Acquiring Domain-Specific Dialog Information from Task-Oriented Human-Human Interaction through an Unsupervised Learning

Ananlada Chotimongkol
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
ananlada@cs.cmu.edu

Alexander I. Rudnicky
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
air@cs.cmu.edu

Abstract

We describe an approach for acquiring the domain-specific dialog knowledge required to configure a task-oriented dialog system that uses human-human interaction data. The key aspects of this problem are the design of a dialog information representation and a learning approach that supports capture of domain information from in-domain dialogs. To represent a dialog for a learning purpose, we based our representation, the *form-based dialog structure representation*, on an observable structure. We show that this representation is sufficient for modeling phenomena that occur regularly in several dissimilar task-oriented domains, including information-access and problem-solving. With the goal of ultimately reducing human annotation effort, we examine the use of unsupervised learning techniques in acquiring the components of the form-based representation (i.e. task, subtask, and concept). These techniques include statistical word clustering based on mutual information and Kullback-Liebler distance, TextTiling, HMM-based segmentation, and bisecting *K*-mean document clustering. With some modifications to make these algorithms more suitable for inferring the structure of a spoken dialog, the unsupervised learning algorithms show promise.

1 Introduction

In recent dialog management frameworks, such as *RavenClaw* (Bohus and Rudnicky, 2003) and *Collagen* (Rich et al., 2001), domain-dependent components of a dialog manager are clearly separated from domain-independent components. This separation allows rapid development of a dialog management module in a new task-oriented domain as dialog system developers can focus only on specifying domain-specific dialog information (e.g. the *Dialog Task Specification* in RavenClaw and *Task Models* in Collagen) while general dialog behaviors (e.g. turn-taking, confirmation mechanism, and generic help) are provided by the framework. For task-oriented domains, the domain-specific dialog information is equivalent to task-specific information. Examples of the task-specific information are steps in a task and domain keywords.

Specifying task-specific knowledge by hand is still a time consuming process (Feng et al., 2003). Furthermore, the hand-crafted knowledge may not reflect users' perceptions of a task (Yankelovich, 1997). To reduce the subjectivity of system developers, recorded conversations of humans performing a similar task as a target dialog system have been used to help the developers design the task specification. Nevertheless, analyzing a corpus of dialogs by hand requires a great deal of human effort (Bangalore et al., 2006). This paper investigates the feasibility of automating this dialog analysis process through a machine-learning approach. By inferring the task-specific dialog information automatically from human-human interaction data, the knowledge engineering effort could be reduced as the developers need to only revise learned information rather than analyzing a large amount of data.

Acquiring the task-specific knowledge from a corpus of human-human dialogs is considered a knowledge acquisition process, where the target task structure has not yet been specified but will be explored from data before a dialog system is built. This is contrasted with a dialog structure recognition process (Alexandersson and Reithinger, 1997;

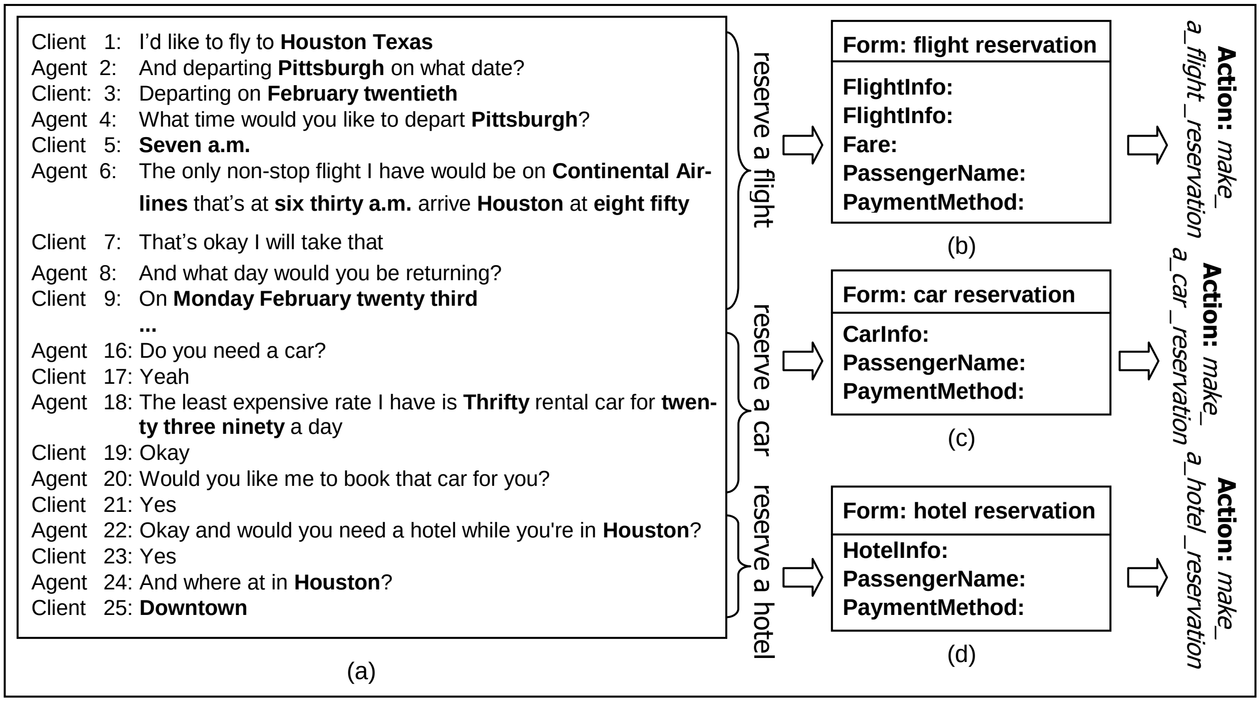
Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 955–964,
Honolulu, October 2008. ©2008 Association for Computational Linguistics

