Example-based Chat-oriented Dialogue System with Personalized Long-term Memory

Jeesoo Bang, Hyungjong Noh, Yonghee Kim, and Gary Geunbae Lee
Pohang University of Science and Technology
{jisus19, nohhj, ttti07, gblee}@postech.ac.kr

***Abstract*—This study introduces an example-based chat-oriented dialogue system with personalization framework using long-term memory. Previous representative chat-bots use simple keyword and pattern matching methodologies. To maintain the quality of systems, generating numerous heuristic rules with human labour is inevitable. The language expert knowledge is also necessary to build those rules and matching patterns. To avoid high annotation cost, example-based dialogue management is adopted for building chat-oriented dialogue system. We also propose three features: POS-tagged tokens for sentence matching, using NE types and values for searching proper responses, and using back-off responses for unmatched user utterances. Also, our system automatically collects user-related facts from user input sentences and stores the facts into a long-term memory. System responses can be modified by applying user-related facts in the long-term memory. A relevance score of a system response is proposed to select responses that include user-related fact, or frequently used responses. In several experiments, we have found that our proposed features contribute to improve the performance and our system shows competitive performance to ALICE system with the same training corpus.**

Keywords*—*Dialogue system, chatting system, conversational agent, example database, personalization

I. Introduction

Recently chat-oriented dialogue systems (or chatting systems) are getting popular as they attempt to get into daily life and achieve some commercial success. The main purpose of chatting systems is simulating the conversation of human beings [1]. This point of view might be stemmed from Turing Test, as many other dialogue systems do. Chatting systems are expected to respond to any user utterances in natural way. To meet the needs, it is necessary to process various user inputs. Also, to make users feel like they are communicating with human, personalization method can be adopted to the chatting system.

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. ELIZA [2] and ALICE [3] are those ones of the famous systems.

Rule-based chatting systems try to keep conversation going while avoiding inappropriate responses, even if they cannot find any matched pattern to the user input. To process the unmatched input, previous systems give generally acceptable response template that is built manually. However, these responses may need no logical context or understating and consider the surface level sentence form of the user utterance. The systems also use some core parts of the user utterance as remarks or references. This offers the user a feeling that the chat-bots pay enough attentions to him/her and try to follow the user's utterance. For this function, well-constructed rules also need to be pre-built.

After the appearance of ALICE, a number of chatting systems have been developed. However, they rarely change the paradigm for sentence matching mechanism of previous systems. Chatscript [4], a chat-bot engine that is basically similar to ALICE, suggests more sophisticated features. 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 [5], 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, as the chat-bots use various features, much handwork is also needed to build the feature data. To bring out maximum potential of these current systems, most of the heuristic rules should be refined delicately with language expert knowledge. To address these limitations of previous approaches, we try to adopt example-based methods for chatting dialogue system development. The basic idea of example-based dialogue management (EBDM) is that a system uses dialogue examples that are semantically indexed to a database, instead of rules or probabilistic models. EBDM framework can be applied to both chat and task-oriented dialogues [6].

Also, to build rapport with users, we adopted personalization method to chatting system. Personalization has been researched in the area of information retrieval and recommendation as personalized search [7], [8] or personalized recommendation systems [9], [10]. The user model is designed based on a specific domain model, as the study is focused on information retrieval or personalized search. However, a domain-specific user model is not suitable for chat-oriented dialog system. Therefore, we designed a new personalization framework for dialogue systems.

In this paper, we describe details of our methodology and the experiments. In section 2, we explain overall methods and our system framework. In section 3, we analyze the performance improvement of our system compared to baseline systems empirically. Finally, we conclude the paper and suggest several future work.

II. Methods

A. Example-based Chat-oriented Dialogue System

In this section, we propose an example-based chatting system. The system is intended to maintain coverage for matching user utterance to the corresponding example with considerably reduced human annotation cost and limited corpus. To accomplish our purposes, we adopted EBDM concept to the chat-bot problem. Although the EBDM methodology was originally suggested for task-oriented dialogue management, its advantages can be used for chat-bot framework, too.

The main idea of EBDM is that when a user input is given, the system conducts the dialogue by selecting the most similar situation from previously indexed dialogue examples. One of advantages of the EBDM is that heavy human labour is not required to build the system. The EBDM only needs to label dialogue act (DA) and named entity (NE) that can be annotated without any domain expert knowledge, and main action (MA) that requires domain expert knowledge. Moreover, any heuristic rules are not required and domains are not restricted to specific ones. We have found that these advantages make the EBDM more appropriate for chat-bot problem. Because the chat-bot system assumes open domain, many previous systems uses numerous heuristic rules to cover various expressions of user inputs.

To avoid this fully supervised approach, we adopt the idea of EBDM, selecting similar examples to the dialogue state. Additionally, we suggest three features that reflect the characteristics of 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 MA annotation in EBDM by replacing it with POS-tagged tokens that are generated automatically.

B. Using Dialogue 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. The DA and POS-tagged tokens are used to find the corresponding examples to a user utterance. We defined DA types with domain-independent intentions (Table 1). The DAs can be annotated manually. We annotated about 3,000 utterances for DAs, and the annotated utterances were used to train the DA classifier. The unigram and bigram of words were used as features of the classifier. The DAs are simplified from the current international standard effort (ISO 24617-2). Note that we do not need to annotate DAs for all sentences in training corpus. The DA classifier can be used for labelling training corpus as well as for predicting DA of a user utterance in execution phase.

