Contextual Constraints based on Dialogue Models in Database Search Task for Spoken Dialogue Systems

Kazunori Komatani, Naoyuki Kanda, Tetsuya Ogata, Hiroshi G. Okuno

Graduate School of Informatics, Kyoto University
Yoshida-Hommachi, Sakyo, Kyoto 606-8501, Japan
komatani@i.kyoto-u.ac.jp

Abstract

This paper describes the incorporation of contextual information into spoken dialogue systems in the database search task. Appropriate dialogue modeling is required to manage automatic speech recognition (ASR) errors using dialogue-level information. We define two dialogue models: a model for dialogue flow and a model of structured dialogue history. The model for dialogue flow assumes dialogues in the database search task consist of only two modes. In the structured dialogue history model, query conditions are maintained as a tree structure, taking into consideration their inputted order. The constraints derived from these models are integrated by using a decision tree learning, so that the system can determine a dialogue act of the utterance and whether each content word should be accepted or rejected, even when it contains ASR errors. The experimental result showed that our method could interpret content words better than conventional one without the contextual information. Furthermore, it was also shown that our method was domain-independent because it achieved equivalent accuracy in another domain without any more training.

1. Introduction

Automatic speech recognition (ASR) errors are inherent to speech interface. Appropriate interaction such as confirmations or disambiguating questions can keep spoken dialogue systems from fatal failure caused by such errors. In generating efficient confirmation, confidence measures of ASR were effective [1], which were derived from single utterances. We address obtaining useful constraints to interpret ASR results in the above dialogue model. Furthermore, we create a model to maintain query conditions as a tree structure, which can be used as a weight between attributes of query conditions. By integrating information derived from these models using a decision tree, we address obtaining useful constraints to interpret ASR results in a database search task without depending on any specific domains.

2. Models in Database Search Task

Araki et al. [4] categorized dialogues in task-oriented spoken dialogue systems into three abstract tasks based on the direction of information flow. The abstract tasks consist of slot filling, database search, and explanation. Our task corresponds to the second one, in which information necessary to achieve the task is offered to each other. Spoken dialogue systems that retrieve information from a relational database, which is one of the most popular back-ends, can be categorized here. In this abstract task, users often determine their next utterance according to the system’s preceding response, because users cannot know whether (or how many) there are entries satisfying their request in the target database until the system tells them. The characteristics of this abstract task are summarized as follows.

• Requisite attributes are different from users
• User’s final goal may change according to intermediate results of queries.

According to these interactive properties, it is impossible to describe a dialogue procedure beforehand as a system-initiated dialogue, so we need to adopt mixed-initiated one.

The target database of this abstract task is an ordinary relational database, which has attribute-value pairs for each entry and has a key attribute whose value is unique in the database. Hereafter, we set a restaurant database as a query target. The key attribute in this database is the restaurant name. Note that our method models the database search task, but never depends on its domain, that is, content of the database.

2.1. Model for Dialogue Flow

Our modeling of a typical dialogue flow in the database search task follows. First, users gradually specify (or retract) query conditions they have thought of, and narrow down entries that meet their request1, into several ones2. Then, users put some questions for the individual entries specified in the preceding interactions, and obtain information from the database. We

---

1This process cannot be realized by a system-initiated manner, because the order of preferred attributes is different from users.
2We assume this number of entries should be a limit the system can read out.
We introduce contextual constraints based on our models. An overview of our system is described in Figure 3. The system mainly consists of an automatic speech recognizer (ASR), a language-understanding module, and a dialogue manager. The language-understanding module determines a dialogue act and whether content words of utterances should be accepted or not, using both ASR outputs and context. A dialogue manager changes dialogue states using the results of the language understanding, and generates responses by accessing a database based on the states. We will describe details of the language-understanding module in the following sections.

3. Language Understanding using Contextual Constraint

We calculate similarities between an ASR result and template sentences, which were manually assigned to each dialogue act beforehand. These similarities represent how an ASR result as a sentence resembles the prepared sentences having each dialogue act. The template sentences amount to 590, for example, “Please tell me the address of Dining KUSUNOKI.” Here, the <TYPE> denotes values for <TYPE> attribute in the target database.

We treat 7 dialogue acts in our language-understanding module. Two dialogue acts such as specify conditions and request information correspond to the two modes in the model of dialogue flow: “specifying query conditions” and “requesting detailed information”, which constitute the main part of dialogues in the database search task. We add 5 supplemental dialogue acts, such as delete conditions, yes, no, clear all conditions, and undo. The content words are defined as all attribute names and their values in the target relational database.

If the similarity for either yes, no, clear all condition, or undo, which contain no content word, is the highest, a dialogue act for the utterance is determined as them. Otherwise, either specify conditions, request information, or delete condition get the highest similarity, which include content words, the dialogue act is determined by the procedures described in the following sections.

