Doctoral Thesis

Ranking-based Dialog Management Approaches for Non-task-oriented Conversations

Hyungjong Noh (노 형 종)

Department of Computer Science and Engineering

Pohang University of Science and Technology

2013

업무 지향 외 대화 처리를 위한 순위 결정 기반 대화 관리 방법

Ranking-based Dialog Management Approaches for Non-task-oriented Conversations

Ranking-based Dialog Management Approaches for Non-task-oriented Conversations

by

Hyungjong Noh
Department of Computer Science and Engineering
Pohang University of Science and Technology

A thesis submitted to the faculty of Pohang University of Science and Technology in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Computer Science and Engineering

Pohang, Korea

2. 27. 2013

Approved by

[illegible]

Academic Advisor

Ranking-based Dialog Management Approaches for Non-task-oriented Conversations

Hyungjong Noh

The undersigned have examined this
dissertation and hereby certify that it is worthy of acceptance for
a doctoral degree from POSTECH

02/27/2013

Committee Char	이 근 배	(Seal) [illegible]
Member	이 종 혁	(Seal) [illegible]
Member	유 환 조	(Seal) [illegible]
Member	정 홍	(Seal) [illegible]
Member	서 정 연	(Seal) [illegible]

DCSE 20065401

노형종, Hyungjong Noh
Ranking-based Dialog Management Approaches for Non-task-oriented Conversations,
업무 지향 외 대화 처리를 위한 순위결정 기반 대화 관리 방법
Department of Computer Science and Engineering, 2013, 71 p, Advisor: Gary Geunbae Lee, Text in English

ABSTRACT

This paper presents a new hybrid dialog management framework that integrates a statistical ranking algorithm into an example-based dialog management approach for chat-like dialogs. The proposed model uses ranking features that consider various aspects of dialogs, including the relative importance of speech acts, dialog history sequences, and the causal relationships among speech acts and slot-filling states. The ranking algorithm enables one to aggregate these feature scores systematically and to generate diverse system responses. Additionally, the model provides detailed feedback by analyzing the causal relationships among speech acts and predicting the user's possible intentions associated with a given dialog states. Simulated experimental results demonstrate that our approach is effective for task-oriented dialogs and chat-like dialogs. Additionally, a case study using elementary school students implies that the proposed system can be used for language learning purposes in addition to task-oriented services.

Contents

I. Introduction

1.1. Background

Because of the straightforward accessibility of natural language, spoken dialog systems (SDSs) have received attention as an attractive interface for human-computer interaction. A person can communicate with an SDS using spoken natural language and thereby cause various services to complete his or her tasks. Many mobile devices and car navigation systems utilize spoken dialog interfaces. Many SDS projects have been developed specifically for information-seeking tasks. JUPITER [1] is a famous SDS that provides weather information. Let's Go! [2] is a telephone-based public bus information system that provides access to bus route information and schedules. Some other systems have incorporated multimodal interface into SDSs. For example, the TALK project's TownInfo [3] system is a multimodal dialog system for tourist information. Recently, DIHANA project [4] proposes a dialog system framework based on client-server paradigm for distributed and modular dialog system. It is developed for robust information systems.

Although various SDS methodologies have been proposed, building an SDS is still a challenge because an intelligent SDS cannot be designed using a simple framework. An SDS involves the integration of various modules, such as speech recognition and synthesis, language understanding and generation, and dialog management (Fig. 1). When a user inputs an audio signal to an SDS, the components of the SDS interact with each other to generate an audible system response. An automatic speech recognition module receives the user’s input and produces a textual recognition hypothesis, which is captured by a spoken language understanding (SLU) module. The SLU module interprets the textual input into a semantic representation. A dialog management (DM) module generates the next system action based on the current input and observed dialog contexts. Then, a natural language generation (NLG) module generates the corresponding textual output as the system response.

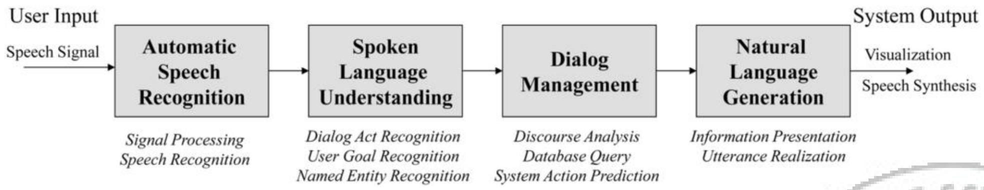

Fig. 1 Traditional architecture of spoken dialog system [5]

Although all components are important, the DM module is the most essential part of an SDS because the methodology of the DM determines the characteristics and the scope of the dialogs that can be processed. The primary role of the DM is to adopt a dialog strategy and to select the most appropriate system action. The DM uses discourse history information from previous conversations to interpret the current user's intention. Typically, the dialog state is defined using both the discourse history and the current user's intention to represent the current context. The problem is defining the dialog state effectively. Because of the exponentially large state space, we cannot consider all of the discourse information completely. Therefore, the dialog managers designed previously use approximations to reduce the dialog state space that is considered. This point will be demonstrated in Section 1.2.

Many previously developed SDSs concentrate on the robustness of dialogs because of the unreliable performance of ASR for their specific service provisions, and because robustness is essential for deploying commercial applications in the real world. Therefore, some SDSs (including the very early SDSs) were designed to use domain-specific knowledge, such as agenda, finite-state automata or hierarchical tree structures, to effectively control the dialog flow [2, 6, 7]. SDSs based on Partially Observable Markov Decision Processes (POMDPs) attempted to solve the automatic speech recognition (ASR) uncertainty problem [8]. This work proposed a POMDP-based framework that can incorporate multiple dialog states by considering the uncertainty of the ASR results. Other SDSs also have attempted to conduct dialogs robustly [9, 10]. For these task-oriented SDSs, the task completion rate (TCR) is one of the most important measures of the system’s performance. For greater performance, these task-oriented SDSs use their own policies to predict the next system action in an optimized manner.

In addition to traditional SDSs, open-domain chat-oriented dialogs have recently received significant attention because they have achieved some commercial success. They concentrate on generating fun and attractive conversations. Chatterbox Challenge [11], one of the famous contests for chat-bots, selects winners by the correctness and creativity of chat-bot responses. It is necessary to process various user inputs and to lead conversations without unnatural responses.

Many famous chat-bots, such as ALICE [12] and ELIZA [13], have been developed. These chat-oriented systems primarily provide entertainment, but there are other chat-bots like Jabberwacky [14] that offer some functional features, such as web searching or games. Recently, SIRI [15] was developed and implemented on smart phones as a personal assistant, but it also works as a chat-bot when a user inputs non-task-related utterances. Additionally, there are chat-bot systems that have been developed for education and teaching [16, 17]. Most of the previously developed chat-oriented systems are based on lexical pattern matching approaches. They search for the most similar question-response pair by computing the utterance similarity using lexical surface forms.

1.2. Previous Technologies

1.2.1. Task-oriented Dialog Management

The DM problem is still a challenging topic, and a number of different approaches have been proposed. The previously developed DM frameworks can be divided with respect to how they define dialog states and how they approximate the number of states. In this section, three representative DM approaches are reviewed.

The early dialog managers adopted knowledge-based methodologies. In such approaches, the developer designs domain-specific knowledge that is used in the dialog system to process a conversation for structured tasks. To represent the domain knowledge, finite-state automata (FSA) were widely used in early systems, such as SUNDIAL [6] or ARISE [7]. To address exception cases, some systems also used additional heuristic rules. However, this approach has the disadvantage of requiring handcrafted rules for each domain, which results in a lack of domain portability and a high cost. To solve these limitations, RavenClaw [2] used a new evolved agenda-based framework that separates the domain-independent and domain-dependent aspects of the dialog control logic. This plan-based approach has advantages for controlling the dialog flow, but still has some drawbacks. Domain expert knowledge is still required to construct the dialog plan, and the lack of flexibility is a limitation because the plan-based approach can only conduct dialogs that are guided by pre-built paths. Information-state-update (ISU) is another approach that represents dialog flow as a set of update rules. TrindiKit [18] and DIPPER [19] are based on the ISU approach. This ISU approach has relatively high dialog flow flexibility, but it also has a scalability problem when there is an excessive increase in the number of rule updates.

To overcome the limitation of knowledge-based approaches, statistical methods have been used for the DM problem. In addition to reducing the human effort necessary for the construction of the domain knowledge, statistical approaches are more robust and flexible. A representative example of a statistical approach is a DM that is based on Markov Decision Processes (MDPs) with Reinforcement Learning (RL) [20]. This approach determines the best next system action by optimizing the reward function given the current dialog state. POMDPs are used because of their robustness to ASR and SLU errors [8]. POMDP eliminated the assumption that the system knows the precise current state; instead, the system can manage parallel belief states that correspond to the n-best hypotheses of the ASR and SLU. These POMDP-based frameworks have several advantages including robustness to errors, domain independence, and dialog flexibility. However, when the state space is too large, the large policy space causes scalability problems with respect to the execution time. The lack of scalability can be a serious problem when there are many types of speech acts and entities. Therefore, there are additional studies that attempt to reduce the dialog state space and approximate the next optimal action [21, 22].

Another DM method that has been proposed is an example-based approach [23-25]. This approach performs dialog modeling using dialog examples. This approach associates each turn of the dialog example with the corresponding dialog state and indexes the next system action. To define the dialog state, various features, such as the previous system action, the slot-filling state, or lexical features (keywords) of utterances, can be used. Then, the next system action can be predicted by matching the current dialog state to the example that has the most similar dialog state. Some heuristic policies are used to calculate the similarity between the current dialog state and an example dialog state. The drawback of this approach is that, in contrast with the statistical approaches, no systematic optimization policy is used when predicting the next system action. Whereas RL provides systematic model learning frameworks, these example-based approaches rely on heuristically defined similarity measures to determine the proper example.

1.2.2. Open-domain Chatting Dialog Management

Many different kinds of chat-bots have been developed with the aim of realizing natural human-computer interaction. Most well-known previous chat-bots were rule-based systems, which retrieve example sentences by applying simple pattern matching technique. As described above, ELIZA and ALICE are those ones of the famous systems.

ELIZA was the first chat-bot and uses a simple keyword and pattern matching algorithm. In ELIZA, input sentences are analyzed using decomposition rules that are triggered by keywords that appear in the input text. Responses are generated using reassembly rules associated with the selected decomposition rules. ALICE shares the same basis of ELIZA. The transformation rules used by ALICE are written in the Artificial Intelligence Mark-up Language, which is an XML language designed for chat-bots. It defines various tags for symbolic reduction, synonyms normalization, and keyword extraction.

After the appearance of ALICE, a number of chatting systems have been developed. Chatscript [26], a chat-bot engine that is basically similar to ALICE, suggests more sophisticated features. For example, it classifies generic dialog act (DA) of the user utterance, or detects and stores some important entities or keywords like user name, etc. It uses more complicated pattern matching technique and ontology for understanding numerous intentions and topics. All of the extracted information is utilized when the system generates the system response. Another chat-bot, CSIEC [27], adopts Natural Language Markup Language as standard format of rules. It also uses large number of linguistic features ontology like Chatscript. These recently developed chat-bots have powerful features and huge potential to attract the interest of users and manage the conversation.

However, they rarely change the traditional paradigm for sentence matching mechanism using lexical surface patterns. Moreover, as these chat-bots use various features, much handwork is also needed to build the feature data. Most of the heuristic rules should be refined delicately with language expert knowledge. The experts should also have knowledge about dialog system mechanism, and iterative testing and feedback is required for maintaining the systems. The high cost of human labor is inevitable because user satisfaction for the chatting system is directly connected to the quality of rules. Without the human knowledge and manual management of the numerous rules, the quality of chatting systems cannot be maintained. Without any linguistic information, well defined heuristic rules are more highly required to expand coverage for user inputs that have wide range of domains and topics.

1.3. Problem Statement

To present our DM, which is suitable for natural and flexible dialogs, we consider chat-like natural dialogs with the following characteristics:

- The dialog participant is a novice user.
- The dialogs include colloquial small talk as well as task-oriented conversations.
- The user's utterances can include varied speech acts[1] and expressions.
- The sequences of speech acts in the dialogs are relatively diverse.

[1] We use the terminology "speech acts" to refer to both user dialog acts and system actions. Our proposed framework treats both types of speech acts in the same manner such that the system can use the same process to predict the next possible user dialog act after a system utterance is generated and the next possible system action after a user utterance is given. Therefore, we use the phrase "speech acts" unless user dialog acts and system actions are treated separately.

- Whereas a user who wants a specific service and a service provider (or agent) participates in a traditional task-oriented dialog, a chat-like dialog occurs between persons.