Bangalore et al., 2006; Hardy et al., 2004), where pre-specified dialog structure components are recognized as a dialog progresses.

We use an unsupervised learning approach in our knowledge acquisition process as it can freely explore the structure in the data without any influence from human supervision. Woszczyna and Waibel (1994) showed that when modeling a dialog state transition diagram from data an unsupervised approach outperformed a supervised one as it better reflects the characteristic of the data. It is also interesting to see how well a machine-learning approach can perform on the problem of task-specific knowledge acquisition when no assumption about the domain is made and no prior knowledge is used.

Examination of task-oriented human-human dialogs show that task-specific information can be observed in dialog transcription; therefore, it should be feasible to be infer it through an unsupervised learning approach. Figure 1 (a) shows a dialog in an air travel domain. This dialog is organized into three parts according to the three steps (i.e. reserve a flight, reserve a car, reserve a hotel) required to accomplish the task, creating a travel itinerary. Domain keywords (highlighted in bold) required to accomplish each step are clearly communicated.

To infer task-specific knowledge from data using an unsupervised learning approach, two problems need to be addressed: 1) choosing an appropriate dialog representation that captures observable task-specific knowledge in a dialog, and 2) developing an unsupervised learning approach that infers the task-specific knowledge modeled by this representation from in-domain human-human dialogs. The first problem is discussed in Section 3 where a *form-based dialog structure representation* is proposed. After describing the definition of each component in the form-based dialog structure representation, examples of how a domain expert models the task-specific information in a dialog with the form-based representation are given Section 3.1. Then the annotation experiment which was used to verify that the form-based representation can be understood and applied by other human annotators is discussed in Section 3.2. For the second problem, we modify existing unsupervised learning approaches to make them suitable for inferring the structure of a spoken dialog. Section 4 describes these modifications and their performances when inferring the components of the form-based dialog structure representation from interaction data.



Client 1: I'd like to fly to **Houston Texas**
Agent 2: And departing **Pittsburgh** on what date?
Client: 3: Departing on **February twentieth**
Agent 4: What time would you like to depart **Pittsburgh**?
Client 5: **Seven a.m.**
Agent 6: The only non-stop flight I have would be on **Continental Airlines** that's at **six thirty a.m.** arrive **Houston** at **eight fifty**
Client 7: That's okay I will take that
Agent 8: And what day would you be returning?
Client 9: On **Monday February twenty third**
...
Agent 16: Do you need a car?
Client 17: Yeah
Agent 18: The least expensive rate I have is **Thrifty** rental car for **twenty three ninety** a day
Client 19: Okay
Agent 20: Would you like me to book that car for you?
Client 21: Yes
Agent 22: Okay and would you need a hotel while you're in **Houston**?
Client 23: Yes
Agent 24: And where at in **Houston**?
Client 25: **Downtown**

Figure 1: An example of a dialog in the air travel domain and its corresponding form-based representation

2 Related Work

Automatic task-specific knowledge acquisition for configuring a dialog system is a relatively new research area. Supervised learning approaches were used to acquire a task model for a collaborative agent (Garland et al., 2001) and task-specific information for a customer care service (Feng et al., 2003). These supervised algorithms were trained on rich knowledge sources (examples described in a specific annotation language and a well-organized website respectively) annotated by domain experts. In contrast, the unsupervised conceptual clustering algorithm in DIA-MOLE (Möller, 1998) requires no additional human annotation to infer a set of domain-specific dialog acts from in-domain dialogs. The motivation behind the use of an unsupervised approach is similar to ours, to reduce human effort in creating a new dialog system.

3 Form-based Dialog Structure Representation

Many models have been proposed to account for the structure of a human-human conversation. Many such models focus on other aspects of a dialog such as coordinated activities, i.e. turn-taking and grounding, (Traum and Hinkelman, 1992) and regular patterns in the dialog (Carletta et al., 1997) rather than the domain-specific information communicated by participants. More complicated dialog representations (Grosz and Sidner, 1986; Litman and Allen, 1987) model several aspects of a dialog including domain-specific information. However, additional components in these models, such as beliefs and intentions, are difficult to observe directly from a conversation and, as for the current technology, may not be learnable through an unsupervised learning approach.

Since the task-specific information that we would like to model will be used for configuring a dialog system, we can view this information from a dialog system perspective. Our dialog representation is based on *form*, a data representation used in a form-based (or frame-based) dialog system. A form is a simple representation that captures necessary task-specific information communicated through dialog. This information is observable from dialog transcription (see below) and thus could be inferred through an unsupervised learning approach.

