ABSTRACT

Domain-specific dialogue system is an important and also commercial-practicable application of speech recognition technique, and it is very helpful to decrease the search space in the aspects of accuracy improvement and search time reduction in speech recognition. Adequate use of dialogue-state-dependent language models in dialogue systems can decrease the search space greatly if a reasonable prediction of the dialogue states is feasible, and will make a dialogue system more robust in real practice. This paper presents a novel method of selecting different rule-based sub-language-models based on dialogue states to decrease the search space, which will select an adequate rule-based sub-language model in different conversation step according to the context. Experiments show that it is simple and effective in improving accuracy and recognition speed, and will be very useful in small and medium task domain.

1. INTRODUCTION

Now speech recognition technique is maturing rapidly. There are more and more applications using this technology to solve the real problems, such as meeting scheduler and email checking etc. But in practice, some key difficulties prevent dialogue system from widely use. One of the key problems is that the search space is often too huge that will make the system response very slow; on the other hand, huge search space will also make the system accuracy lower. If the system search space can be reduced, it will improve the system performance.

In a spoken dialog system, speech recognition and understanding can be improved by using contextual knowledge as an additional constraint during recognition process. In practice, with the process of the conversation, the range of the conversation will often become more and more narrow. If a smaller but adequate language model can be selected in different step based on the context with the process of conversation, the search space will become more and more smaller, and obviously the system recognition speed and accuracy will be improved more and more[2].

There are many papers have discussed various methods to decrease the search space by using the heuristic information based on dialog state [4][5]. The basic idea is that the sub-statistical language model for each dialogue state is trained separately, and the responding sub-statistical language model is selected as the system language model according the current dialogue state. One drawback of these approaches is that, it is often very difficult to collect enough corpus of a dialogue state, especially in a domain-specific task, and with the very limited amount of training material, the trained statistical language models are often not robust, which will make the system performance bad[2].

Generally, there are two kinds of language models in various speech application systems: statistical language model and rule-based language model. LVCSR system often use statistical language model to describe language knowledge; but in some small or medium speech application systems, rule-based language model may be more adequate to describe language knowledge, and often more accurate than the normal one. For example, a finite state word network can be used to explicitly constrain the recognition process; moreover, this approach also simplifies subsequent interpretation since the important parts of the recognition network can be tagged so that the semantic function of the recognized words can be output along with their identity. [1]

This paper presents a novel method which divides the task search space into several sub-search space according to the dialogue state, and builds a rule-based language model for each sub-search space, then use the corresponding sub-language model as the system language model in different dialogue state to improve the system performance.

We briefly illustrate the system architecture in section 2, and discuss the task division and the definition of dialogue state based on task structure in section 3. In section 4, we show some experimental results on using the proposed dialogue state dependent rule-based language model for a specific task domain-meeting scheduler and give the summary and future plan in section 5.
2. ORIGIN OF THE IDEA

Spoken dialogue system combines speech recognition with natural language understanding, language generation, and dialogue management. It engages the user with a multi-utterance conversation in order to complete a task.

Figure 1 shows the architecture of our spoken dialogue system. In the architecture, we integrate speech recognition module with language understanding module to form the speech understanding module. We use finite state grammar to describe the syntax structure of user’s utterances for a given task domain; use tags being attached with finite state grammar to represent the meaning of information pieces of user’s utterances. We can extend the finite state grammar with attached meaning tags to recursive transition network with each node being attached with being attached with meaning tags. We call this as a rule based language model. The speech understanding module use the rule based language model as the searching space, and output directly the meaning of a user utterance (a tag string). We call it as a semantic parse of a user’s utterance.

The information extractor module is used to extract information from a semantic parse to a slot list that is an internal expression of dialogue management module based on a task space for a given task. The task space can be defined by a spoken dialogue developer, which will be discussed in detail in next section.

The dialogue management module will be instantiated by the task space definition for a give task domain, and be adapted to the task domain. The dialogue management will handle some general “conversational skills”, such as prompting for missing information, clarifying ambiguous information, intention interpretation, handling errors and mutually grounding between human and computer based on confidence measure. In one word, dialogue management can know the current state of the current dialogue process; can determine what the system will prompt to user, and can predict what the user may probably say.

![Figure 1. Common Dialogue Application Architecture](image1)

Figure 1. Common Dialogue Application Architecture

In Figure 1, what we want to emphasize is that the dialogue management module can pass the current dialogue state constraint to speech understanding module. This means that dialogue management module can guide the speech understanding module based on the current process of the dialogue. In detail, the dialogue management module can provide the speech understanding module the dialogue state constraint information. The speech understanding module can use this information to generate the corresponding rule based language model that is a more constrained searching space in general, and then to improve the accuracy and speed of speech understanding.