Here is the conversation that is an example of a chat-like dialogs (Table 1-1). The conversation includes many colloquial utterances that are not directly related to task completion. For example, utterances such as "Calm down", "Don't worry",

Table 1-1. An example of a chat-like conversation that occurs in path-finding situations

Turn	Speaker	Utterance
1	Stranger (S)	Excuse me.
	Resident (R)	What can I do for you?
2	S	I think I am totally lost.
	R	Calm down. ; Now, tell me where you were heading?
3	S	Happy Market.
	R	Don't worry. ; It's only a few blocks away from here.
4	S	Great!
	R	You know where ABC Square is, don't you?
5	S	Yes, I do.
	R	Just turn left at the West Gate of the Square, go two blocks from there and turn left. Go straight and turn left at the corner. From there, you need to walk up to High Street and turn right.
…	…	…

Table 1-2. An example of a chatting conversation

Turn	Speaker	Utterance
1	User (U)	It's too hot.
	Chatbot (C)	Try something cold!
2	U	Yeah! It's really hot.
	C	I want to drink ice water!
3	U	Me, too.
	C	Do you like sherbet with red beans?
4	U	No, I hate red beans.
	C	Well… many people don't like red beans because they taste too sweet.
…	…	…

"Great!" and "Yes, I do" are relatively colloquial in style and make the conversation more natural. These diverse expressions appear more often in human-human conversations than in human-machine conversations. The dialog patterns can be varied when these expressions are used.

We focus on three novel objectives rather than traditional objectives such as robustness, TCR and high prediction accuracy. First, we investigate the chat-like naturalness of dialogs. It is certainly very difficult to define chat-like naturalness. We define the naturalness of a dialog in terms of the number of speech act types and entity slots. Speech acts contain user dialog acts and system actions, and the types of entity slots correlate with the content database. We also consider the number of speech act bigrams or trigrams as a measure of naturalness. If more varied types of speech acts and entities and complex combinations of speech acts appear in a dialog, we regard the dialog as more chat-like and natural. Our proposed DM attempts to conduct these chat-like natural dialogs effectively. Diversity is another important point in this paper. We intend for our proposed system to conduct dialog of various patterns. Therefore, we must determine how our system can conduct more diverse dialogs. Finally, we consider some feedback and guidance for novice users. Specifically, we concentrate on helping users select the appropriate dialog acts in discourse flows. When the user inputs an improper utterance into the system, the system could provide feedback and explain to the user why the utterance cannot be processed in the discourse flow. Additionally, for each turn, the system could suggest some possible utterances to the user such that the user could continue the dialog.

We also consider a conversation that is solely for entertainment (Table 1-2). This conversation has no specific purpose, which is in contrast with other task-oriented dialogs. The users just talk about a specific topic, and the dialog is not a series of questions and answers. The dialog contains relatively various types of utterances in terms of lexical patterns.

In this paper, we argue that our proposed DM, which is an example-based DM framework with a statistical ranking policy, can conduct various chat-like dialogs as well as task-oriented dialogs. To treat many types of speech acts and entities, three different features are suggested. First, the discourse similarity score can be used to find a dialog instance that has a discourse sequence of speech acts similar to that of the current dialog. Second, the ordinal position score considers the causal relationships among speech acts. We expect that this feature can complement the function of the agenda. Third, the entity constraint score reflects the slot-filling state of the entity slots. These features are aggregated using a ranking algorithm, which enables the generation of diverse responses. Because we target conversations that include various speech acts, the relative importance of the speech acts should be considered. Therefore, we also define a measure of importance, called the Relative Importance of Speech Acts (RISA), and use this measure to calculate the discourse similarity score and the ordinal position score. Additionally, our system treats user DAs and system actions in the same manner such that the system can predict both the next system action and the proper user dialog act. This feature enables the system to assist users by suggesting the next possible utterance or providing feedback. Our system generates a dynamic help guide that contains previously defined feature scores, and this feedback is demonstrated to be useful not only for task completion but also for other purposes, such as language learning.

Especially for open-domain chatting dialog management, we suggest additional features according to the characteristics of small talks. First, we suggest Part-Of-Speech (POS)-tagged token matching to match the example sentences. POS-tagged token matching can make us set priorities and weights on each POS and then take them into account for calculating similarity of two sentences. We need to set different weights on each POS considering its general importance. This technique can work very well especially in brief spoken language. Second, named entity (NE) also has an important role in conversation. In many case they are subject or topic of conversation. This fact leads to the conclusion that they should be regarded as more important information than general words. Also some NEs are bound to specific sentence in which they are included. For example, in a sentence "*Alan Turing* was born in 1912", the meaning of the sentence is adequate only for '*Alan Turing*'. Other entities with the same type, says '*Mozart*', cannot be substituted for '*Alan Turing*'. This kind of dependency should be considered in sentence similarity calculation. Third, DA is still essential information. DA represents types of utterances categorized according to syntactic and pragmatic criteria [28]. Some sample of DA is *wh-question, statement*, *yn-question*. Suitable type of response (i.e. system action) is determined by the DA of user utterance. For instance, for the *yn-question* "Do you have apple?", the answer "Yes, I can." should be unreasonable. By adopting DA, system response would appear to be much natural, preventing such an unreasonable conversation. Even if there is no matched POS-tagged token and NE, back-off response can be selected corresponding to user DA. Responses that are generally acceptable in various situations are prepared as back-off responses and this is effective strategy to cope with any exceptional user utterances.

1.4. Motivation

We had difficulty building an SDS that could conduct some complex conversations using conventional task-oriented dialog technology. We present some information regarding the corpus used for the experiments to explain these problems. We prepared a total of four-domain corpora that can be roughly divided into traditional task-oriented dialogs and chat-like dialogs (Table 2). Two of the corpora (T1 and T2) contain task-oriented dialogs between a service agent and a customer. T1 contains conversations for car navigation, in which an agent provides information about some places to which a user wishes to go. T2 contains conversations for guidance in a building, in which an agent provides varied information to a user, such as a person’s name, room number, and telephone number and the use of a particular room. The other corpora are considered like a chat-like dialogs. One corpus (PF) contains conversations about a path-finding situation in which the participants are a stranger and a local resident. Another corpus (CT) contains conversations involving a city tour guide. We classified this conversation as a chat-like dialog although it is

Table 2-1. Basic statistics for each corpus

Corpus	Domain	# of speech acts (User dialog acts + System actions)	# of entity slots
T1	Navigation	35 (25 + 10)	9
T2	Guidance robot	51 (45 + 6)	11
PF	Path-finding	106 (39 + 67)	17
CT	City tour guide	198 (91 + 107)	26

Table 2-2. More statistical data for each corpus

Corpus	Average normalized perplexity per speech act	Average # of system actions per turn	# of combinations of multiple system actions	Ratio of speech acts that are not task-related
T1	0.064	1.404	3	9%
T2	0.035	1.048	9	7%
PF	0.354	1.712	47	29%
CT	0.103	1.474	188	11%

actually a task-oriented dialog because its statistical data have characteristics different from T1 and T2. CT has a more diverse pattern of dialogs and numerous types of user dialog acts, system actions and entities compared with T1 and T2.

Table 2-1 indicates that PF and CT have relatively many speech acts compared with the other corpora. We found that some problems are inevitable when previous DM methodologies are used for chat-like corpora. Plan-based methodologies have difficulty treating these corpora because pre-building agenda nodes requires significant human effort. Other statistical methodologies have sparseness problems when there are too many colloquial intentions. These intentions, which are not directly related to the target task, make predicting the next speech act more difficult. In addition to the sparseness problem for intentions, generating multiple speech acts for one utterance is not simple. For example, the utterance “It will be not hard. Good bye.” is composed of two speech acts; such an utterance requires an additional policy to correctly predict this speech act combination because many previous statistical DMs only predict the best speech act given the current dialog state. If we treat the two utterances as a new speech act, the sparseness problem becomes more significant.

Table 2-2 presents more statistical data that indicate the complexity of each domain corpus. We measured the normalized perplexity for each speech act according to the distribution of the bigram contexts that include the speech act. The perplexity is defined as the diversity of the context near the target speech act, such that a speech act with a perplexity of zero has the simplest contexts, and a speech act with a perplexity of one has the most complex contexts. We present the method used to calculate the normalized perplexity in detail in Section 2.2.1. The PF and CT corpora have relatively complicated dialog structures in terms of the speech acts. These domains have a relatively large number of system actions per turn and also a large number of combinations of multiple system actions. There are also many speech acts that are not task-related and difficult to treat in a chat-like corpus. We believe that these statistics necessitate a new DM framework that operates with naturalness and flexibility.

II. A Multiple Dialog States Ranking Approach

2.1. Overview of Our Example-based Ranking Approach

The DM approach we introduce is based on the example-based approaches. We start from the example-based approaches for a few reasons. First, the other approaches typically assume that dialogs have a specific purpose. Under this assumption, users try to achieve their goals and SDSs help the users. Therefore, plan-based approaches build agenda plan graphs that lead toward the goal, and statistical approaches define and optimize the reward function to be maximized as the dialog flows to the goal. However, we also consider dialogs that have no task-related goals. Because of the different point of view inherent in SDSs, we base our method on example-based DM frameworks, which do not require task-related goals. Second, the dialog corpora we use have numerous types of speech acts and entities. As described above, plan-based approaches have difficulty building the agenda graph as the number of intention nodes increases. Moreover, when intentions that are not directly related to the task are inserted into the agenda, the complexity of the agenda becomes a serious problem and these intentions can cause noise. In POMDP-based approaches, numerous types of speech acts and entities cause scalability and data sparseness problems. The state sets are defined as (S_u, A_u, S_d), where S_u is the user's state, A_u is the user's action, and S_d is the dialog history [8]. We must consider the search space for dialog states, which has a complexity of $O(n(S_u)\cdot n(A_u)\cdot n(S_d))$. As the number of types of speech acts and entities increases, the number of S_d increases exponentially. Finally, because multiple speech acts can exist in the utterances of a single turn, the data sparseness problem becomes more significant because in previous approaches, the combinations of speech acts must be treated as a new speech acts.

However, we have also noticed that the previously proposed example-based approaches alone cannot satisfy our requirements. First, the heuristic similarity metrics for these approaches are too simple to process our dialog corpora. These approaches could not address various intentions when calculating discourse similarities. We think that a principled solution is needed to process the various dialogs without using domain characteristics, and some optimization process is required when predicting the next system action. We want an SDS that can be robustly applied to many speech acts and diverse dialog patterns. The lack of consideration for user feedback and help guides is another reason for our suggestion of a new DM framework. For example, utterance suggestions for users can be useful for language learning SDSs. Even if we consider only task-oriented SDSs, user feedback is still useful because it can notify users when their utterances are incorrect given the current dialog state. Detailed feedback enables the users to complete their dialog more easily.

Because of these motivations, we suggest a hybrid framework that incorporates a statistical ranking policy in an example-based DM. To the best of our knowledge, no empirical dialog systems method that uses ranking algorithms to select the best next speech act (including user dialog acts and system actions) and that provides detailed feedback according to the user utterance error type exists. Additionally, the proposed approach is more effective than POMDP-based approaches when the dialog includes a large number of speech acts and entity slots. The proposed approach considers a dialog state search space that has a complexity of O(n(E)), where E is the set of pairs of user and system utterances in the training corpus. Generally, the complexity of dialog states of POMDP-based approaches is relatively larger than that of example-based approach. When considering the T2 corpus, n(E) is not greater than 2000, whereas n(S_d) exceeds 100000 because this number is proportional to the number of combinations of every slot value. Because of its robustness and efficiency for numerous types of speech acts and entities, our method can be valuable for conducting more flexible dialogs and for providing users with various types of feedback.

2.2. Proposed Hybrid Dialog Management Framework

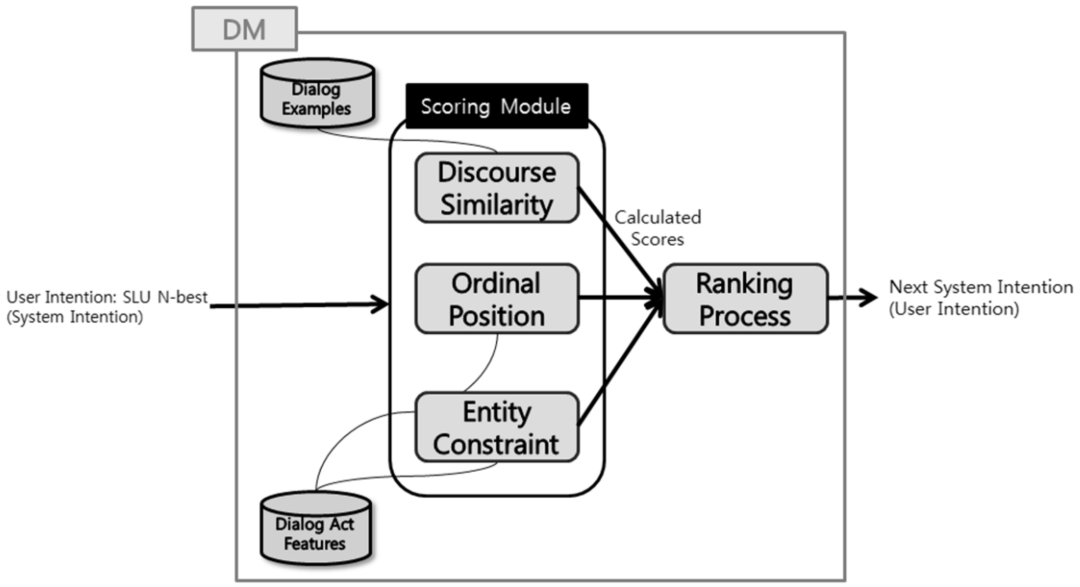

Fig. 2 Our DM framework

We propose a DM framework that uses a statistical ranking algorithm in an example-based DM (Fig. 2). To overcome the drawbacks of using heuristic metrics when selecting the best dialog example, a ranking algorithm that can aggregate various features that represent dialog states, is adopted. Three features are defined to reflect various aspects of the dialog states:

1) The discourse similarity score represents how similar the sequence of intentions of a dialog example is to that of the current dialog flow.