Typically, a form corresponds to a database query form while slots in the form represent search criteria. Nevertheless, a form can represent related pieces of information required to perform any domain action not just a database query action. With this more general definition of a form, a *form-based dialog structure representation* can be applied to various types of task-oriented domains where dialog participants have to gather pieces of information, analogous to search criteria, through dialog in order to perform domain actions that fulfill a dialog goal. Chotimongkol (2008) provided examples of these domains, for instance, meeting (Banerjee and Rudnicky, 2006) and flight simulation control (Gorman et al., 2003).

In the form-based dialog structure representation, task-specific information in each dialog is organized into a three-level structure of concept, subtask and task. A *concept* is a word or a group of words which captures a piece of information required to perform a domain action. A *subtask* is a subset of a dialog which contains sufficient concepts to execute a domain action that advances a dialog toward its goal. A *task* is a subset of a dialog (usually the entire dialog) which contains all the subtasks that belong to the same goal. A subtask can also be considered as a step in a task. In terms of representation, a task is represented by a set of forms, one for each of its subtasks. A concept is a slot in a form.

To model the structure of a dialog in a new domain with the form-based dialog structure representation, a list of tasks, subtasks, and concepts in that domain has to be specified. This list is considered a domain-specific tagset. The form-based dialog structure framework only provides the definitions of these components (i.e. task, subtask, and concept), which can be regarded as meta-tags and are domain-independent. A list of tasks, subtasks, and concepts can be identified manually as shown in Section 3.1 or automatically through a machine-learning approach as discussed in Section 4. Section 3.1 illustrates how a domain expert models the task-specific information in two task-oriented domains, air travel planning (information-accessing) and map reading (problem-solving), with the form-based representation. These examples also show that the form-based dialog structure representation is sufficient for modeling task-specific information in dissimilar domains.

Nonetheless, by focusing on observable task-specific information and describing this information using a simple model, the form-based dialog structure representation cannot model the information that is not clearly expressed in a dialog. Example of such information in an air travel domain is the pickup date of a car rental which may not be discussed in a dialog as it can be inferred from the arrival date of the corresponding flight. Furthermore, the form-based representation is not well suited for modeling a complex dialog that has a dynamic structure such as a tutoring dialog.

3.1 Dialog Structure Modeling Examples

Figure 1 illustrates how a dialog in the air travel domain (Eskenazi et al., 1999) can be represented with the form-based dialog structure representation. A dialog in this domain usually has a single goal, to create an air-travel itinerary which may include hotel and car reservations. Thus, the entire dialog corresponds to one task. The dialog in Figure 1 (a) contains three subtasks, one for each ***make_•_reservation*** action. The forms that represent these subtasks are shown in Figure 1 (b) – (d). Each form contains a set of concepts necessary for making the corresponding reservation. For a display purpose, the values of these slots are omitted.

A subtask can be further decomposed. For example, to reserve a round trip ticket, two database lookup actions, one for each leg, are required. A **reserve_flight** subtask in Figure 1 is decomposed into two **query_flight_info** subtasks. The corresponding forms of these subtasks are illustrated in Figure 2. Each **FlighInfo** concept in the flight reservation form is a result of a database lookup action that corresponds to each flight query form.

Form: flight query

DepartCity: Pittsburgh
ArriveCity: Houston
ArriveState: Texas
DepartDate: February twentieth
DepartTime: seven a.m.

Form: flight query

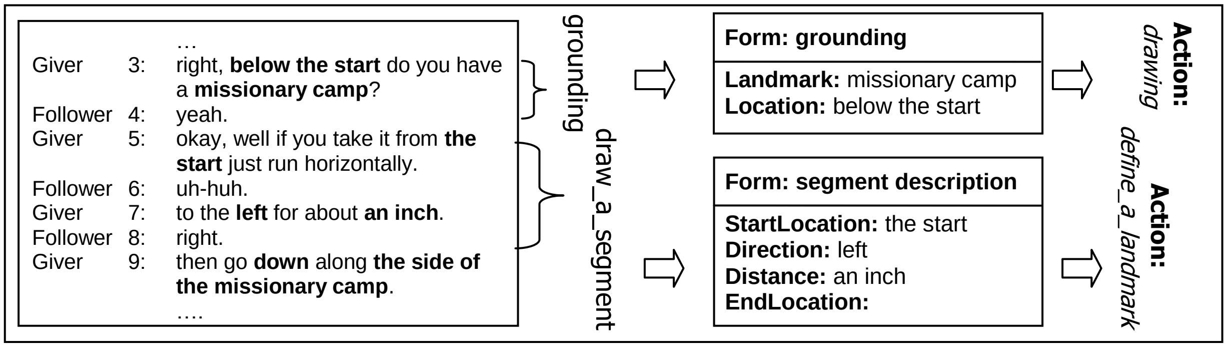
DepartCity: Houston
ArriveCity: Pittsburgh
DepartDate: Monday February twenty third
DepartTime: five p.m.

