need a car, taxi, limousine, train
TIME_UNIT: a m, p m, o' clock, o'clock, o clock, hour, hours
TIME_VALUE: DIGIT + TIME_UNIT
TO: to, arrive to, arriving to, arrives to, arrived to, landing in, land in, fly to, destination, back to, go to, arrive, arrives, arriving, arrived, landed, land, lands, landing, landings, arrival, reach, reaches, reaching
VIA: via, by way, stop, stopover, stopovers, stopping, stoping in, stops in, stopover in, stop over in, stopping over in, layover in, laying over in, make a stop

Table B.2: The semi-automatically generated grammar (G_{SA}).

Appendix C

Probability Propagation in Belief Networks

The algorithm for computing the probabilities of the Belief Network (BN) in Bayesian inferencing is quite complicated as the probability propagating effect will be different for the networks with different topologies. We will illustrate an example using a network as depicted in Figure C.1. For quantitative modeling, we will assign conditional probabilities for each node $P(x|\, parent(x))$ if there are arcs coming into the node x; or prior probability $P(x)$ if the node x has no parents nodes. Hence we have several probability assessments for our BN : $P(G)$, $P(C_1|G)$, $P(C_2|C_1, G)$ and $P(C_3|G)$, in which $P(G) = (0.5, 0.5)$ and the remaining probabilities are listed in Table C.1.

As introduced in Section 3.4.2, Bayes' rule can be applied directly for BN with pre-defined structure for Bayesian inferencing. However, for our example BN with $U = (G, C_1, C_2, C_3)$, in order to compute the probability of each node, we have to represent our BN as a cluster tree [20]. A cluster tree which is corresponded to our BN is depicted in Figure C.2, it is a tree of

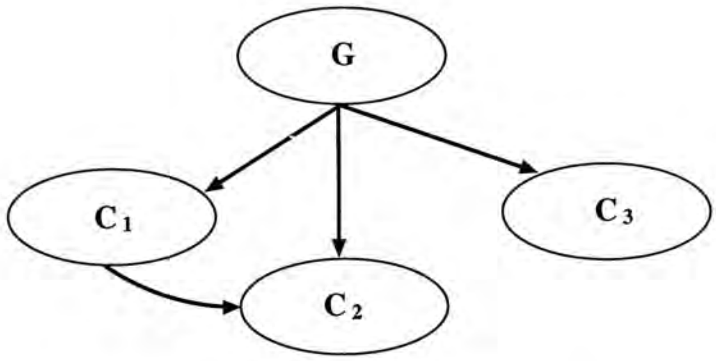

Figure C.1: The topology of the BN.

$P(C_1 \mid G)$	$G=1$	$G=0$
$C_1=1$	0.8	0.3
$C_1=0$	0.2	0.7

$P(C_2 \mid C_1, G)$	$G=1$	$G=0$
$C_1=1$	$(0.8, 0.2)$	$(0.2, 0.8)$
$C_1=0$	$(0.5, 0.5)$	$(0.1, 0.9)$

$P(C_3 \mid G)$	$G=1$	$G=0$
$C_3=1$	0.55	0.4
$C_3=0$	0.45	0.6

Table C.1: The conditional probabilities for the BN in Figure C.1. The vector (α, β) in the middle table represents $(C_2 = 1, C_2 = 0)$.

clusters of variable from U. There are two clusters: (G, C_1, C_2) and (G, C_3). The cluster is also known as clique is subset of U, and the union of all cliques is U. Notice that a link is labeled as separator which consists of the intersection of the adjacent nodes. The probability propagation will communicate through the separator for Bayesian inferencing. Besides, each clique is associated with a joint probability table while the separator is associated with the prior probability.

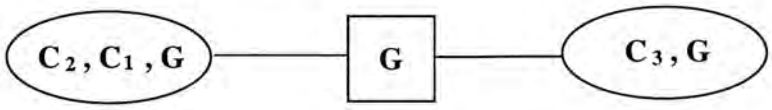

Figure C.2: The clusters for our BN example. The clusters can communicate through the separator variable G.

We have established joint probability tables for the two cliques based on the Bayes' rule:

i. $P(C_2, C_1, G) = P(C_2|C_1, G)P(C_1|G)P(G)$

ii. $P(C_3, G) = P(C_3|G)P(G)$

The results of the $P(C_2, C_1, G)$ and $P(C_3, G)$ are tabulated in Table C.2. Marginalizing the C_1 and C_2 out of $P(C_2, C_1, G)$ and C_3 out of $P(C_3, G)$ yield the prior information of C_1, C_2 and C_3: $P(C_1) = (0.55, 0.45)$, $P(C_2) = (0.435, 0.565)$, and $P(C_3) = (0.475, 0.525)$.