2) The ordinal position score measures the appropriateness of each intention given

the causal relations derived from the previous intentions.

3) The entity constraint score quantifies the suitability of each intention according to the current slot-filling state.

When a dialog example has multiple possible intentions for the next action, the second and last scores that are measured for each intention are converted to scores for the dialog example. For these calculating and converting processes, the concept of importance of intention is used. Therefore, we also defined the relative importance of a speech act according to the distributions of intention in the dialog corpus. The system predicts not only the next system action but also the next possible user dialog acts by ranking dialog examples using these feature scores.

In addition to performing the ranking process for flexible dialogs, the score analysis process yields information that is useful for providing user feedback. When it is the user's turn, the system predicts the next possible user dialog acts using the ranking algorithm. Each feature represents its own relation between the dialog state and the next intention candidate, such that the system can analyze the unexpected errors for the current dialog. With this information, each feature can be used to generate dynamic feedback that informs the user when an utterance is not appropriate. Moreover, the system can predict the next possible utterances for users, which are useful for various purposes including language learning.

Because of the additional proposed features, the indexed information of dialog examples is much larger than previous example-based approaches. Basically, every speech acts that appear in training corpus are stored separately in a database. For the speech acts, some information corresponding to the speech acts is also indexed together. The sequences of speech acts in the dialog that are used for calculating the discourse similarity score and the slot-filling states are included as the dialog state. In addition to the information about dialog examples, the statistics and data defined in Section 2.2.3. and 2.2.4 are indexed as information for each speech act.

In this section, we explain the features that are used in the ranking process and how to calculate the RISA for generating the features. Next, the process of converting feature scores for intentions into scores for dialog examples and ranking the examples using the features is described. A description of the policy of analyzing the feature scores and providing feedback follows.

2.2.1. Calculation of the Relative Importance of the Speech Acts (RISA)

We define the importance of each speech act using the perplexity concept. Perplexity is widely used in information theory to measure how good a language model is [29]. Perplexity is a measure of the size of the set of words from which the next word is chosen given that we observe the history of spoken words. This characteristic can be used for measuring the importance of speech acts. For example, a speech act *request/location* (e.g., “How can I get to the Happy Market?”) or a speech act *inform/path_explain* (e.g., “Go straight and turn right.”) should be assigned a higher importance in a discourse history for the path-finding domain, in which such speech

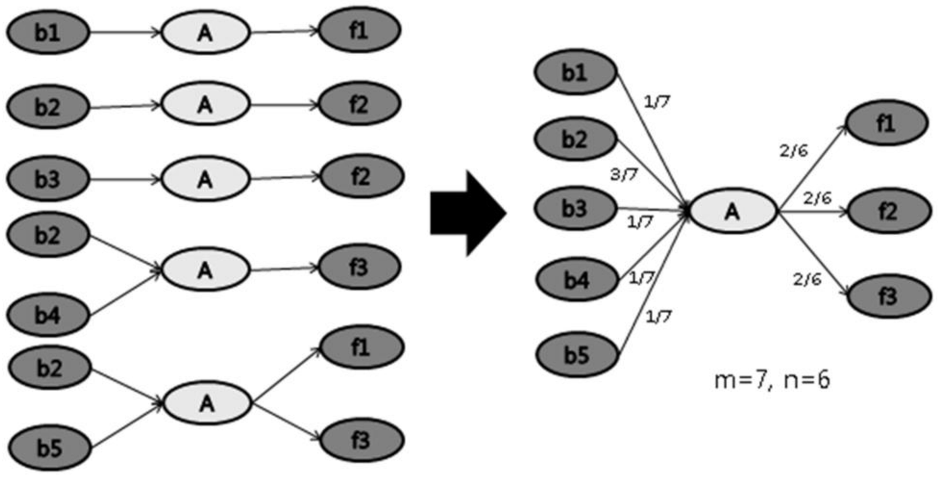

Fig. 3 Generation of the backward and forward speech act probability distributions for speech act A

acts require closely relevant speech acts to follow to maintain the coherence of the dialog. In contrast, a speech act *express/great* (e.g., “That sounds great.”) or a speech act *express/agree* (e.g., “I do think so, too.”) may have less importance than the former speech acts because relatively arbitrary speech acts can follow such speech acts. In other words, the perplexity of a speech act is inversely proportional to its importance.

Therefore, we must compute the perplexity of each speech act using the backward and forward speech act probability distributions before calculating the RISA (Fig. 3). To obtain the RISA of a target speech act A, we find all occurrences of A and its previous and next speech acts. Let us define a set I = {I_1, I_2, …, I_M} as a set of all possible speech acts without making a distinction between the user and the system. Additionally, let every occurrence of A in a training corpus be denoted A_1,

A_2, …, A_n. Each A_i corresponds to a backward speech act(s) B_i and a forward speech act(s) F_i. Multiple speech acts in a single position indicates that the speaker expresses more than one speech acts in a turn. The values for A_i, B_i, and F_i are elements of the set I. Let b be the total number of speech acts in the previous position and f be the total number of speech acts in the next position. Then, all occurrences of A_i can be aggregated into one node A, and the backward probabilities and forward probabilities are computed as

$$p_b(I_k|A) = \text{bigram\_freq}(I_k, A) / b$$

$$p_f(I_k|A) = \text{bigram\_freq}(A, I_k) / f,$$

where k=1, 2. …, M.

Given a probability distribution, the perplexity can be obtained using the equation

$$P(A) = 2^{H(p)} = 2^{-\sum_z p(z|A) \log_2 p(z|A)},$$

where H(p) is the entropy of the distribution p and z ranges over all the speech acts I.

We calculate the backward perplexity P_b and forward perplexity P_f separately. The value of P_b is in the range $[1, n_b]$, and P_f is in the range $[1, n_f]$, where n_b is the number of distinct speech acts in the previous position of A and n_f is the number of distinct speech acts in the next position of A. We can normalize and summarize the two values as

$$S(A) = (P_b(A) + P_f(A)) / (n_b + n_f).$$

Note that the value of S is in the range (0, 1] regardless of the values of n_b and n_f. Because a greater RISA value should correspond to a lower perplexity, the inverse value of S is used as the RISA,

$$RISA(A) = 1 / S(A).$$

We compute the RISAs for all of the speech acts using their previous and next speech acts. Table 3 shows the RISA values for speech acts of PF. Note that the RISA value properly reflects the impact of each speech act on the discourse history. The speech act *express/courtesy* has the greatest RISA value because, in most cases, it has *express/thank* as its previous speech act and *END*, which represents the end of a dialog, as its next speech act. This implies that the speech act *express/courtesy* strongly indicates which speech acts should be the previous and the next speech acts; therefore, the large RISA value is reasonable.

Table 3. Examples of speech acts and their RISA values

Speech Act	RISA value	Appearance Frequency
express/courtesy	60.98	315
inform/path_explain	45.84	813
END	44.30	321
express/thank	27.14	311
confirm/path	19.16	102
START	14.25	321
express/attention	13.56	43
ask/distance	12.01	77
…	…	…

2.2.2. *Discourse Similarity Feature*

To determine the dialog that is most similar to the current dialog, the discourse similarity should be considered. The definition of the discourse similarity is a primary component of our DM framework. We present the following dialog examples to illustrate the aspects that are used to define the discourse similarity.

In Table 4, the dialog example #0 is more similar to example #1 than to example #2. In the dialog examples #0 and #1, the user wants to know how to get to his destination and the system may inform the user of the path. Conversely, in example #2, the user has already learned how to get to the destination and shows his surprise. However, if we assigned greater weight to the last intention or consider the number of turns in each dialog, example #2 would be considered as a more similar example. The utterances "don't worry" in Table 4-1 and "don't you worry" in Table 4-3, which imply the same speech act, can also result in an incorrect similarity calculation. To avoid this error, the entire sequence of intentions in a dialog flow should be considered. Additionally, the intention implied by the utterances "great" or "don't worry" can be considered less important than the other intentions. Conversely, the intentions of asking for the path and the ensuing response are more important in this path-finding domain. Using a method for calculating the importance of the intentions would make it possible to score the intentions correctly.

Table 4-1. Dialog example #0

Turn	Speaker	Utterance
1	Stranger (S)	Excuse me.
	Resident (R)	What can I do for you?
2	S	I think I am totally lost.
	R	Calm down. ; Now, tell me where you were heading?
3	S	Happy Market.
	R	Don't worry. ; It's only a few blocks away from here.
4	S	That’s great!
	R	…

Table 4-2. Dialog example #1

Turn	Speaker	Utterance
1	Stranger (S)	Excuse me.
	Resident (R)	Can I help you?
2	S	I think I got lost on the way to Happy Market.
	R	Oh, dear. ; Don’t panic. ; Happy Market isn’t far from here.
3	S	How lucky!
	R	…

Table 4-3. Dialog example #2

Turn	Speaker	Utterance
1	Stranger (S)	Excuse me.
	Resident (R)	Do you need any help?
2	S	I'm afraid I am lost.
	R	Well, calm down, dear. ; I can show you the way. ; Now, tell me, where were you heading?
3	S	Happy Market.
	R	Don't you worry. ; It's only a few blocks from here. ; Just turn left at the West Gate of the Square, then you'll see a flower stand on your left. ; You can see the main gate of Happy Market at the end of High Street.
4	S	Great!
	R	…

We defined a similarity score that reflects these aspects. The Levenshtein distance [30], also known as the edit distance, is appropriate for comparing two entire speech act sequences. Each speech act is the unit for the insertion, deletion, and substitution processes. After all processes are performed on the two sequences, the path that has the lowest distance is selected; the distance represents the dissimilarity between the two sequences. There are some advantages to using the Levenshtein distance. First, we can effectively measure the similarity of two sequences that have a different number of speech acts. Second, multiple speech acts in one utterance can be compared without using an additional process. This flexibility for comparing speech act sequences is appropriate for our purpose because it permits diverse dialog patterns. Moreover, we can avoid selecting incorrect examples by calculating the distance using the importance weights for speech acts.

Additionally, we must also consider other factors when using this distance as the speech act sequence similarity metric. Generally, more recent utterances are more important than earlier utterances. Therefore, a turn weight factor is also necessary. Moreover, each speech act has its own importance. Therefore, we must modify the original Levenshtein distance by adding an intention weight and a turn weight factor. The following description formalizes how we define the modified distance.

Let the two lists L1 = $\{A_{11}, A_{12}, \ldots, A_{1m}\}$ and L2 = $\{A_{21}, A_{22}, \ldots, A_{2n}\}$ be sequences of speech acts, and also consider a set of user dialog acts U = $\{U_1, U_2, \ldots, U_u\}$ and a set of system actions S = $\{S_1, S_2, \ldots, S_s\}$. Every node A_{ij} in the lists L1 and L2 is an element of the set U+S. There exists an $(m+1)\times(n+1)$ matrix D that can be used for dynamic programming to find the distance between L1 and L2. The distance is calculated as follows:

1) Set the values of D[0, i] to RISA(A_{1i}) and the values of D[j, 0] to RISA(A_{2j}), where $0 < i \leq m$ and $0 < j \leq n$.

2) If $A_{1i} = A_{2j}$, then set the value of D[i, j] to (D[i-1, j-1] – CoincidenceCost),

   where CoincidenceCost = RISA(A_{1i}).

3) If A_{1i} != A_{2j}, then set the value of D[i, j] to min((D[i-1, j] + DeletionCost), (D[i, j-1] + InsertionCost), (D[i-1, j-1] + SubstitutionCost)),

   where DeletionCost = RISA(A_{1i}), InsertionCost = RISA(A_{2j}), and SubstitutionCost = $\sqrt{}$(RISA(A_{1i})·RISA(A_{2j})).

4) Let the value of D[m, n] be the distance between L1 and L2.

Note that RISA(A), which was used as the intention weight, is the relative importance of speech act A, which is explained in detail in Section 2.2.1. The insertion, deletion, and substitution costs are modified from those used in the original Levenshtein distance. We also added a cost for the coincident case, which is calculated when two nodes are identical. The importance of the speech acts can be considered using these costs, which are related to the RISA. A shorter distance implies a higher similarity between two lists.

To assign more importance to speech acts that appeared recently, the cost values are multiplied by a turn weight factor as follows:

When calculating the value of D[i, j], the coincidence, insertion, deletion or substitution cost is multiplied by $\sqrt{(\lambda^{i+j})}$, where λ is the discount rate parameter $(0 < \lambda \leq 1)$.

The discount rate parameter depends on the dialog domain and is determined empirically. A parameter value of 1 indicates that a speech act at the start of conversation has importance equal to that of the latest speech act.