Form: flight reservation

FlightInfo:
Airline: Continental
DepartTime: six thirty a.m.
ArriveCity: Houston
ArriveTime: eight fifty
FlightInfo:
Airline: Continental
DepartCity: Houston
DepartTime: six forty p.m.
ArriveCity: Pittsburgh
ArriveTime: ten twenty p.m
Fare: four hundred dollars
Name:
PaymentMethod:

Figure 2: An example of subtask decomposition

Figure 3 show a dialog in the map reading domain (Anderson et al., 1991) and its corresponding form-based dialog structure representation. The goal of a dialog in this domain is to have a route follower draw a route on his/her map according to a description given by a route giver. Since drawing an entire route involves several drawing strokes, a **draw_a_route** task is divided into several **draw_a_segment** subtasks, one for each ***drawing*** action. This action required a set of concepts that describe a segment as shown in a segment description form. Since the landmarks on the giver's map can be different from those in the follower's map, the participants have to explicitly define the **Location** of a mismatched **Landmark** before using it in a segment description. In this case **grounding** becomes another subtask and can be represented by a form. This type of grounding is not necessary in the air travel domain.



…
Giver 3: right, **below the start** do you have a **missionary camp**?
Follower 4: yeah.
Giver 5: okay, well if you take it from **the start** just run horizontally.
Follower 6: uh-huh.
Giver 7: to the **left** for about **an inch**.
Follower 8: right.
Giver 9: then go **down** along **the side of the missionary camp**.
….

Form: grounding

Landmark: missionary camp
Location: below the start

Form: segment description

StartLocation: the start
Direction: left
Distance: an inch
EndLocation:

Figure 3: An example of a dialog in the map reading domain and its corresponding form-based representation

3.2 Annotation Experiment

The goal of this annotation experiment is to verify that the form-based dialog structure framework can be understood by human annotators other than its developers, and that they can consistently apply the framework to model task-specific information in a dialog. In this experiment, each annotator had to design a form-based dialog structure representation for a given task-oriented domain by specifying a hierarchical structure of tasks, sub-tasks and concepts in that domain. Note that we are interested in the process of designing a domain-specific tagset from the definitions of task, subtask, and concept provided by the framework, not in the process of using an existing tagset to annotate data (see for example (Carletta et al., 1997)). The description of the framework is provided in annotation guidelines along with examples from the domains that were not used in the experiment.

The experimental procedure is as follows: the subjects first developed their own tagset according to the guidelines by analyzing a set of in-domain dialogs, and then annotate those dialogs with the tagset they had designed. To obtain enough annotated instances for each dialog structure component and to make the annotation simple, the dialog structure annotation part of the experiment was divided into two sub-parts: concept annotation and task/sub-task annotation. Two domains were used in the experiment, air travel planning and map reading. Four subjects were assigned to each domain. None had used the scheme previously. The average number of tags that each subject annotated is shown in the first row of Table 1.

Since some variations in tagset designs are acceptable as long as they conform to the guidelines, each subject's annotation is judged against the guidelines rather than one specific reference annotation. An annotation instance is marked as incorrect only when it does not conform to the guidelines. Each subject's annotation was evaluated by both a coding scheme expert and by other subjects. *Accuracy* is computed from the expert's judgment while *acceptability* is computed from peers' judgments. Acceptability scores shown in Table 1 were averaged from all other subjects in the same group. Please note that the result presented in this table should not be compared to the results from machine-learning approaches presented in Table 2 and Table 3 as the evaluation procedures and data sets are different.

Measure	**Air Travel**		**Map Reading**	
	C	**T**	**C**	**T**
Number of tags	178.8	50.5	347.8	60.8
Accuracy (%)	96.5	89.7	89.0	65.2
Acceptability (%)	95.6	81.1	94.9	84.5

Table 1: Accuracy and acceptability on concept annotation (C), and task/subtask annotation (T)

Both accuracy and acceptability are high for all annotation tasks except for the accuracy of task/subtask annotation in the map reading domain. Most of the errors come from the annotation of the **grounding** subtasks. Since its corresponding action is quite difficult to observe, subjects may not have a concrete definition of **grounding** and were more likely to produce errors. In addition, they were less critical when judging other subjects' annotations. Consistency in applying the form-based dialog structure representation shows that the representation is unambiguous and could potentially be identified through a machine-learning approach.

When comparing among components, concepts were annotated more consistently than tasks and subtasks in terms of both accuracy and acceptability. One possible reason is that, a concept is easier to observe as its unit is smaller than a task or a subtask. Moreover, dialog participants have to clearly communicate the concepts in order to execute a domain action. The subjects usually agreed on tasks and top-level subtasks, but did not quite agree on low-level subtasks. The low-level subtasks are correlated with the implementation of a dialog system; hence, the designs of these subtasks are more subjective and likely to be different.

4 Learning Approaches

This section describes machine-learning approaches for inferring the task-specific information modeled by the form-based dialog structure representation from human-human conversations. Specifically, the learning approach has to infer a list of tasks, sub-tasks and concepts in a given domain from in-domain dialogs similar to what a human does in Section 3. To make the problem tractable, components in the form-based representation are acquired separately. For most task-oriented dialogs that we encountered, each dialog corresponds to one task. Hence, the learning effort can be focused on identifying concept and subtask. Since we can only observe instances or values of these components in a dialog, we have to first identify these instances and then make a generalization for its type. For instant, to infer that there is a concept **City** in the air travel domain, a set of city names has to be identified and grouped together.