	$P(C_2, C_1, G)$	
	$G = 1$	$G = 0$
$C_1 = 1$	$(0.32, 0.08)$	$(0.03, 0.12)$
$C_1 = 0$	$(0.05, 0.05)$	$(0.035, 0.315)$

	$P(C_3, G)$	
	$G = 1$	$G = 0$
$C_3 = 1$	0.275	0.2
$C_3 = 0$	0.225	0.3

Table C.2: The joint probabilities for the two clusters. The vector (α, β) in the left-hand table represents $(C_2 = 1, C_2 = 0)$.

C.1 Computing the aposteriori probability of $P^*(G)$ based on input concepts

Suppose we have evidence $C_1 = 1$, $C_2 = 1$ and $C_3 = 0$ for the BN, and we aim to compute the updated probability $P^*(G)$. Notice that the probability associated with the asterisk (*) means its value has been updated by the instantiation of its / other nodes.

Step 1: For the input evidence $C_1 = 1$, (i.e. $P^*(C_1 = 1) = 1$)

We will instantiate the node C_1, (i.e. the value of the node is known from observation), the probabilities of the network will be updated and it is done by the propagation of the instantiated node C_1. Since C_1 belongs to the clique (C_2, C_1, G), the joint probability of $P(C_2, C_1, G)$ will be updated first. Based on the Bayes' rule, we have:

$$P^*(C,G) = P(G|C)P^*(C) \quad and \quad P^*(C,G) = P(G,C)\frac{P^*(C)}{P(C)}$$

Hence we get the equation below for updating the joint probability of $P(C_2, C_1, G)$, and the table for the updated $P^*(C_2, C_1, G)$ is tabulated in Table C.3.

$$P^*(C_2, C_1, G) = P(C_2, C_1, G)\frac{P^*(C_1)}{P(C_1)} \tag{C.1}$$

	G=1	G=0
$C_1 = 1$	$(0.32, 0.08) \times \frac{1}{0.55}$	$(0.03, 0.12) \times \frac{1}{0.55}$
$C_1 = 0$	$(0.05, 0.05) \times 0$	$(0.035, 0.315) \times 0$

$\rightarrow$

	G=1	G=0
$C_2 = 1$	0.5818	0.0545
$C_2 = 0$	0.1455	0.218

Table C.3: Updated joint probability $P^*(C_2, C_1, G)$.

By marginalization, we get $P^*(C_2) = (0.6363, 0.3637)$. The updated joint probability table of $P^*(C_2, C_1, G)$ is also reduced to $P^*(C_2, G)$.

Step 2: For the input evidence $C_2 = 1$, (i.e. $P^{**}(C_2 = 1) = 1$)

The joint probability of $P^*(C_2, G)$ will be updated again by Equation C.2 and the result is shown in Table C.4.

$$P^{**}(C_2, G) = P^*(C_2, G)\frac{P^{**}(C_2)}{P^*(C_2)} \tag{C.2}$$

	$G=1$	$G=0$
$C_2=1$	$0.5818 \times \frac{1}{0.6363}$	$0.0545 \times \frac{1}{0.6363}$
$C_2=0$	0.1455×0	0.218×0

$\rightarrow$

	$G=1$	$G=0$
$C_2=1$	0.9143	0.0857
$C_2=0$	0	0

Table C.4: Updated joint probability $P^{**}(C_2, G)$.

Again by marginalization, we get $P^*(G) = (0.9143, 0.0857)$. Since G is the separator between the two clusters, we will then update the probability in the cluster (C_3, G) by G based on the Equation C.3, and the result is shown in Table C.5.

$$P^*(C_3, G) = P(C_3, G)\frac{P^*(G)}{P(G)} \tag{C.3}$$

	$G=1$	$G=0$
$C_3=1$	$0.275 \times \frac{0.9143}{0.5}$	$0.2 \times \frac{0.0875}{0.5}$
$C_3=0$	$0.225 \times \frac{0.9143}{0.5}$	$0.3 \times \frac{0.0875}{0.5}$

$\rightarrow$

	$G=1$	$G=0$
$C_3=1$	0.5029	0.0343
$C_3=0$	0.4114	0.0514

Table C.5: Updated joint probability $P^*(C_3, G)$.