2.2.3. Ordinal Position Feature

In addition to the discourse similarity feature, the ordinal position feature is used to quantify the characteristics of the speech acts. However, the ordinal position feature has a different focus than the discourse similarity feature. The discourse similarity is measured by comparing two sequences of speech acts; it considers the similarity between two distinct dialog sequences, the current dialog flow and the candidate dialog instance. Conversely, the ordinal position feature reflects the relationship between two speech acts in the current dialog flow. This feature measures the appropriateness of the order of the speech acts in the current dialog state regardless of the other dialog examples (Fig. 4).

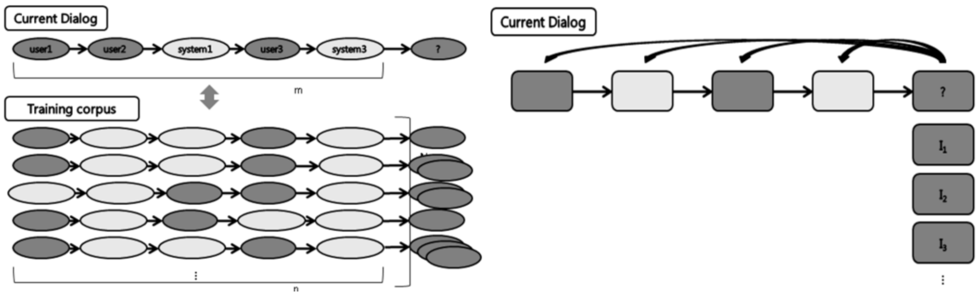

Fig. 4 Comparison of the discourse similarity feature and the ordinal position feature

Table 5 demonstrates what we want to quantify using the ordinal position feature. We focus on the causal relationships among speech acts. The stranger first gets the resident’s attention, and then requests the path to the destination. After learning how to get there, he asks for the distance and thanks the resident for the help. The flow of dialog can be considered as “(opening)-(asking location)-(asking distance)-(closing)”. If the stranger asks for the distance before the destination is confirmed or he thanks the resident before he obtains any information about the destination from the resident, it would not be a proper dialog sequence. Thus, the

Table 5. Sample conversation in path-finding situation

Turn	Speaker	Utterance
1	Stranger (S)	Excuse me. ; Can you tell me where to take a bus?
	Resident (R)	You want to go to the bus stop, don’t you?
2	S	Yes. ; Can you let me know how to get there?
	R	Just turn left at the bank, and walk along the street for three blocks. It’s next to the police station.
3	S	How far is it?
	R	It’s about one mile.
4	S	Thank you.
	R	You're welcome. [END]

ordinal relations among speech acts can be useful for predicting the next system action or for verifying the current user input. Another consideration is the relative position of each speech act in the entire dialog. In the training corpus, we have observed that two specific speech acts tend to be separated by specific intervals. For example, the system action of instructing the path to the destination is often followed immediately by the user dialog act of expressing thanks, whereas the user dialog act of getting the resident's attention tends to be separated by a significant interval from the user dialog act of expressing thanks in many conversations.

Moreover, the ordinal position feature complements Levenstein distance score in some cases. When two speech acts A_1 and A_2 can appear between speech acts A_b and A_f, both sequences (A_b-A_1-A_2-A_f) and (A_b-A_2-A_1-A_f) are possible. However, discourse similarity score would make insertion or deletion cost when the two sequences are compared. The ordinal position feature considers the causal relationship between speech acts, and the both sequences can be assigned relatively high scores. Therefore, we define the ordinal position feature to reflect these causal relationships and the relative positions in conversations such that proper candidates of speech acts can be ranked higher corresponding to the current dialog flow.

To acquire the ordinal position score for each intention, we first calculate some probabilities. We define some notations to represent the probabilities calculated using the training corpus.

For each speech act A_i, these numbers are calculated using the training corpus:

- $T(A_i)$: The average number of speech act nodes from the start of the dialog to the appearance of A_i.

- $C(A_i)$: The number of dialogs in which A_i appears.

For every possible combination of two speech acts A_i and A_j, these numbers are calculated:

- $C(A_i, A_j)$: The frequency at which A_i and A_j appear in the same dialog.

- $Pre(A_i, A_j)$: The frequency at which A_i appears before A_j in the same dialog.

If $Pre(A_i, A_j) > 0$, then we additionally calculate the interval ratio $Itv_e(A_i, A_j)$,

$$Itv_e(A_i, A_j) = T(A_i) / T(A_j).$$

The notation $Itv_c(A_i, A_j)$, which is similar to $Itv_e(A_i, A_j)$, will be used to denote the ordinal position score.

$$Itv_c(A_i, A_j) = T_c(A_i) / T_c(A_j),$$

where $T_c(A)$ denotes the number of speech act nodes from the start of the dialog to the appearance of A in the current dialog flow. Both $Itv_e(A_i, A_j)$ and $Itv_c(A_i, A_j)$ can range between 0 and 1.

After obtaining the probabilities from the training dialog examples, the ordinal position score is calculated as follows. Given the current dialog flow and the ordinal position score of speech act A_T, we consider three sets defined as follows for every speech act A_i:

- Set1: When A_i appears before A_T in the training corpus and A_i appears in the current dialog flow, A_i is added to Set1.

- Set2: When A_i appears before A_T in the training corpus and A_i does not appear in the current dialog flow, A_i is added to Set2.

- Set3: When A_i does not appear before A_T in the training corpus and A_i appears in the current dialog flow, A_i is added to Set3.

Then, three scores are defined as follows:

Score1 = $\sum_{Ai \in Set1}$ { $Pre(A_i, A_T) / C(A_i, A_T) \times$ adjusted_weight(A_i) $\times$ relative_position(A_i) }

Score2 = $-1 \times \sum_{Ai \in Set2} (Pre(A_i, A_T) / C(A_i, A_T)) \times$ adjusted_weight(A_i)

Score3 = $-1 \times \sum_{Ai \in Set3} (1 - Pre(A_i, A_T) / C(A_i, A_T)) \times$ adjusted_weight(A_i),

where adjusted_weight(A_i) = $C(A_i, A_T)^2 / (C(A_i) \times C(A_T))$ and relative_position(Ai) = $-1 \times |Itv_e(A_i, A_T) - Itv_c(A_i, A_T)| + 1$. Note that higher adjusted_weight values imply a higher likelihood of A_i and A_T appearing simultaneously, and higher relative_position(A_i) value imply that A_i and A_T are separated by similar intervals in both the training corpus and the current dialog flow.

The ordinal position score is calculated by aggregating the subscores:

$\text{OrdinalPositionScore}(A_T) = (\text{Score1} \times N(\text{Set1}) + \text{Score2} \times N(\text{Set2}) + \text{Score3} \times N(\text{Set3})) \;/\; (\sum_{A_i \in \text{Set1}} \text{adjusted\_weight}(A_i) \times N(\text{Set1}) + \sum_{A_i \in \text{Set2}} \text{adjusted\_weight}(A_i) \times N(\text{Set2}) + \sum_{A_i \in \text{Set3}} \text{adjusted\_weight}(A_i) \times N(\text{Set3}))$.

2.2.4. Entity Constraint Feature

This feature quantifies the previous slot-filling states concept and is used as one of the ranking features. The primary difference from other approaches is that the entity constraint feature uses a weighted slot vector for each speech act, whereas previous approaches use slot-filling states and a simple slot vector representation that has binary values (filled or not filled) for each slot to represent the current dialog state. We also use a binary-valued slot vector for the current slot-filling state, but each speech act has its own weighted slot vector that encodes the relationships between the speech act and each entity. By comparing the slot vector similarity, each speech act can be assigned its own entity constraint score.

Fig. 5 illustrates how the entity constraint score is calculated. Given the current dialog flow, the state is represented as a simple binary-valued slot vector. Then, the entity constraint score can be calculated for each speech act by calculating the cosine similarity of the slot vectors.

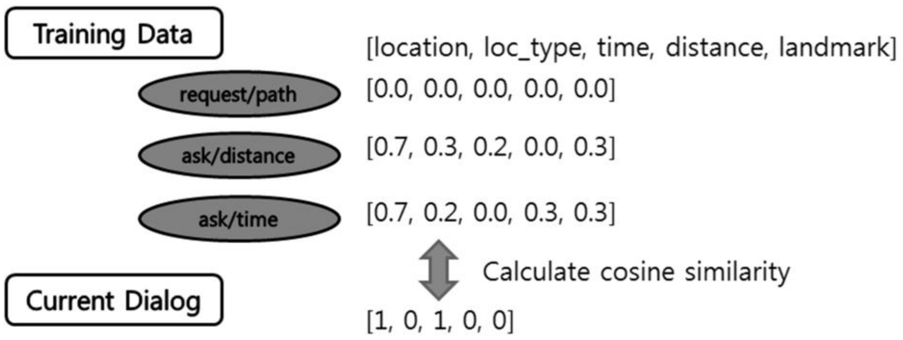

Fig 5. An example for the entity constraint feature

2.2.5. Converting Scores and Ranking Dialog Examples

The discourse similarity score is calculated for each dialog example, whereas the ordinal position score and the entity constraint score are calculated for each speech act. Because we intend to rank the dialog examples, the ordinal position and entity constraint score must be converted to scores for each dialog example. If a dialog example has only one speech act, then the score can be used directly. However, when a dialog example has more than two speech acts in a turn, we must convert the scores for each speech act into scores for each dialog example. We use the RISA as a weight value for each speech act such that dialog example E, which has multiple speech acts S_1, S_2, …, S_k, has the converted score $\sum_i (\text{Score}(S_i) \times \text{RISA}(S_i)) / \sum \text{RISA}(S_i)$.

Therefore, we assign three scores for each dialog example and use the ranking SVM algorithm [31] to aggregate the scores. The ranking SVM algorithm is one of the variants of support vector machines that are used to solve the ranking problem in information retrieval. Traditional ranking algorithms score documents according to their relevance to a given query. To adapt the ranking algorithm to the DM framework, the ranking SVM algorithm generates a prediction score for each dialog example, and then the example list is sorted according to the prediction score. The current dialog state is used instead of a query. To generate the score, the above features are used as feature scores for the ranking SVM algorithm. We determine the next speech act by selecting the corresponding dialog example using a two-phase process. First, the candidate examples are extracted using their prediction scores. If example E has a prediction score S_E, then every E satisfying "$S_E \geq$ (MAX_SCORE $\times$ threshold_rate)" is added to the candidate set C, where MAX_SCORE is the greatest prediction score from the example list and threshold_rate is the parameter value for the selection, which ranges from 0 to 1. Note that the greatest 1-best value will be chosen only if the threshold_rate is set to 1. Second, when the selection of candidates is completed, we determine the example to be used for the next speech act according to the probability value $S_E / \sum_{i \in C} S_i$. As shown in Section IV, we can control the response diversity by adjusting the value of threshold_rate.

2.2.6. Error Analysis and Feedback

The proposed features can be used for provide detailed feedback when a user inputs an improper utterance. If SLU n-best scores are provided, then the system can combine the SLU n-best scores and the score yielded by the ranking algorithm to recognize the intention [32]. According to the re-ranked scores, the system confirms that the user utterance is recognized properly. If it is not recognized properly, the discourse similarity score is used to provide some utterance suggestions to the user.

Furthermore, the system can provide the user with a specific reason why the utterance is not proper for the current dialog state. The ordinal position score has subscores for each speech act that reflect the causal relationships between two speech acts. Therefore, when the user tries to ask for the distance without identifying his destination, for example, the system can give feedback such as “You can ask for the distance after the destination is identified.” according to the subscores of the ordinal position score. This feedback would help the user to input more proper utterances.

III. Open-domain Chatting Dialog Management using POS-Tag and NE Information

3.1. Overview of Our Chatting Dialog Management Approach

In this section, we present the general architecture for open-domain chatting system, which is intended to maintain coverage for matching a user utterance to the corresponding example with considerably reduced human annotation cost and a limited corpus. To accomplish our purposes, we adapt the EBDM concept to the chat-bot problem. By adopting the advantages of the EBDM approach, we can avoid requiring significant human labor to build the system. Additionally, we suggest three features that are useful for chat-bot systems: POS-tagged tokens for sentence matching, NE types and values for searching the proper responses, and back-off response for unmatched user utterances. We can avoid the cost for main action annotation in EBDM by replacing it with POS-tagged tokens that are generated automatically.

Fig. 6 shows the general architecture and demonstrates how the processes work. In the training phase, DA and NE classifier models are trained using the training corpus. DA and NE types are defined, and every NE in training corpus is annotated by a human. Conditional random field (CRF) classifiers [33] are built for the DA and NE detection. Additionally, a POS-tagged token generator abstracts user utterances, and the results are indexed in a dialog example database. A dialog example consists of a user utterance and the corresponding system response. Whereas we annotate DA only for user utterances, NE and POS-tagged tokens are annotated for both user and system utterances.