To identify a set of domain concepts from the transcription of in-domain dialogs, we follow the algorithm described in (Chotimongkol and Rudnicky, 2002). This algorithm utilizes an unsupervised clustering algorithm which clusters words based on context similarity, e.g. mutual information-based and Kullback-Liebler-based clustering, since the members of the same domain concept are usually used in similar contexts in a particular domain. Examples of the clusters obtained from the KL-based clustering algorithm are shown in Figure 4. These clusters represent **Hour**, **RentalCompany**, and **City** respectively. Underlined cluster members belong to other concepts. The clustering algorithm can identify all 12 members of **Hour** and about half of **RentalCompany**. In the third cluster, some airport names got merged with city names because they occur in quite similar context.

- ONE, TWO, THREE, NINE, SIX, FOUR, SEVEN, FIVE, EIGHT, TEN, TWELVE, ELEVEN
- HERTZ, BUDGET, THRIFTY
- MIDWAY, LAGUARDIA, GATWICK, PHILADELPHIA, DALLAS, DENVER, MONTEREY, BOSTON, CHICAGO, AUSTIN, NEWARK, PITTSBURGH, SEATTLE, OTTAWA, SYRACUSE, BALTIMORE, HOUSTON, MADRID, L.A., ATLANTA, DULLES, HONOLULU

Figure 4: Learned concepts in the air travel domain

The rest of this section describes an approach for identifying subtasks and their corresponding forms in a given domain. We decided to simplify the form-learning problem by first segmenting a dialog into form-filling episodes (which are equivalent to sub-tasks), then grouping the ones that correspond to the same form together so that we can determine a set of necessary slots in each form from the concepts present in its corresponding cluster. We further simplify the problem by concentrating on the domains that have only one top-level task (though in principle the approach can be extended to the domains that have multiple top-level tasks). Since we utilize well-known unsupervised algorithms, only the modifications which are applied to make these algorithms suitable for inferring the structure of a spoken dialog are discussed.

Two unsupervised discourse segmentation algorithms are investigated: TextTiling (Hearst, 1997) and Hidden Markov Modeling (Barzilay and Lee, 2004). These algorithms only recover the sequence of subtasks but not the hierarchical structure of subtasks similar to Bangalore et al.'s (2006) chunk-based model. Nevertheless, this simplification is sufficient when a subtask is embedded at the beginning or the end of the higher-level subtask which is the case for most embedded structures we have found. Both algorithms, while performing well with expository text, require modifications when applying to a fine-grained segmentation problem of spoken dialogs. In WSJ text, the average topic length is 428 words (Beeferman et al., 1999) while in the air travel domain the average subtask length is 84 words (10 utterances).

For TextTiling, the modifications include a distance weight and a data-driven stop word list. For the subtasks that are much shorter than the average length, distant words in the context window can be irrelevant. A *distance weight* demotes the importance of the context word that is far away from the considered boundary by giving it a lower weight. A manually prepared stop word list, containing common words, may not be suitable for every application domain. We propose a novel approach that determines a list of stop words directly from word distribution in each data set. TextTiling assumes that words that occur regularly throughout a dialog are not informative. However, the regularity of a particular word is determined from its distribution over the dialog rather than from its frequency. A high frequency word is useful if its instances occur only in a specific location. For example, the word "delta" which occurs many times in a **reserve_flight** subtask but does not occur in other subtasks is undoubtedly useful for determining subtask boundaries while the word "you" which can occur anywhere in a dialog is not useful. Specifically, a *regularity count* of word *w* is defined as the number of sliding context windows in the similarity score calculation of TextTiling that contain the word *w* in each dialog. A *data-driven stop word list* contains words that have a regularity count greater than a pre-defined threshold.

For HMM-based segmentation, we modified Barzilay and Lee's (2004) content models by using larger text spans when inducing the HMM states. HMM states are created automatically by clustering similar text spans together. When using an utterance as a text span, it may not contain enough information to indicate its relevant subtask as some utterances in a task-oriented dialog are very short and can occur in any subtask (e.g. acknowledgements and yes/no responses). Larger text spans, reference topics, were used in (Yamron et al., 1998). Nevertheless, this approach requires true segment boundaries. To eliminate the need of annotated data in our algorithm, HMM states are induced from predicted segments generated by TextTiling instead.

After segmenting all dialogs into sequences of subtasks, the bisecting *K*-means clustering algorithm (Steinbach et al., 2000) is used to group the segments that belong to the same type of subtask together as they represent the same form type. The clustering is done based on cosine similarity between segments. This unsupervised clustering algorithm is also used to infer a set of HMM states in the HMM-based segmentation described above.

Words are used as features for both segmentation and clustering algorithms. If a set of domain concepts has already been identified, we can use this information to enhance the features. When concept annotation is available, we can incorporate a concept label into a representation of a concept word. A *Label+Word* representation joins a word string and its label and can help disambiguate between similar words that belong to different concepts. For instance, "one" in "that one" is not the same token as "**[Hour]:**one". A *Label* representation, on the other hand, only represents a concept word by its label. This representation is based on the assumption that a list of concepts occurring in one subtask is distinguishable from a list of concepts occurring in other subtasks regardless of the values of the concepts; hence, a concept label is more informative than its value. This representation provides an abstraction over all different values of the same concept type. For example, **[Airline]:**northwest and **[Airline]:**delta are represented with the same token **[Airline]**. In all experiments, concept labels are provided by a domain expert as we assume that a set of domain concepts has already been identified.