3. DEFINITION OF DIALOGUE STATE

In this section, we will define the dialogue state. The dialogue state is derived from task space. Here it is defined as dialogue state can be seen as the completion status of a task. So the dialogue state can be derived from task space for a given task domain. When the task space is determined, the dialogue states can be determined.

3.1 Task space

When a spoken dialogue developer starts to develop a spoken dialogue application for a given task domain, he/she must want to solve certain problem. So at the beginning, the developer user must define the whole problems: what he/she wants to solve; we call this as task space. In our dialogue system, we use a tree structure as showed in Figure 2 to describe the problem or task space from dialogue management module’s view [3].

We use the concept of Frame to describe one problem, each frame consists of many slots; each slot describes one aspect of the problem. Each slot consists of many keys, which describe in detail the information we need in order to solve the task. Each key has a tag cluster.

From Figure 2, we can get the following observations:

1. From DM module’s view, a tree structure was used to describe the task space for a given task domain, every slot in this tree structure is loosely coupled with each other.

2. The whole task can be divided into some sub tasks, in different layers.

![Figure 2. Problem Description From DM Module’s View](image2)

3.2 Dialogue State

From above discussion, we can see that state can be defined based on the task space. Our dialogue management module uses form-filling schema, that is, at the beginning of the dialogue, all slots are
unfilled, and as the dialogue moves on, slots are filled based on the interactions between the DM and user. We can use the un-filled slots as the representation of dialogue state. From another point of view, dialogue state can also indicate the completion status of the task.

Let’s take the meeting scheduler domain as an example. Figure 3 illustrates the task space. We can see the whole task can be divided into three parts: Time, Attendee and Location, and all of these parts are described in task space definition file.

When user speaks out an utterance and the semantic slot embedded in the utterance is understood by the speech understanding engine, the slot is filled after it is grounded by dialogue management module and user. At the moment, the filled slot changes the completion status of the task, and then drives the dialogue state transition from previous one to the currently predicted dialogue state according to the current completion status of the task with the filled slot being ignored. At this moment the dialogue management module will send the predicted dialogue state constraint to the speech understanding module, and the speech understanding module will generate a rule-based language model dynamically according to the predicted state constraint. In this way, we can reduce the search paths and improve the accuracy of speed of speech understanding module.

To illustrate the whole process, let’s see the following example:

When user speaks out a task with the filled slot being ignored. At this moment the dialogue management module will send the predicted dialogue state constraint to the speech understanding module, and the speech understanding module will generate a rule-based language model dynamically according to the predicted state constraint. In this way, we can reduce the search paths and improve the accuracy of speed of speech understanding module.

With this definition of dialogue state, we have implemented a method to generate rule-based language model dynamically, which a searching net generated by a sub finite state grammar with attached tags that is corresponding to current dialogue state constraint. We have split the task space based on the number of slot. From figure 4, we can see that in meeting scheduler domain, there are three slots, ten states and seven rule-based language models.

4. EXPERIMENTAL RESULTS

In order to evaluate the speed and accuracy of our method, we have done several experiments using some corpus. Experiment data come from the telephone record of meeting scheduler domain. The corpus covers all kinds of questions that our slots define, such as the utterance asking attendee and time, the utterance asking time and location, and so on. The number of sentences is 2263.

Table 1. The corpus of meeting scheduler

<table>
<thead>
<tr>
<th>Sentence Type</th>
<th>attendee time</th>
<th>location time</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of sentence</td>
<td>357</td>
<td>1008</td>
<td>35</td>
</tr>
</tbody>
</table>

Table 2 shows the experiment results.

<table>
<thead>
<tr>
<th></th>
<th>Global model</th>
<th>Small Model on DLG State</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTR</td>
<td>10.3</td>
<td>3.1</td>
</tr>
<tr>
<td>TER</td>
<td>10.5%</td>
<td>8.1%</td>
</tr>
</tbody>
</table>

Table 2. Results using different rule-based language model

From experiments, we can see some results:

1. Using the different rule-based language model, we can get good accuracy and fast speed compare with global language model.

2. In some cases, we lose the flexibility of the dialogue, since we delete the model from the parent model based on the state.
5. SUMMARY AND FUTURE PLAN

This paper presents a novel method to divide the problem space into sub-space based on the description of problem that is used in DM, and select different rule-based sub-language model dynamically based on the dialogue running state. In this way, we can improve the system speed and accuracy.

But from the above, we can find that there are some problems in the system. The major one is the flexibility. If we dynamically cut net based on dialogue states, we will lose some good features such as editing the slot. The second problem is adaptation of the rule-based language model. User has to spend much time on writing grammar and defining the dialogue states.

We plan to solve the problems using the following way:

1. Introduce weight in different rule-based language models. While searching in the model, the speech understanding engine can use the weight information to determine which path can get the optimal.
2. Use rule and statistics hybrid method to process the language model.
3. Divide task space automatically based on the task space definition file.

6. REFERENCES