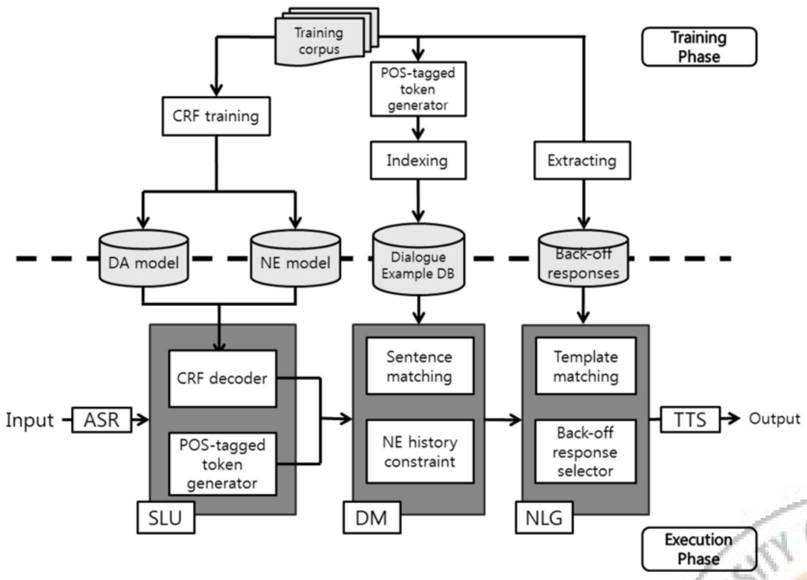

Fig 6. The open-domain chat-bot architecture

In the execution phase, user inputs are recognized by the ASR module, and the CRF classifiers and the POS-tagged token generator use the recognized result as input for the SLU module. The SLU module produces the prediction results for DA and NE, and POS-tagged tokens that are used in the DM module for predicting the most proper response. In the DM module, the example matching process finds examples that are similar to the current user utterance, and the NE history constraint process filters out the improper examples in terms of NE types and values. After the DM module selects the next system action, the surface form for the system action is generated by the NLG module using a template-based method. If the DM module cannot find the proper example, then the NLG generates the back-off response using the DA information.

3.2. Open-domain Chatting Dialog Management Framework

3.2.1. Using Dialog Acts and POS-tagged Tokens

First, we demonstrate how the annotation is performed to explain the example matching process using DAs and POS-tagged tokens. We defined DA types with domain-independent intentions and annotated approximately 3000 utterances. The annotated utterances were used to train the DA classifier. The unigram and bigram of words were used as features of the classifier. Note that we do not need to annotate DAs for all sentences in the training corpus. The DA classifier can be used for labeling the training corpus and for predicting the DA of a user utterance in the execution phase.

The POS-tagged tokens can be acquired automatically using a POS-tagger. A user utterance is tagged by the POS-tagger [34], and each token is weighted and chosen by the POS-tag. We set the POS priority using heuristic linguistic knowledge to consider the different importance of the POS tags. For example, nouns and verbs are more important than prepositions and determiners for expressing the meaning of the utterance. Additionally, we defined a higher-priority POS-tag class. According to the pre-defined priority, only higher priority POS-tagged tokens are selected and assigned weights. For example, the utterance “Would you like an apple?” has these tokens: would/VB/1.5, you/PRP/1.2, like/VBP/1.5, and apple/NN/1.5. We annotated the DAs and POS-tagged tokens of all sentences in the training corpus using the aforementioned automatic processes. The semi-supervised DA labeling and fully automatic POS-tagging processes effectively replace numerous heuristic rules that were built manually using native language knowledge in previous chat-bot approaches.

The annotated information is used when a given user utterance is matched to the corresponding examples. In this process, DAs are used to reduce the search space. For example, when a user inputs an utterance that has DA type *wh-question*, the system will not search examples that have declarative sentences. After the DA matching, the system finds more examples that are similar to the user utterance by using the POS-tagged tokens. For the POS-tagged tokens $S = (T_1, T_2, \ldots, T_n)$ of the utterance given by the real user and the tokens SE = (T_{E1}, T_{E2}, …, T_{Ex}) of the user utterance in an example, the similarity between them is calculated as follows:

$$Similarity(S, S_E) = \frac{\sum M(T_i, S_E)w(T_i) + \sum M(T_{Ei}, S)w(T_{Ei})}{\sum w(T_i) + \sum w(T_{Ei})}$$

where $M(T_i, S) \begin{cases} 1, \text{if } T_i \text{ has matching token in S} \\ 0, \text{otherwise} \end{cases}$ and $w(T_i)$ is the weight of a token T_i.

3.2.2. Using Named Entity Binding Information

We adopt the concept of NE binding in the chat-bot system for two purposes. One purpose is to prevent the system from selecting incorrect examples, and the other purpose is to increase the coverage of the scope of user utterances. We define various binding relation between the NEs and utterances, and use the relations when searching dialog examples. The terminology binding means how the NEs are tightly coupled with utterances. We will discuss each case in detail.

First, we consider the NEs that appear in the response utterance. Consider the example pair that has the user utterance "Do you know *Michael Jordan*?" and the system response "Certainly, *Michael Jordan* is great!" The term '*Michael Jordan*' can be regarded as the NE type 'athlete', and the response utterance also has that term. In this case, we annotate the term, and we would also match this example to "Do you know *Tiger Woods*?", which may result in the response "Certainly, *Tiger*

Woods is great!" In contrast, we could not use the same policy for the example pair "Who is *Bill Gates*?"-"*Bill Gates* is the boss of Microsoft.". The term '*Bill Gates*' cannot be replaced with any other person because it is highly coupled to the system utterance. We annotated these binding relations for all response utterances in the dialog examples and used this information in the execution phase.

Second, we also consider the relations between the response utterance and the NEs that appear in the user utterance. We want to consider the topic of the discourse when selecting the response utterance. Three cases can be considered for this relation and we explain each case with the corresponding example (Table 6). In example #1, the response "Negative." would be proper regardless of the NE type of "*banana*". If the term "/*banana*/food/" is replaced with "/*milk*/food/" or "/*Smith*/person/", the response utterance is still valid. Example #2 shows a different case. The response "I have been there. I love that city." would be valid only when the user utterance has the NE type "city". The response utterance is bound to the NE type. In example #3, the response "He is a famous wrestler and actor." Can only

Table 6. NE-related examples

Example number	User utterance →System utterance	Binding information
#1	Do you like /*banana*/food/? →Negative.	None
#2	Do you like /*New York*/city/? →I have been there. I love that city.	[type:city]
#3	Who is /*Hulk Hogan*/person/? →He is a famous wrestler and actor.	[value:*Hunk Hogan* type:person]

appear when the user utterance refers to the term ‘*Hulk Hogan*’ exactly. In this case, the response utterance is bound to the NE value. We annotated this binding information as discourse context information.

Using the binding relations of NEs, the system can select more highly related responses to the current topic while maintaining the coherency. If the value *V* and type T of an NE are detected from the user input, the system selects dialog examples that satisfy these conditions:

1) The examples have binding relations with the NE type T.
2) The examples do not have binding relations with any NE values except *V* that have the type T.

3.2.3. Using Back-off Responses for Corresponding Dialog Acts

We have suggested using POS-tagged tokens and NE binding information to match as many of the appropriate dialog examples as possible. However, every utterance cannot be covered by the system. When a user inputs utterances that are not matched to any examples in the training corpus, the response would not be selected properly. It is very important to determine how to generate proper responses for this case. Even if the user utterance is not matched to proper examples, an adequate back-off response makes the user think that the system understands his or her meaning. For example, a user can say “Let’s see a funny animation now!”, which is not matched to any examples. However, the DA classifier notices that the utterance has the intention of *suggestion*; then, the system could response to the user with a response such as “OK. I will follow your suggestion.” For more diverse and natural responses, we have made three responses for each DA and the system selects one of them randomly. The three responses for each DA are extracted from the training corpus. To obtain the back-off responses for each DA, the system responses were clustered using the DAs of user utterances, and the system responses that can be used for a general response to a DA were selected manually. Because we defined approximately 30 DAs, selecting less than 100 back-off responses is sufficient.

IV. Experiments

In this section, we present various empirical results to demonstrate that our assumptions are appropriate in practical tests and that our methods have distinct advantages.

4.1. Evaluations for the Multiple Dialog States Ranking Approach

4.1.1. Experimental Setup Using User Simulators

To conduct an automated evaluation of the SDSs, we implemented a user simulator that can simulate plausible utterances when a dialog state is given. Using the user simulator enables us to avoid evaluation problems with human subjects and to conduct experiments effectively under various different conditions without changing other control variables. The user simulator was implemented based on the work of [35] and includes user intention modeling using a linear-chain conditional random field (CRF), data-driven user utterance simulations, and ASR channel simulation using linguistic knowledge. When a dialog state is given, the user simulator generates a user intention based on the CRF model. The CRF model is trained using dialog history information from the training corpus. From the selected user intention, the corresponding user utterance is generated statistically according to the characteristics of the training corpus. N-gram language model is one of the characteristics of the corpus. Then some ASR errors are simulated using the ASR channel simulator. The utterance may have insertion, deletion, or substitution errors, which depend on the word error rate (WER) setting of the ASR channel simulator. When the substitution errors are generated, phoneme similarity is used to find similar words. The results presented in this section were obtained by simulating conversations with the SDSs and the user simulator. We describe a case study that uses human evaluations rather than the user simulator in Section 4.1.7.

For each experiment, 1000 conversations were simulated to acquire the TCR and average turn length (ATL). To calculate the TCR, we considered a dialog that satisfies following conditions as a completed dialog:

1) The dialog takes turns less than (average turns per dialog in training corpus) × 2.

Table 7. Basic information for training corpus

Corpus	# of total utterances	# of total dialogs	Average turns per dialog	# of content instances
T1	823	138	4.23	165
T2	1850	429	4.11	91
PF	1201	105	6.40	33
CT	1904	135	9.57	693

2) The content information that the user wants to access is provided during the conversation.

3) The speech act of giving information to the user appears properly.

The basic information for each domain corpus is provided in Table 7. We divided each corpus into a development set and a test set with ratio of 1:9. The development set was used for training the Ranking SVM algorithm, and the test set was used for simulating dialogs. To train the ranking SVM model, we annotated the combinations of dialog examples in development corpus and all speech acts as two groups. The combinations that appear in training corpus are annotated as positive samples and the combinations that do not appear in training corpus are annotated as negative samples. The positive samples are valued as 1 or 2 according to their relevance by human judgment and the negative samples are valued as 0. The ranking SVM model was trained using these samples such that the ranking algorithm predicts scores for each speech act when a dialog state is given.

4.1.2. The Impact of Each Feature Used for the Ranking Dialog Examples

We suggest using the ranking approach for selecting similar dialog examples, and three features are proposed. First, we discuss how these features can be used to improve the performance of the SDSs. To validate each feature, six systems were developed:

1) The MDP_BL system uses an MDP policy when selecting the next system action.

2) The EBDM_BL system adopts an example-based policy that does not use the proposed features.

3) The DSF system uses the discourse similarity feature to rank the dialog examples.

4) The OPF system uses the ordinal position feature to rank the dialog examples.

5) The ECF system uses the entity constraint feature to rank the dialog examples.

6) The ALL-F system uses all three features to rank the dialog examples.

The MDP_BL system was implemented based on MDP policy [36], and it was trained with Dyna algorithm [37], one of the reinforcement learning algorithms. The dialog states are defined as combinations of the last user speech act and the slot-filling status. To select the next best system action, reward function is defined according to the transaction probability. Every transaction that appears in the training corpus is rewarded with a positive score, whereas any transaction that does not appear in the training corpus is punished with a negative score. The EBDM_BL system can be considered as the lower bound baseline system because it does not use any of the features described in Section 2.2. For each system, the T1, T2, PF and CT corpora were used to investigate the performance for various dialogs. In these experiments, the systems selected only the first-ranked system action as the next action to concentrate on the effect of each feature on the performance.

Fig. 7 demonstrates how the TCR performance is improved by using each of our proposed features for each domain corpus. Generally, the performance is better for the task-oriented dialogs (T1, T2) rather than the chat-like dialogs (PF, CT) because of the different complexity of the dialogs. We expected that the discourse similarity feature is the most important feature among the three features because it

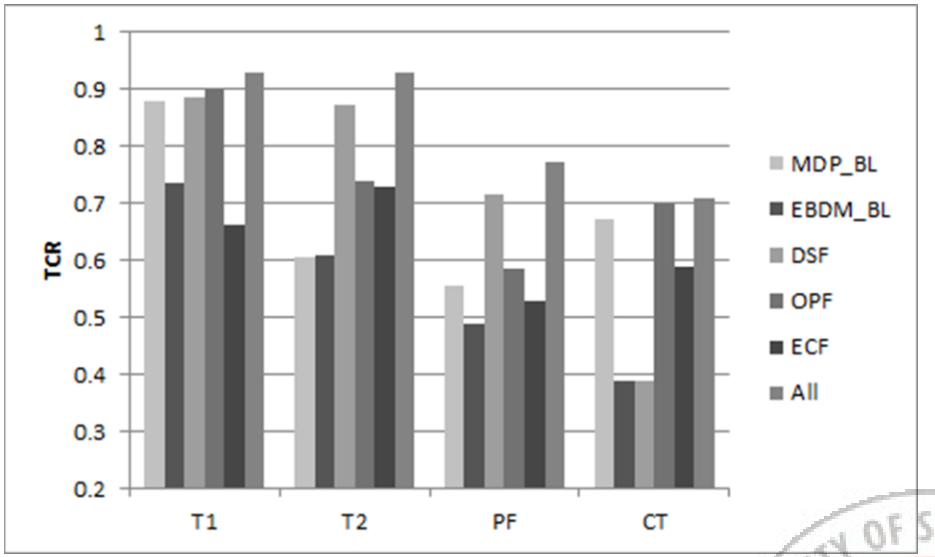

Fig. 7 Task completion rate according to the features used

ranks dialog examples directly using the entire discourse, and the results demonstrate that this expectation is true. One interesting thing is the performance of DSF for the CT corpus. For CT, the performance of OPF is absolutely higher than that of DSF. We think that too many speech acts and multiple system actions per turn caused the discourse similarity feature to be less effective. Under such conditions, using the causal relationships significantly improves the performance. Because of the different characteristics of the domains, the ranking algorithm is needed to systematically adjust the weights of the features. When the ranking algorithm is used, each feature contributes to the improvement in the performance.