4.1 Dialog Segmentation Results

To evaluate dialog segmentation performance, we compare predicted boundaries against subtask boundaries annotated by a domain expert. Subtask boundaries could occur only at utterance boundaries. Two metrics are used: P_k (Beeferman et al., 1999) and concept-based f-measure (C. F-1). P_k measures the probability of misclassifying two utterances that are *k* utterances apart as belonging to the same sub-task or different sub-tasks. *k* is set to half the average sub-task length. *C. F-1* is a modification of the standard f-measure (a harmonic mean of precision and recall) that gives credit to some near misses. Since the segmented dialogs will later be used to identify a set of forms and their associated slots, the segment that contains the same set of concepts as the reference segment is acceptable even if its boundaries are slightly different from the reference. For this reason, a near-miss counts as a match if there is no concept between the near-miss boundary and the reference boundary.

We evaluated the proposed dialog segmentation algorithms with 24 dialogs from the air travel domain and 20 dialogs from the map reading domain. The window size for TextTiling was set to 4 utterances. The cut-off threshold for choosing subtask boundaries was set to $\mu - \sigma/2$; where μ is the mean of the *depth scores* (Hearst, 1997), the relative change in word co-occurrence similarities on both sides of a candidate boundary, in each dialog and σ is their standard deviation. We found that a small window size and a low cut-off threshold are more suitable for identifying fine-grained segments as in the case of subtasks. However, we also found that TextTiling is quite robust as varying these two parameters doesn't severely degrade its performance (Chotimongkol, 2008). The threshold for selecting data-driven stop words was set to $\mu + 2*\sigma$; where μ is the mean of the regularity counts of all the words in a given dialog and σ is their standard deviation. The performance of TextTiling and HMM-based segmentation algorithm is shown in Table 2.

Augmented TextTiling, which uses a data-driven stop word list, distance weights, and the *Label+Word* representation, performed significantly better than the baseline in both domains. Each of these augmenting techniques can on their own improve segmentation performance but not significantly. Unsurprisingly, the proposed regularity counts discover stop words that are specific to spoken dialogs, but are absent from the hand-crafted list[1], e.g. “okay” and “yeah”.

Algorithm	Air Travel		Map Reading	
	P_k	C. F-1	P_k	C. F-1
TextTiling (baseline)	0.387	0.621	0.412	0.396
TextTiling (augmented)	0.371	0.712	0.384	0.464
HMM-based (utterance)	0.398	0.624	0.392	0.436
HMM-based (segment)	0.385	0.698	0.355	0.507
HMM-based (segment + Label representation)	0.386	0.706	0.250	0.686

Table 2: Dialog segmentation results

For HMM-based segmentation, the segmentation result obtained when modeling the HMM states from predicted subtasks generated by TextTiling (4th row) is better than the result obtained when modeling the HMM states from utterances (3rd row). Predicted segments provide more context to the clustering algorithm that induces the HMM states. As a result a more robust state representation is obtained. A more efficient clustering algorithm can also improve the performance of the HMM-based segmentation algorithm since it provides a state representation that better differentiates among dialog segments which belong to dissimilar subtasks. When the *Label* representation which yielded a better subtask clustering result (see Section 4.2) was used, HMM-based segmentation produced a better result (5th row) especially in the map reading domain. These numbers may appear modest compared to the numbers obtained when segmenting expository text. However, predicting the boundaries of fine-grained subtasks is more difficult even with a supervised learning approach (Arguello and Rosé, 2006). Our results are comparable to Arguello and Rosé’s (2006) results.

Between the two segmentation algorithms, the HMM-based algorithm performed slightly worse than TextTiling in the air travel domain but performed significantly better in the map reading domain. The HMM-based algorithm can identify more boundaries between fine-grained subtasks, which occur more often in the map reading domain. TextTiling, which relies on local lexical cohesion, is unlikely to find two significant drops in lexical similarity that are only a couple of utterances apart, and thus fails to detect boundaries of short segments. However, HMM-based segmentation misses more boundaries between two subtask occurrences of the same type, which occurs more often in the air travel domain, as they are usually represented by the same state.

4.2 Subtask Clustering Results

We evaluated the subtask clustering algorithm on the same data set used in the dialog segmentation evaluation. Table 3 presents the quality score (QS) for each clustering result. These QSs were obtained by comparing the output clusters against a set of reference subtasks. See (Chotimongkol and Rudnicky, 2002) for the definition of QS.