3.2. Construction of Decision Tree Incorporating Contextual Constraint

We constructed a decision tree [5] that outputs likelihoods which dialogue act a word should be assigned to or whether a word should be rejected, for each content word. In the training phrase, correct dialogue acts were given to each content word by hand, and a reject label was given to speech recognition errors. The influence due to recognition errors can be avoided.
3.3. Determining Dialogue Act and Selecting Content Words

The language-understanding module determines the dialogue act as a whole utterance by integrating every output of the decision tree for content words. Then, it selects each content word based on both the determined dialogue act and the likelihood that the word should be rejected.

First, likelihoods $CF(s|F_i, w_j)$ of a dialogue act $s$ for each content word $w_j$ are obtained by the decision tree of features for the decision tree. The likelihoods are summed up for every content word, and the dialogue act $S_i$ having the largest sum total is determined as that of the utterance $i$. Here, $s \in \{\text{specify condition, require information, delete condition}\}$, which have content words.

$$S_i = \arg \max_s \sum_j CF(s|F_i, w_j)$$

Then, we also calculate likelihoods $R_j = CF(\text{reject}|F_i, w_j)$, which represents the word should be rejected, for each content word $w_j$. Then, each content word is accepted if the likelihood of the determined dialogue act $CF(S_i|F_i, w_j)$ is greater than $R_j$. Otherwise, it is rejected.

We use two groups of features in classification: features obtained from a single utterance, we adopt the results of the matching (S1, S2, S3, S4) and a confidence measure of ASR (S6). On the other hand, modes of the current utterance are based on the dialogue flow model. C8 denotes the rate of entries that have ever been checked since entering current mode.
4.2. Evaluation of Language Understanding Module

We compared the following three methods to evaluate our language-understanding module with the proposed models.

**Method #1** A baseline method, in which a dialogue act having the highest likelihood in the matching was simply adopted, and content words whose confidence measures of the ASR were less than a threshold were rejected. No decision tree learning was used.

**Method #2** Decision tree learning based on the proposed method without contextual features shown in Figure 5.

**Method #3** The proposed method with all features.

In method #1, we had optimized the threshold for the confidence measure of the ASR, and set it to 0.05, which minimized errors. In methods #2 and #3, the experiments were carried out with subject-open 10-fold cross validation, that is, utterances of 2 subjects out of 20 were used as the test data, and the remainder was used as the training data. The process was repeated 10 times, and the average of the accuracies was computed. We also optimized the pruning parameter in constructing a decision tree to minimize errors. The classification accuracy of the language-understanding module per content word is shown in Table 1.

In method #3, the accuracy was improved by 3.3%, which was owed to the incorporation of contextual features. This improvement was notable in `reject`. Although the first split of the decision tree in method #3 was made by “dialogue act having the highest likelihood” as results of the matching, features derived from the proposed dialogue model played an important role in the remaining part of the tree. For classifying content words corresponding to values in relational databases, these were C1 and features based on the structured dialogue history (C16-C19). For either attributes or values corresponding to the key attribute, C2, C11, and C12 were effective.

### 4.3. Application to Another Domain

We also apply our method to a system retrieving a hotel database. The number of entries was 2,004. The key attribute was the hotel name, and there were 6 other attributes, such as type, location, equipment, and so on. We also collected dialogue data from 10 subjects, and obtained 1,271 utterances, which contained 1,426 content words. The average word accuracy of the ASR was 85.8%. We added another condition to verify domain-independency.

**Method #4** Decision tree trained with data in another domain was used. Here, decision tree was trained with the restaurant data, and was evaluated by the hotel data.

The accuracy of the language-understanding module for the hotel system is shown in Table 2.

<table>
<thead>
<tr>
<th></th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify condition</td>
<td>0.926</td>
<td>0.907</td>
<td>0.903</td>
</tr>
<tr>
<td>Request information</td>
<td>0.946</td>
<td>0.953</td>
<td>0.946</td>
</tr>
<tr>
<td>Delete conditions</td>
<td>0.815</td>
<td>0.730</td>
<td>0.857</td>
</tr>
<tr>
<td>Reject</td>
<td>0.100</td>
<td>0.368</td>
<td>0.550</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>0.809</td>
<td>0.834</td>
<td>0.867</td>
</tr>
</tbody>
</table>

#3. Furthermore, accuracy in method #4 was nearly equivalent to that in method #3, in which both the training and the test data were in the same domain. This means contextual constraint derived from the decision tree can be applied to other domains.

### 4.4. Discussion

In the evaluation for the restaurant system (Table 1), there were still 373 interpretation errors (content words which had interpreted as incorrect dialogue acts). Note that, out of 2,803 recognized content words, 389 words had been ASR errors. This fact meant that the system managed to interpret utterances even though they contained errors. The most frequent interpretation errors were words incorrectly classified as specifying conditions, which should be `reject`. This kind of the error amounted to 133 words. One of causes of these errors was because sufficient constraints cannot be obtained in initial parts of dialogues.

### 5. Conclusion

We incorporated contextual information by defining new models in the database search task. The models are not dependent on the specific domain. Dialogue modeling that does not depend on specific domains and is informative enough to screen out errors is required to complement statistical information because of inherent difficulty in collecting sufficient amounts of dialogue data. We showed an example of dialogue modeling that works as useful constraints.

### 6. References