4.1.3. The Impact of Using RISA Values on the Performance

Next, a verification of how the use of RISA values influences the performances is required. We defined two systems: one used the RISA values to calculate the feature scores, and the other did not. The performance was measured using TCR and ATL by changing WER. The results shown in Fig. 8 demonstrate that considering the importance of speech acts significantly increases the performance. The improvement occurs for both the task-oriented and the chat-like dialogs.

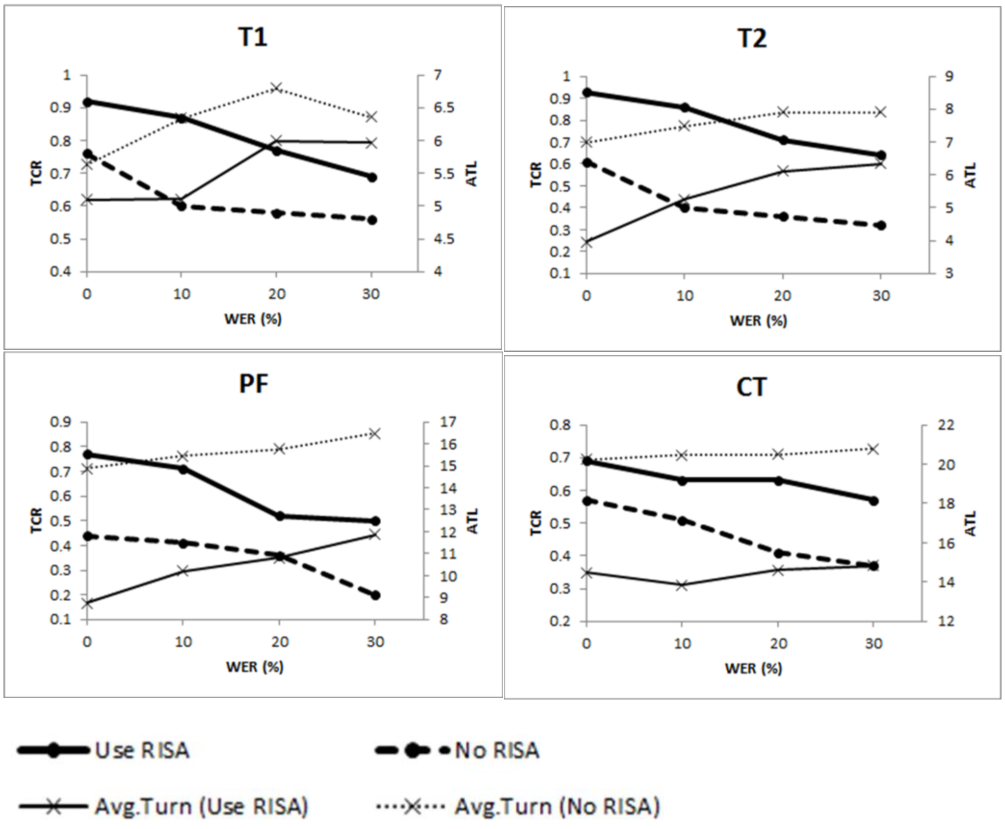

Fig. 8 The performance improvement when RISA is used

4.1.4. *Measuring the Ranking Performance*

We illustrated the performance of the proposed DM framework in terms of TCR and ATL, which are traditional metrics for task-oriented dialogs. However, we focus not only on task completion but also on the diversity of the dialog patterns. To verify the efficiency of the ranking policy, we also measured the Normalized Discounted Cumulative Gain (nDCG), which is generally used to measure the effectiveness of a web search engine algorithm. We used the previous example-based dialog

Table 8. The ranking performance for the next speech acts

System	nDCG*
Baseline system	0.764
Proposed system	0.824

* p-value < 0.0001

management system [25] as the baseline system. We extracted 300 discourse histories from the simulated dialogs and annotated the relevance of the speech acts that are predicted by the system. The relevance annotation takes values ranging from 0 to 2, where two means that the speech act is very relevant to the discourse history. The baseline system and the proposed system generated the 5-best responses for every turn, and the nDCGs were calculated for the responses. Because of the lack of annotation labor, we conducted this experiment only for the PF corpus. The results demonstrate that the ranking algorithm has significantly higher nDCG values than the baseline system does (Table 8). The higher nDCG values indicate that our algorithm chose more relevant speech acts and assigned them higher ranks.

4.1.5. The Relationship Between the Diversity of Dialogs and the Performance

The experimental results in Section 4.1.4 imply that the proper speech acts are assigned to higher ranks in the list when the suggested ranking policy is used. One of our assumptions is that when a dialog state is given, selecting relatively varied speech acts facilitates the treatment of more diverse dialog patterns. To confirm this assumption, we observed how the performance of the SDSs and the diversity of the dialogs are changed according to the selection from the ranked list of speech acts. To

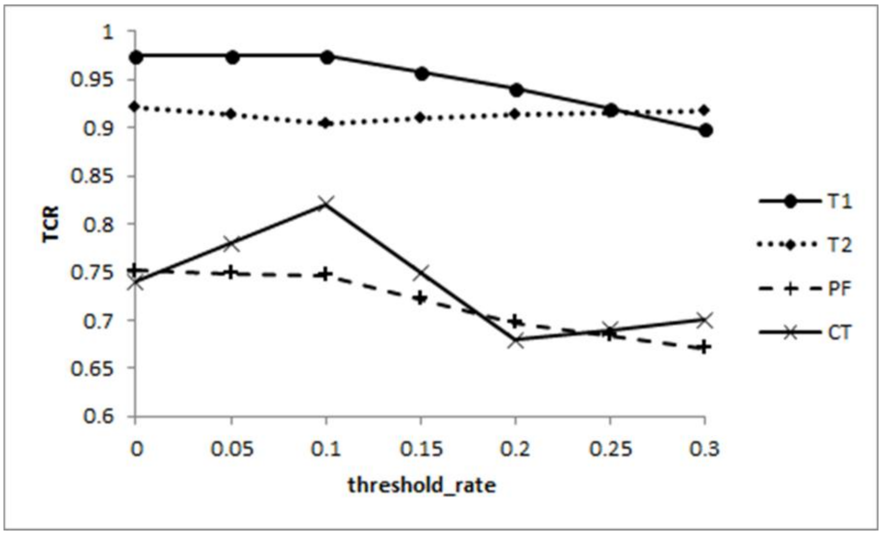

Fig. 9-1 The TCR performance versus threshold_rate

evaluate the diversity of the simulated dialogs, we measured two numbers. One measurement was the average forward perplexity per speech act, which is already used when the RISA is calculated. A greater perplexity indicates that the responses are more diverse. The other measurement used was the number of trigram sequences of speech acts. If the simulated dialogs have diverse patterns, then a large number of trigram sequences will also appear. We conducted experiments using various values of threshold_rate (See Section 2.2.5).

As we expected, the distribution of speech acts in the simulated dialogs varies as the value of threshold_rate increases (Fig. 9-2). The TCR performance slightly decreases because the system does not only select the one-best accurate next system action (Fig. 9-1). However, the CT case was somewhat different. The TCR did not decrease significantly as the dialog diversity increased, and the TCR increased when the threshold value was 10. For domains that include complicated dialogs, such as

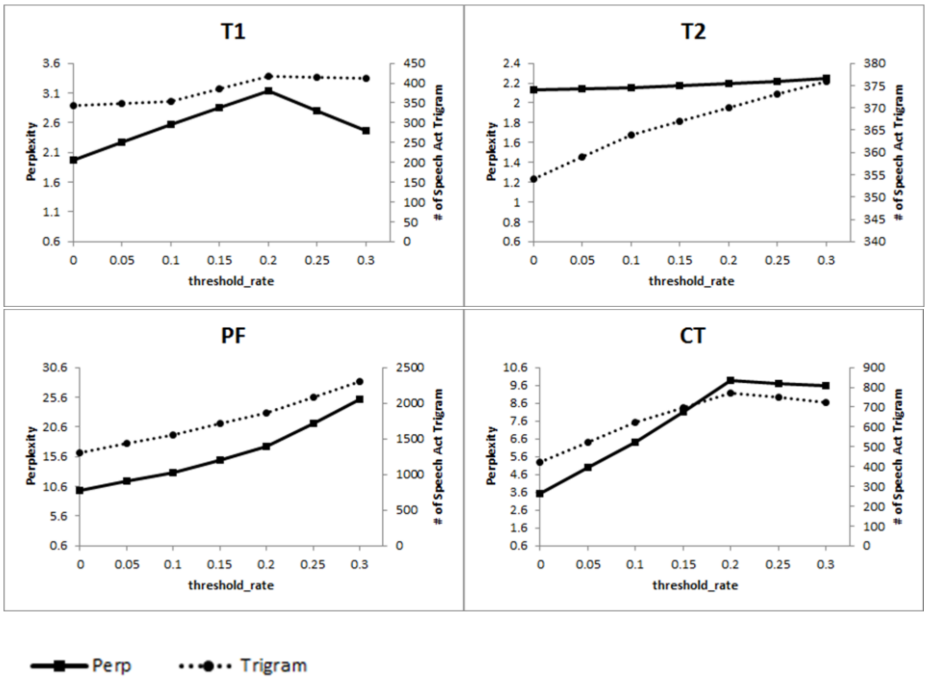

Fig. 9-2 The perplexity and number of speech act trigrams versus threshold_rate

CT, allowing a selection of more diverse system responses may be an advantage. We could set the threshold_rate value based on the purpose of the system. For the task completion purpose, the diversity of dialogs is less important and a threshold_rate value of 1 can be sufficient. However, if we consider other purposes, such as language learning or chat-bot, then we should set the threshold_rate value to balance the dialog diversity with task completion performance.

4.1.6. Providing Feedback to the User Simulator

Last, we want to verify the effectiveness of the feedback generated using the

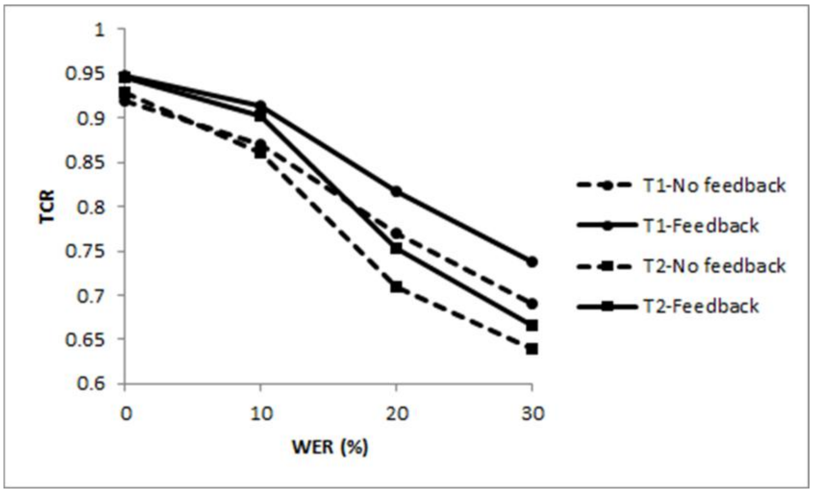

Fig. 10 The effect of the use of feedback on the TCR performance

discourse similarity and ordinal position feature scores. We used two methods of providing feedback to the user simulator when the user simulator input an improper utterance. In one method, the user did not repeat the input with the same intention as the input that was rejected by the system. In the other method, the user was more likely to input the suggested utterance with higher probability. The former method assumes that real users would use the feedback information passively, and the latter pattern assumes that they would use the information actively. Using both methods, the performance improved when the feedback was used by the user simulators (Fig. 10). The experimental results demonstrate that the feedback generated by the system increased the performance for task-oriented domains.

4.1.7. Case Study (Human Evaluation)

Evaluating a dialog management framework still poses difficult challenges. In particular, characteristics such as ease-of-use, portability, and domain-independence are very difficult to quantify [2]. Nevertheless, we attempted to present empirical observations and statistics that justify our assumptions and algorithms by using user simulators. To confirm the effectiveness of our proposed framework for real-world application, we conducted a case study using human subjects.