Feature Representation	Air Travel	Map Reading
Label+Word (oracle)	0.738	0.791
Label+Word	0.577	0.675
Label	0.601	0.823

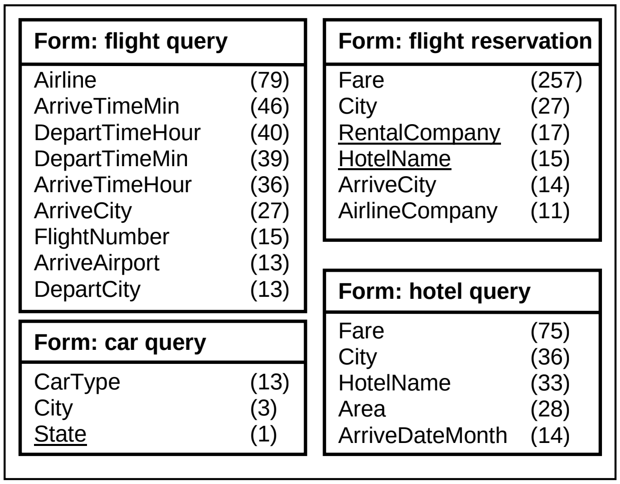
Table 3: Subtask clustering results

When predicted segments were clustered, the quality of the output (2nd row) is not as good as when the reference segments were used (1st row) as inaccurate segment boundaries affected the performance of the clustering algorithm. However, the qualities of subtasks that occur frequently are not much different. In terms of feature representation, the clustering algorithm that uses the *Label* representation achieved better performance in both domains. When the sets of concepts in all of the subtasks are disjoint, the clustering algorithm that uses the *Label* representation can achieve a very good result as in the map reading domain. This result is even better than the result obtained when the reference segments were clustered by the algorithm that uses the *Label+Word* representation. These results demonstrate that an appropriate feature representation provides more useful information to the clustering algorithm than accurate segment boundaries. However, when the subtasks contain overlapping sets of concepts as in the air travel domain, the performance gain obtained from the *Label* representation is quite small.

Figure 5 shows four types of forms in the air travel domain that were acquired by the proposed form identification approach. The slot names are taken from concept labels. The number in parentheses is slot frequency in the corresponding cluster. The underlined slots are the ones that belong to other forms. Some slots in the car query form are

[1] http://search.cpan.org/~creamyg/Lingua-StopWords-0.08/lib/Lingua/StopWords.pm.

missing as some instances of its corresponding subtask get merged into other clusters.



Form: flight query	
Airline	(79)
ArriveTimeMin	(46)
DepartTimeHour	(40)
DepartTimeMin	(39)
ArriveTimeHour	(36)
ArriveCity	(27)
FlightNumber	(15)
ArriveAirport	(13)
DepartCity	(13)

Form: car query	
CarType	(13)
City	(3)
State	(1)

Form: flight reservation	
Fare	(257)
City	(27)
RentalCompany	(17)
HotelName	(15)
ArriveCity	(14)
AirlineCompany	(11)

Form: hotel query	
Fare	(75)
City	(36)
HotelName	(33)
Area	(28)
ArriveDateMonth	(14)

Figure 5: Examples of forms obtained by the proposed unsupervised learning approach

4.3 Discussions on Learning Approaches

The results presented in the previous sections show that existing unsupervised learning algorithms are able to identify components of the form-based dialog structure representation.. However, some modifications are required to make these algorithms more suitable for inferring the structure of a spoken dialog. The advantages of different learning algorithms can be combined to improve performance. For example, TextTiling and HMM-based segmentation are good at detecting different types of boundaries; therefore, combining the predictions made by both algorithms could improve segmentation performance. Additional features such as prosodic features could also be useful.

Subsequent steps in the learning process are subjected to propagation errors. However, the proposed learning algorithms, which are based on generalization of recurring patterns, are able to learn from inaccurate information given that the number of errors is moderate, so that there are enough correct examples to learn from. Given redundant information in dialog corpora, a domain knowledge acquisition process does not require high learning accuracy and an unsupervised learning approach is reasonable. The overall quality of the learning result is acceptable. The proposed unsupervised learning approach can infer much useful task-specific dialog information needed for automatically configuring a task-oriented dialog system from data.

5 Conclusion and Future Directions

To represent a dialog for a learning purpose, we based our representation, the form-based dialog structure representation, on observable information. Components of the form-based representation can be acquired with acceptable accuracy from observable structures in dialogs without requiring human supervision. We show that this dialog representation can capture task-specific information in dissimilar domains. Additionally, it can be understood and applied by annotators other than the developers.

Our investigation shows that it is feasible to automatically acquire the domain-specific dialog information necessary for configuring a task-oriented dialog system from a corpus of in-domain dialogs. This corpus-based approach could potentially reduce human effort in dialog system development. A limitation of this approach is that it can discover only information present in the data. For instance, the corpus-based approach cannot identify city names absent in the corpus while a human developer would know to include these. Revision may be required to make learned information more accurate and complete before deployment; we expect that this effort would be less than the one required for manual analysis. A detailed evaluation of correction effort would be desirable.

In this paper, task-specific knowledge was acquired from in-domain dialogs without using any prior knowledge about the domain. In practice, existing knowledge sources about the world and the domain, such as WordNet, could be used to improve learning. Some human supervision can be valuable particularly in the form of semi-supervised learning and active learning. In particular a process that integrates human input at appropriate times (for example seeding or correction) is likely to be part of a successful approach.