By marginalization, we get $P^*(C_3) = (0.5372, 0.4628)$.

Step 3: For the input evidence $C_3 = 0$, (i.e. $P^{**}(C_3 = 1) = 0$)

Again, the joint probability of $P^*(C_3, G)$ will be updated according to Equation C.4 and the result is tabulated in Table C.6.

$$P^{**}(C_3, G) = P^*(C_3, G)\frac{P^{**}(C_3)}{P^*(C_3)} \tag{C.4}$$

	$G=1$	$G=0$
$C_3=1$	0.5029×0	0.0343×0
$C_3=0$	$0.4114 \times \frac{1}{0.4628}$	$0.0514 \times \frac{1}{0.4628}$

$\rightarrow$

	$G=1$	$G=0$
$C_3=1$	0	0
$C_3=0$	0.8889	0.1111

Table C.6: Updated joint probability $P^*(C_3, G)$.

We get the final updated $P^{**}(G)$ by marginalization of the joint probability in Table C.6, and the updated value of $P^{**}(G)$ is (0.8889,0.1111).

C.2 Computing the aposteriori probability of $P^*(C_j)$ by backward inference

When the BN performs backward inference, the node G is instantiated to a value (either 1 or 0). In this example, we will illustrate the calculation of the updated value of $P^*(C_1)$, $P^*(C_2)$ and $P^*(C_3)$ when the node G is instantiated to value of 1 (i.e. with the observed evidence of $P^*(G=1)=1$). Again, since our BN is divided into two cluster trees, the aposteriori probabilities of $P^*(C_j)$ is obtained by the marginalization of the corresponding updated joint probabilities.

Step 1: Update the joint probability of $P(C_2, C_1, G)$ to get $P^*(C_1)$ and $P^*(C_2)$

The new joint probability, $P^*(C_2, C_1, G)$ can be obtained based on Equation C.5 with the evidence of $P^*(G=1)=1$ and the result is shown in Table C.7.

$$P^*(C_2, C_1, G) = P(C_2, C_1, G)\frac{P^*(G)}{P(G)} \tag{C.5}$$

	$G=1$	$G=0$
$C_1=1$	$(0.32, 0.08) \times \frac{1}{0.5}$	$(0.03, 0.12) \times 0$
$C_1=0$	$(0.05, 0.05) \times \frac{1}{0.5}$	$(0.035, 0.315) \times 0$

$\rightarrow$

	$G=1$	$G=0$
$C_1=1$	$(0.64, 0.16)$	$(0,0)$
$C_1=0$	$(0.1, 0.1)$	$(0,0)$

Table C.7: Updated joint probability $P^*(C_2, C_1, G)$. The vector (α, β) represents $(C_2 = 1, C_2 = 0)$.

By marginalization from the updated joint probability $P^*(C_2, C_1, G)$, we get the updated values of $P^*(C_1) = (0.8, 0.2)$ and $P^*(C_2) = (0.74, 0.26)$ respectively.

Step 2: Update the joint probability of $P(C_3, G)$ to get $P^*(C_3)$.
The new joint probability is again obtained by the Equation C.6, and the result is illustrated in Table C.8.

$$P^*(C_3, G) = P(C_3, G)\frac{P^*(G)}{P(G)} \tag{C.6}$$

	$G=1$	$G=0$
$C_3=1$	$0.275 \times \frac{1}{0.5}$	0.2×0
$C_3=0$	$0.225 \times \frac{1}{0.5}$	0.3×0

$\rightarrow$

	$G=1$	$G=0$
$C_3=1$	0.55	0
$C_3=0$	0.45	0

Table C.8: Updated joint probability $P^*(C_3, G)$.

Finally,we get the updated value of $P^*(C_3) = (0.55, 0.45)$ by the marginalization of the updated joint probability $P(C_3, G)$.

Appendix D

Total 23 Concepts for the Handcrafted BN

<AIRCRAFT>	<CODE>	<MONTH>
<AIRLINE>	<CODE_NAME>	<PERIOD>
<AIRLINE_NAME>	<DAY>	<STOPS>
<AIRPORT>	<DAY_NAME>	<TIME_VALUE>
<AIRPORT_NAME>	<FARE>	<TO>
<CITY_ORIGIN>	<FLIGHT>	<TRANSPORT>
<CITY_DESTINATION>	<FLIGHT_NUMBER>	<TRANSPORT_TYPE>
<CLASS_NAME>	<FROM>	

Table D.1: Total 23 concetps for handcrafted BN.