Previously, we developed dialog-based computer-assisted language learning (DB-CALL) systems [38, 39] that adopt example-based DM approaches [25] and modified the DM component for our proposed hybrid DM framework. We conducted a field test with 40 elementary school students using the modified DB-CALL system. The students participated in the field test for 12 days and 40 minutes per day. Various scenarios, such as path-finding, borrowing books, sending letters, and talking to a friend, were used for constructing the SDSs, and the students worked on the given tasks in a virtual 3D environment combined with the SDSs. We conducted vocabulary tests before and after the students used the SDSs. The results demonstrate that the students significantly improved their vocabularies (Table 9). Our SDSs generated diverse response utterances and suggestion utterances for the

Table 9. Results of the vocabulary test

	Pre-test		Post-test		Mean
	Mean	SD	Mean	SD	difference
Vocabulary	74.0	31.4	117.6	32.7	43.6*

* p-value < 0.0001

students, and the students could learn various expressions. The increase in vocabulary implies the effectiveness of the proposed framework for language learning, which is not task-oriented in purpose.

4.2. Evaluations for the Open-domain Chatting Dialog Management

4.2.1. Experimental setup

To verify the effects of the three proposed characteristics for the open-domain chatting dialog system, we built several versions of the systems:

1) A system that does not use the proposed policies. (The system finds similar examples using only lexical similarity). (S-None, the baseline system)
2) A system that only adopts an utterance matching policy using POS-tagged tokens. (S-POS)
3) A system that only adopts the NE features. (S-NE)
4) A system that adopts all proposed features. (S-All)

Additionally, we prepared an open-source ALICE system for comparison with our system. We built our system using the same corpus that is used for the ALICE system, such that we can compare the two systems without any biases. We extracted every utterance pair used in the ALICE system and used them as the training corpus. The training corpus was annotated to train our proposed system.

For the experiments, 8 participants (aged 20-30) were employed. They each made 30 utterances to be used as system input; therefore, total 240 utterances were used for each experiment. The participants generated utterances according to the following instructions:

1) Generate 10 free utterances.

2) Given every possible NE type, generate 10 utterances that contain at least one of the NEs.

3) Given 5 random sentences from the training corpus, generate two paraphrased utterances for each random sentence.

The utterances were input into ALICE and our systems equally, and the responses were rated from 1 to 5 according to the appropriateness for the input utterances. The correlation between the similarity scores and the ratings was 0.397, which implies that the ratings for the responses were proportional to the matching scores for the user utterances.

4.2.2. Experimental results & discussion

We designed experiments focusing on verifying two aspects of the system: how to find sentences that are similar to the user utterances using the proposed features, and how to generate the proper response while maintaining performance competitive with previously presented chat-bot systems.

Fig. 11 presents the individual ratings of the baseline and proposed systems for each participant. As expected, the S-All system had the highest performance for every participant. Conversely, S-POS or S-NE did not improve the performance compared with S-None for every participant. Thus, both POS and NE information are simultaneously needed to cover various sentence patterns of user utterances, and they are not significantly effective if used separately because using only one feature can result in incorrect example matching. Using the NE information and POS tagged token matching together; the system can overcome sparseness problems caused by many values of NEs. For example, one of the tested sentences "How can I make a *sandwich*?" does not exactly match any sentences in the training corpus. However, using the proposed utterance matching policy and NE type information, the system finds the example sentence "How do you make a *sandwich*?" as the most similar sentence, which results in the response "Bread, cheese, meat, condiments."

To analyze in detail the effects of each feature for various types of user inputs in detail, average ratings were calculated by combining the processed test sets and

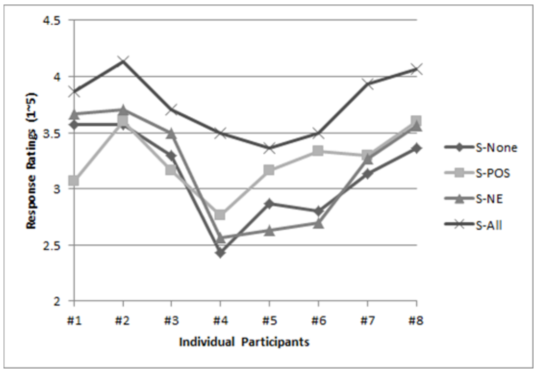

Fig 11. Ratings of various baseline and proposed systems for all

systems (Table 10). S-ALL had the most significantly improved performance compared with that of ALICE. Moreover, S-POS and S-NE also had better performance than the baseline system although the differences are not significant when each feature was applied separately. ALICE had relatively low ratings on rephrased inputs and high ratings on free inputs compared with our proposed system. The numerous heuristic patterns in ALICE yield advantages for processing many typical sentences. For example, ALICE has rules such as “WHICH ONE IS *” such that ALICE can generate proper responses even if similar sentences do not exist in the training corpus. However, many rephrased inputs cannot be processed by ALICE because ALICE cannot have all possible sentence structures as rules. S-POS and S-NE also had significantly improved performance when processing free utterances and NE-containing utterances, respectively.

Table 10. Average ratings for various systems and inputs

	Test sets			
	All (30)	Rephrased (10)	Free utterance (10)	NE-containing (10)
ALICE	3.438	3.475	3.738	3.100
S-None	3.129	3.650	3.088	2.650
S-POS	3.250	3.763	3.338*	2.650
S-NE	3.200	3.513	3.200	2.950*
S-ALL	3.754¶¶**	4.163¶¶**	3.700**	3.400**

¶: significantly different from ALICE, $p<0.05$, ¶¶: $p<0.01$
*: significantly different from S-None, $p<0.05$, **: $p<0.01$

V. Conclusions

In this paper, we have proposed a new hybrid DM framework that integrates a ranking algorithm into an example-based approach. To determine similar dialog examples, three features are defined and used in the ranking algorithm. The discourse similarity feature is used to compare entire sequences of speech acts. The ordinal position feature reflects the causal relationships among speech acts such that we can use the advantages like using agendas or planning architectures roughly. The entity constraint feature considers slot-filling states for the current dialog state and each speech act. The framework aggregates the feature scores using a ranking algorithm such that the dialog example candidates are ranked according to the prediction results. We think that the proposed features can be used for other DM frameworks as well as our example-based ranking DM. In addition to the ranking purpose, the feature scores can be used to provide user feedback. The proposed framework can predict possible user dialog acts as well as the next system action; thus the predicted user dialog act can be provided to users as a suggested utterance. Moreover, detailed feedback, which includes the causal relationship of the speech acts, can be generated using the ordinal position feature score. This feedback is useful not only for task-oriented conversations but also for other purposes such as language learning dialogs. Moreover, additional methodologies have been presented for open-domain chatting dialog. To improve the performance while minimizing the human effort required, we suggested three features: using POS-tagged token matching, using NE-related information, and using back-off responses according to the DA type. The experimental results verified that the proposed features are useful for improving the performance of the system. Although each feature alone could not improve the performance significantly compared with the baseline system, the system showed dramatically improved performance when the features were adopted simultaneously.

References

[1] Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T., and Hetherington, L. 2000. JUPITER: a telephone-based conversatioinal interface for weather information. IEEE Trans. on Speech and Audio Processing 8, 85−96.

[2] Bohus, D. AND RUDNICKY, A. I. 2009. The RavenClaw dialog management framework: Architecture and systems. Computer Speech and Language 23, 332−361.

[3] Lemon, O., Georgila, K., and Henderson, J. 2006. Evaluating effectiveness and portability of reinforcement learned dialogue strategies with real users: the TALK TownInfo evaluation. In Proc. of IEEE/ACL workshop on Spoken Language Technology (SLT).

[4] Griol, D., Torres, F., Hurtado, L., Grau, S., Garcia, F., Sanchis, E., Segarra, E.

2006. A dialog system for the DIHANA project. In: Proc. of International Conference Speech and Computer (SPECOM'06). pp. 131-136.

[5] Lee, C., Jung, S., Kim, K., Lee, D., and Lee, G. G., 2010. Recent Approaches to Dialog Management for Spoken Dialog Systems, Journal of Computing Science and Engineering,4(1), 1-22.

[6] Peckham, J. 1993. A new generation of spoken dialog systems: results and lessons from the SUNDIAL project. In Proc. of the European Conference on Speech, Communication and Technology. 33−40.

[7] Lamel, L., Rosset, S., Gauvain, J., and Nennacef, S. 1999. The LIMSI ARISE system for train travel information. In Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing. 501−504.

[8] Williams, J. D. and Young, S. 2007. Partially observable Markov decision processes for spoken dialog systems. Computer Speech and Language 21, 393−422.

[9] Pouteau, X., E. Krahmer, J. Landsbergen. 1997. Robust Spoken Dialogue Management for DriverInformation Systems. In: G. Kokkinakis, N. Fakotakis & E. Dermatis (eds.), Proceedings of Eurospeech’97, Patras, Greece, 2207-2211.

[10] Hurtado, L. F., Griol, D., Sanchis, E., and Segarra, E. 2005. A Stochastic Approach to Dialog Management. In Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding. 226−231.

[11] http://www.chatterboxchallenge.com

[12] Artificial Linguistic Internet Computer Entity (ALICE) Resources. Available at http://alice.sunlitsurf.com/alice/about.html

[13] Weizenbaum, J. 1983. ELIZA - A Computer Program For the Study of Natural Language Communication Between Man And Machine (Reprint). Communications of the ACM.

[14] Jabberwacky Resources. Available at http://www.jabberwacky.com

[15] http://www.apple.com/iphone/features/siri.html

[16] Kerly, A., Hall, P., & Bull, S. 2007. Bringing Chatbots into Education: Towards Natural Language Negotiation of Open Learner Models. Knowledge-Based Systems, 20, 177-185.

[17] Heller, B., Procter, M., Mah, D., Jewell, L., Cheung, B. 2005. Freudbot, An Investigation of Chatbot Technology in Distance Education. Centre for Psychology, Athabasca University Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications (pp. 3913-3918).

[18] Larsson, S. and Traum, D. R. 2006. Information state and dialogue management in the TRINDI dialogue move engine toolkit. Natural Language Engineering 6, 323−340.

[19] Bos, J., Klein, E., Lemon, O., and Oka, T. 2003. DIPPER: Description and Formalisation of an Information-State Update Dialogue System Architecture, in 4th SIGdial Workshop on Discourse and Dialogue, Sapporo, pp. 115–124.

[20] Singh, S., Litman, D.J., Kearns, M., Walker, M.A. 2002. Optimizing dialogue management with reinforcement leaning: experiments with the NJFun system Journal of Artificial Intelligence, 16, pp. 105–133

[21] Thomson, B., Schatzmann, J., and Young, S. 2008. Bayesian Update of Dialogue State For Robust Dialogue Systems. In Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing. 4937−4940.

[22] Kurniawati, H., Hsu, D., and Lee, W.S. 2008. SARSOP: efficient point-based POMDP planning by approximating optimally reachable belief spaces. Proceedings of Robotics: Science and Systems.

[23] Inui, M., Ebe, T., Indurkhya, B., and Kotani, Y. 2001. A Case-Based Natural Language Dialogue System using Dialogue Act. In Proc. of the IEEE International Conference on Systems, Man, and Cybernetics. 193−198.

[24] Murao, H., Kawaguchi, N., Matcubara, S., Yamaguchi, Y., and Inagaki, Y. 2003. Example-based Spoken Dialogue System using WOZ System Log. In Proc. of the SIGDIAL Workshop on Discourse and Dialogue. 140−148.

[25] Lee, C., Jung, S., Eun, J., Jeong, M., Lee, G., 2006. A situation-based dialog management using dialog examples. In: Proc. IEEE Internat. Conf. on Acoustics, Speech and Signal Processing, pp. 69–72.

[26] Wilcox, B. 2011. Beyond Façade: Pattern matching for natural language applications.

[27] Jia, J. 2009. CSIEC: A computer assisted English learning chatbot based on textual knowledge and reasoning. Knowledge-Based Systems 22 (4), pp. 249-255.

[28] Stolcke, A. and Shriberg, E. 1988. Dialogue Act Modeling for Conversational Speech. In papers from the AAAI Spring Symposium on Applying Machine Learning to Discourse Processing, pp. 98-105.

[29] Brown, P.F., Della Pieta, V.J., DeSouza, P.V. and Lai, J.C. 1992. Class-based n-gram Models of Natural Language. Computational Linguistics, vol. 18, pp. 467-479

[30] Gilleland, M. Levenshtein distance, in three flavors. http://www.merriampark.com/ld.htm

[31] Herbrich, R., Graepel, T., & Obermayer, K. 1999. Support vector learning for ordinal regression. Proceedings of ICANN 1999 (pp. 97–102).

[32] Lee, S., Lee, C., Lee, J., Noh, H., and Lee, G.G. 2010. Intention-based Corrective Feedback Generation using Context-aware Model, Proceedings of International Conference on Computer Supported Education.

[33] Lafferty, J., McCallum, A., Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proc. 18th International Conf. on Machine Learning. Morgan Kaufmann. pp. 282–289.

[34] DeRose, S. J. 1988. Grammatical category disambiguation by statistical optimization. Computational Linguistics 14(1): 31–39.

[35] Jung, S., Lee, C., Kim, K., Jeong, M. and Lee, G.G. 2009. Data-driven user simulation for automated evaluation of spoken dialog systems. Computer Speech and Language, 23(4): pp. 479-509.

[36] Puterman, M. L. 1994. Markov Decision Processes – Discrete Stochastic Dynamic Programming. John Wiley & Sons, Inc.