Acknowledgments

This research was supported by DARPA grant NB CH-D-03-0010. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

References

J. Alexandersson and N. Reithinger. 1997. Learning Dialogue Structures From A Corpus. In *Proceedings of EuroSpeech-97*. Rhodes, Greece.

A. H. Anderson, M. Bader, E. G. Bard, E. Boyle, G. Doherty, S. Garrod, S. Isard, J. Kowtko, J. McAllister, J. Miller, C. Sotillo, H. Thompson, and R. Weinert. 1991. The HCRC Map Task Corpus. *Language and Speech,* 34(4):351-366.

J. Arguello and C. P. Rosé. 2006. Topic Segmentation of Dialogue. In *Proceedings of Workshop on Analyzing Conversations in Text and Speech*.

S. Banerjee and A. I. Rudnicky. 2006. You Are What You Say: Using Meeting Participants' Speech to Detect their Roles and Expertise. In *the NAACL-HLT 2006 workshop on Analyzing Conversations in Text and Speech*. New York, NY.

S. Bangalore, G. D. Fabbrizio, and A. Stent. 2006. Learning the Structure of Task-Driven Human-Human Dialogs. In *Proceedings of COLING/ACL 2006*. Sydney, Australia.

R. Barzilay and L. Lee. 2004. Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization. In *Proceedings of HLT-NAACL 2004*. Boston, MA.

D. Beeferman, A. Berger, and J. Lafferty. 1999. Statistical Models for Text Segmentation. *Machine Learning,* 34(1-3):177-210.

D. Bohus and A. I. Rudnicky. 2003. RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. In *Proceedings of Eurospeech2003*. Geneva, Switzerland.

J. Carletta, S. Isard, G. Doherty-Sneddon, A. Isard, J. C. Kowtko, and A. H. Anderson. 1997. The reliability of a dialogue structure coding scheme. *Computational Linguistics*, 23(1):13-31.

A. Chotimongkol. 2008. *Learning the Structure of Task-Oriented Conversations from the Corpus of In-Domain Dialogs,* Ph.D. Thesis CMU-LTI-08-001. Pittsburgh, Carnegie Mellon University.

A. Chotimongkol and A. Rudnicky. 2002. Automatic Concept Identification in Goal-Oriented Conversations. In *Proceedings of ICSLP 2002*. Denver, CO.

M. Eskenazi, A. Rudnicky, K. Gregory, P. Constantinides, R. Brennan, C. Bennett, and J. Allen. 1999. Data Collection and Processing in the Carnegie Mellon Communicator. In *Proceedings of Eurospeech 1999*. Budapest, Hungary.

J. Feng, S. Bangalore, and M. Rahim. 2003. WebTalk: Mining Websites for Automatically Building Dialog Systems. In *Proceedings of ASRU '03*. St. Thomas, U.S. Virgin Islands.

A. Garland, N. Lesh, and C. Sidner. 2001. Learning Task Models for Collaborative Discourse. In *Proceedings of Workshop on Adaptation in Dialogue Systems, NAACL '01*. Pittsburgh, PA.

J. C. Gorman, N. J. Cooke, P. W. Foltz, P. A. Kiekel, and M. J. Martin. 2003. Evaluation of Latent Semantic Analysis-based measures of team communications content. In *Proceedings of the Human Factors and Ergonomic Society 47th Annual Meeting, (HFES 2003)*.

B. J. Grosz and C. L. Sidner. 1986. Attention, Intentions, and the Structure of Discourse. *Computational Linguistics,* 12(3):175-204.

H. Hardy, A. Biermann, R. B. Inouye, A. Mckenzie, T. Strzalkowski, C. Ursu, N. Webb, and M. Wu. 2004. Data-Driven Strategies for an Automated Dialogue System. In *Proceedings of ACL '04*. Barcelona, Spain.

M. A. Hearst. 1997. TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*, 23(1):33-64.

D. Litman and J. Allen. 1987. A Plan Recognition Model for Subdialogues in Conversations. *Cognitive Science,* 11(2):163-200.

J.-U. Möller. 1998. Using Unsupervised Learning for Engineering of Spoken Dialogues. In *Proceedings of AAAI 1998 Spring Symposium on Applying Machine Learning to Discourse Processing*.

C. Rich, C. L. Sidner, and N. Lesh. 2001. Collagen: applying collaborative discourse theory to human-computer interaction. *AI Magazine,* 22(4):15-25.

M. Steinbach, G. Karypis, and V. Kumar. 2000. A Comparison of Document Clustering Techniques. In *Proceedings of KDD Workshop on Text Mining*.

D. R. Traum and E. A. Hinkelman. 1992. Conversation Acts in Task-Oriented Spoken Dialogue. *Computational Intelligence*, 8(3):575--599.

M. Woszczyna and A. Waibel. 1994. Inferring linguistic structure in spoken language. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*.

J. P. Yamron, I. Carp, L. Gillick, S. Lowe, and P. v. Mulbregt. 1998. A Hidden Markov Model Approach to Text Segmentation and Event Tracking. In *Proceedings of ICASSP '98*. Seattle, WA.

N. Yankelovich. 1997. Using Natural Dialogs as the Basis for Speech Interface Design. In Susann Luperfoy (Ed.), *Automated Spoken Dialog Systems*. Cambridge, MA: MIT Press.