[37] Sutton, R. S. 1991. Planning by incremental dynamic programming. In Proceedings of the Eighth International Workshop on Machine Learning. Pp. 353-357.

[38] Lee, K., Kweon, S., Lee, S., Noh, H., Lee, J., Lee, J., Kim, H., and Lee, G.G. 2011. Effects of language learning game on Korean elementary school students. Proceedings of the SLaTE 2011 – workshop on speech and language technology in education (SLaTE).

[39] Noh, H., Lee, K., Lee, S., and Lee, G.G. 2011. POMY: a conversational virtual environment for language learning in POSTECH. Proceedings of the 12th sigdial workshop on discourse and dialog (SIGDial 2011) (demo presentation).

요 약 문

음성 대화 시스템은 자연어를 이용한 직관적이고 편리한 인터페이스 수단으로서 주목을 받고 있다. 기존에 제안되었던 대화 시스템들은 주로 서비스 제공의 목적으로 개발되었고, 그에 따라 업무 수행이나 강건한 작동에 초점을 맞추어 연구가 진행되었다. 본 논문에서는 업무 수행 외의 다양한 대화 처리를 가능케 하는 새로운 대화 관리 방법론을 제시한다. 우리가 제안하는 대화 시스템은 예제 기반 대화 관리 방법론에 통계적 순위 결정 알고리즘을 결합한 것으로, 다양한 패턴의 대화 처리를 가능케 하기 위하여 대화 관리 모듈에서 출력할 최적의 다음 행동을 구할 때 다양한 자질 점수를 이용한다. 첫째로 사용자와 시스템의 모든 화행(speech act)에 대한 상대적 중요도(RISA)를 계산하고, 이를 이용하여 대화 문맥 순서의 유사도를 구한다. 둘째로 화행간의 인과관계에 기반한 자질 점수를 이용한다. 마지막으로 슬롯 저장 상태(slot-filling state)에 따른 자질 점수를 계산한다. 이렇게 구한 자질 점수를 이용하여 순위 결정 알고리즘을 적용, 최적의 다음 행동을 구하게 된다. 이에 더하여 화행간 인과관계 점수를 이용하여 사용자가 잘못된 행동을 취할 경우 자세한 피드백의 제공 또한 가능하다. 그리고 열린 도메인에서의 다양한 잡담 대화를 효율적으로 처리하기 위하여 추가적인 방법론을 제시한다. 높은 말뭉치 제작비용을 피하면서도 다양한 발화를 처리하기 위해 POS-tag 정보와 개체명 정보를 이용한다. 또한 화행 정보를 이용한 예비 응답을 생성하여 최대한 적절한 응답을 제공하도록 한다. 우리는 여러 실험을 통하여 본 논문에서 제안한 대화 관리 방법론이 다양한 양상을 보이는 대화 처리에 효율적임을 확인할 수 있었다. 본 논문에서 제시한 다양한 자질들을 순위 결정 알고리즘을 통해 활용하여 대화 관리의 성능 향상에 도움이 됨을 확인하였다

Acknowledgements

감사의 글

2006년 대학원에 처음 입학할 때에만 해도 막연하게 인공지능이나 자연어 처리에 대한 공부를 더 해보겠다라는 정도의 생각은 있었지만, 앞날에 대한 구체적인 계획이나 확신이 없이 연구실 생활을 시작했었습니다. 다행히 교수님을 비롯해 연구실 사람들에게 많은 도움을 받을 수 있었고, 연구실 생활도 처음 걱정했던 것보다 할만했기에 통합과정 동안 나름 즐거운 대학원 생활을 했던 것 같습니다. 저 혼자만의 힘으로 지금의 제가 있을 수 없었다는 것을 알고 있기에 많은 분들에게 감사한 마음을 가지고 있습니다.

처음 대학교를 포항으로 간다고 하였을 때, 걱정되는 마음을 가지시면서도 저를 믿어주시고 허락해 주신 부모님께 감사드립니다.

그리고 연구실에 처음 들어올 때부터 졸업할 때까지 끊임없이 신경써 주시고 지도해 주신 지도 교수님께 감사드립니다. 또한 졸업 논문 작성 과정에서 아낌없는 조언을 해 주신 심사 위원회 교수님들께도 감사드립니다.

길다면 긴 연구실에서의 6~7년의 시간 동안, 실질적으로 가장 오래 같이 생활했었던 연구실 구성원들에게도 감사의 마음을 전하고 싶습니다. 생각보다 연구나 공부가 잘 진행되지 않아 대학원 진학을 선택한 것이 잘 한 것인가 고민하게 될 때마다 포기하지 않도록 도움을 아끼지 않으신 선배님들, 연구나 과제 진행에 있어서 서로 자기의 일처럼 협력해 주신 후배님들 모두 감사드립니다.

그 외에도 저에게 많은 관심 가져주시고 자기 일처럼 생각해 주셨던 모든 분들께 감사의 말씀 드립니다.

감사합니다.

Curriculum Vitae

Name: Hyungjong Noh (노형종)

Education

2001.03 ~ 2006.08	B.S. in Computer Science and Engineering, POSTECH
2006.09 ~ 2013.08	Ph.D. in Computer Science and Engineering, POSTECH

Publications

International Journals

Hyungjong Noh, Seonghan Ryu, Donghyeon Lee, Kyusong Lee, Cheongjae Lee, and Gary Geunbae Lee, An Example-based Approach to Ranking Multiple Dialog States for Flexible Dialog Management, IEEE Journal of Selected Topics in Signal Processing (JSTSP), Special Issue on Advances in Spoken Dialogue Systems and Mobile Interfaces: Theory and Applications, 6(8): 943-958, December 2012

Jonghoon Lee, Sungjin Lee, Hyungjong Noh, and Gary Geunbae Lee. Iteratively constrained selection of word alignment links from knowledge and statistics, Knowledge-Based Systems, 24(7): 1120-1130, Oct, 2011

Sungjin Lee, Jonghoon Lee, Hyungjong Noh, Kyusong Lee, Gary Geunbae Lee. Grammar error simulation for computer-assisted language learning, Knowledge-Based Systems, 24(6): 868-876, Aug, 2011

Sungjin Lee, Hyungjong Noh, Jonghoon Lee, Kyusong Lee, Gary Geunbae Lee, SeongDae Sagong, Moonsang Kim. On the effectivness of robot-assisted language learning. ReCALL: The Journal of EUROCALL, vol 23 (1), pp 25-58, Jan 2011

International Conferences/Workshops/Symposium

Sungjin Lee, Hyungjong Noh, Kyusong Lee, Gary Geunbae Lee. Grammatical error detection for corrective feedback provision in oral conversations. Proceedings of the 25th AAAI conference on artificial intelligence (AAAI-11), Aug 2011, Sanfransisco

Hyungjong Noh, Sungjin Lee, Kyusong Lee. Gary Geunbae Lee. Ranking dialog acts using discourse coherence indicator for English tutoring dialog systems. Proceedings of the 3rd international workshop on spoken dialog systems technology (IWSDS 2011), Sept 2011, Granada Spain

Kyusong Lee, Soo-Ok Kweon, Sungjin Lee, Hyungjong Noh, Jonghoon Lee, Jinsik Lee, Hae-Ri Kim, Gary Geunbae Lee. Effects of language learning game on Korean elementary school students. Proceedings of the SLaTE 2011 – workshop on speech and language technology in education (SLaTE), Aug 2011. Venice.

Hyungjong Noh, Kyusong Lee, Sungjin Lee, Gary Geunbae Lee. POMY: a conversational virtual environment for language learning in POSTECH. Proceedings of the 12th sigdial workshop on discourse and dialog (sigdial 2011), June 2011 Portland (demo presentation)

Sungjin Lee, Changgu Kim, Jonghoon Lee, Hyungjong Noh, Kyusong Lee, Gary Geunbae Lee. Affective Effects of Speech-enabled Robots for Language Learning. Proceedings of the 2010 IEEE Workshop on Spoken Language Technology (SLT 2010), Berkeley, December 2010

Sungjin Lee, Hyungjong Noh, Jonghoon Lee, Kyusong Lee, Gary Geunbae Lee. POSTECH Approaches for Dialog-based English Conversation Tutoring. Proceedings of the 2010 APSIPA annual summit and conference, special session on computer-assisted language learning (CALL) based on speech and language technology, Singapore, December 2010

Sungjin Lee, Hyungjong Noh, Jonghoon Lee, Kyusong Lee, Gary Geunbae Lee Cognitive Effects of Robot-Assisted Language Learning on Oral Skills. Proceedings of Interspeech Second Language Studies Workshop, Tokyo, September 2010.

Sungjin Lee, Cheongjae Lee, Jonghoon Lee, Hyungjong Noh, Gary Geunbae Lee. Intention-based Corrective Feedback Generation using Context-aware Model. Proceedings of the 2nd international conference on computer supported education (CSEDU 2010), Valencia, April 2010

Hyungjong Noh, Minwoo Jeong, Sungjin Lee, Jonghoon Lee, Gary Geunbae Lee. Script-Description Pair Extraction from Text Documents of English as Second Language Podcast Proceedings of the 2nd international conference on computer supported education (CSEDU 2010), Valencia, April 2010

Sungjin Lee, Hyungjong Noh, Jonghoon Lee, Gary Geunbae Lee. Importing human tutor's conversation strategy into dialog systems for language learning using fuzzy logic. Proceedings of the 1st international workshop on spoken dialog systems (IWSDS2009), Germany, December 2009

Hyungjong Noh, Cheongjae Lee, Gary Geunbae Lee. Ontology-based inference for information-seeking in natural language dialog system. Proceeding of the 6th IEEE international conference on industribal informatics (IEEE INDIN 2008), Daejeon, July 2008 (best presentation paper award)

HyungJong Noh, Jeong-Won Cha, Gary Geunbae Lee. A joint statistical model of simultaneous error correction for word spacing and spelling in Korean. Proceedings of the 47th annual meeting of the association for computational linguistics (ACL 2007), Prague, June 2007 (poster)

Domestic Conferences/Workshops/Symposium

노형종,서홍석,이규송, 이성진, 이근배. POMY: 가상환경 영어 몰입교육 대화 시스템. 2011 한국컴퓨터 종합 학술대회 (Kcc2011), 2011.6 월, 경주 (demo 시연)

이규송, 이성진, 이종훈, 노형종, 이근배. 자연어 대화기반 영어교육 게임 플랫폼. 2010 년 한국 음성학회 봄 학술 대회 논문집, 2010.5 월 KAIST

이규송.이성진, 이종훈, 노형종, 이근배. 자연어 대화 기반 몰입환경 영어교육 시스템. 제 22 회 한글 및 한국어 정보처리 학술대회 (HCLT10), 2010.10 월, 광주

노형종, 차정원, 이근배. 띄어쓰기 및 철자 오류 동시교정을 위한 통계적 모델. 제 18회 한글 및 한국어 정보처리 학술대회 (HLT06) 발표논문집, 2006, 10월, 포항공대

Patens

이규송, 노형종, 이근배, 어학학습 시스템 및 학습 방법, 대한민국 특허출원, 출원번호 10-2012-0052646, 출원일 2012 년 5 월 17 일

노형종, 이규송, 이근배, 채팅 대화 관리 시스템 및 방법, 대한민국특허출원, 출원번호 10-2012-0052647, 출원일 2012, 5, 17 일

이규송, 이성진, 노형종, 이근배, 권수옥, 외국어교육 시스템 및 방법, 및 이를 이용한 코퍼스 수집 방법, 대한민국특허출원, 출원번호 10-2011-0069047, 2011, 7 월 12 일

노형종, 이성진, 이규송, 이근배, 대화 관리 방법 및 이를 실행하는 시스템, 대한민국특허출원, 출원번호 10-2011-0016755, 2011, 2/24

이규송, 이근배, 이성진, 노형종, 이종훈, 자연어 대화 기술을 이용한 외국어 학습 게임 시스템 및 방법, 대한민국 특허 출원, 10-2010-0040014, 2010 년 4 월 29 일

노형종, 이종훈, 이성진, 이근배. 표현 및 설명추출을 위한 문서처리 장치 및 방법. 한국 특허 출원, 출 원 번 호: 10-2009-0100962, 출원일: 2009 년 10 월 23 일

이종훈, 노형종, 이성진, 이근배. 문서처리장치 및 방법. 한국 특허출원; 출원번호: 10-2009-0092234 / 출원일자: 2009.09.29

이동현, 노형종, 김석환, 이근배. 미등록어를 포함한 환경에서 오디오 및 비디오의 음성데이터 검색 방법 및 장치. 한국특허 출원 출원번호: 10-2009-0039889 / 출원일자: 2009.05.07

노형종, 김석환, 이동현, 이근배. 음성 대화 의미처리를 통한 사용자 프로파일 자동 구성 방법 및 장치, 및 그에 따른 콘텐츠 추천 방법 및 장치. 한국특허출원 출원번호: 10-2009-0046508 / 출원일자: 2009.05.2

노형종, 차정원, 이근배. 자소후보생성 및 음절 n-gram 을 이용한 띄어쓰기 및 철자 오류 동시교정 방법. 대한민국 특허출원 (출원번호 10-2006-0106635) 2006.10.31