The POS-tagged tokens can be acquired automatically using a POS-tagger. A user utterance is tagged by the POS-tagger [11], and each token is weighted and chosen by the POS-tag. We set POS priority with heuristic linguistic knowledge to consider the different importance between POS-tags. For example, nouns and verbs are more important than prepositions and determinations for expressing the meaning of the utterance. Also, we classified POS-tags into two classes broadly, higher priority class and lower priority class. According to the predefined priority, every token that belongs to the higher priority class is selected and has its own weight. For example, the utterance "Do you like an apple?" is tagged as "do/VB you/PRP like/VBP an/DT apple/NN", and the selected tokens are expressed like "do/VB/1.5 you/PRP/1.2 like/VBP/1.5 apple/NN/1.5". We annotated DAs and POS-tagged tokens of all sentences in training corpus with those automatic processes. The semi-supervised DA labelling and fully automatic POS-tagging process effectively replaces numerous heuristic rules, which are built manually with native language knowledge, in previous chat-bot approaches.

Table I. DA list and example sentences

DA type	Example
yn-q	Do you like Harry?
wh-q	How do you sound?
choice-q	Do you prefer ice cream or pizza?
feedback-p	Alright. I see.
fedback-n	No.
request	Tell me about your feeling.
suggest	Would you like to meet my friend?
statement	My job is musician.
...	...

The annotated information is used when a given user utterance matched to the corresponding examples. In this process, DA is used to reduce the search space. For example, when a user inputs an utterance that is wh-question, the system would not search examples that have declarative sentences. After the DA matching, the system finds more similar examples to the user utterance by using the POS-tagged tokens. The following steps are executed:

1) Define a set S that has POS-tagged tokens (T_1, T_2, ..., T_n) of the user utterance as elements.
2) Search examples that have all of elements of S.
3) If no example is searched, then remove the token Ti from S that has the lowest POS priority. If any examples are searched, then return the example set E and end the steps.
4) Adjust the Set S as S – {Ti}, and go back to the step 2).

With these steps, the system can find the examples that have similar user utterances. Once the examples are selected, the system calculates detailed similarities between the current user utterance and user utterances in the selected examples. For the POS-tagged tokens S = (T_1, T_2, ..., T_n) of the utterance given by the real user and the tokens S_E = (T_{E1}, T_{E2}, ..., T_{Ex}) of the user utterance in an example, the similarity between them is calculated like this:

$$\begin{aligned} &Similarity(S, S_E) \\ &= \frac{\sum M(T_i, S_E)w(T_i) + \sum M(T_{Ei}, S)w(T_{Ei})}{\sum w(T_i) + \sum w(T_{Ei})} \\ &= \frac{2\sum M(T_i, S_E)w(T_i)}{\sum w(T_i) + \sum w(T_{Ei})} \end{aligned} \quad (1)$$

The weight of each POS tag is defined with the POS priority heuristically. For example, the POS tag NN has the weight of 1.5, whereas DT has the weight of 0.5. The system chooses the example that has the highest similarity, and the corresponding SA is selected. The SA is converted into the system response with the pre-built system utterance templates.

Table II. THREE CASES OF THE RELATION BETWEEN THE NEs THAT APPEAR IN THE USER UTTERANCE AND THE RESPONSE UTTERANCE

Example number	User utterance	System response	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: *Hulk Hogan*, type: person]

- Matching Example:

1) S = (do/VBP, table/NN, have/VBP, foot/NN, on/IN, their/PRP, leg/NN) from the user utterance "Does a table have feet on their legs?"
2) Search examples that have all of elements of S, but it has no matched example; According to the POS priority and weight IN (0.8) $<$ PRP (1.2) $<$ VB (1.5) $<$ NN (1.5), remove tokens in the order on/IN $\rightarrow$ their/PRP $\rightarrow$ do/VB before any examples are searched.
3) Among the searched examples, S_E = (do/VBP, table/NN, leg/NN, have/VBP, foot/NN) from the example user utterance "Do table legs have feet?" gets highest similarity 0.88.

C. Using Named Entity Binding Information

We adopt the concept of NE binding in the chat-bot system for two purposes. One is to prevent the system from selecting wrong examples, and another is to increase the coverage of the scope of user utterances. For example, a user utterance "Who is Abraham Lincoln?" can be matched to the example user utterance "Who is your father?" that have the response "I have no father.". However, this match is not proper because the response is not coherent to the user utterance in terms of the topic "Abraham Lincoln". To avoid this, we define various binding relation between the NEs and utterances, and use the relations when searching dialogue examples. The terminology binding means how the NEs are tightly coupled with utterances. We will see each case in detail.

First, we consider the NEs that appear in the response utterance. Consider the example pair that has user utterance "Do you know Michael Jordan?" and 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 the term. In this case, we annotate that and we also would match this example to "Do you know Tiger Woods?" that may result in the response "Certainly, Tiger Woods is great!". On the other hand, we could not use the same policy for the example pair "Who is Tim Cook?"-"Tim Cook is the CEO of Apple". The term "Tim Cook" is highly coupled to the utterance "... is the CEO of Apple", so that "Tim Cook" cannot be replaced with any other person. Therefore, we annotated these binding relations to all response utterances in dialogue examples, and used this information in execution phase. To represent the relation, we annotate the previous two response utterances like this: "Certainly, /Michael Jordan/athlete/ is great!" and "/$Tim Cook$/person/ is the CEO of Apple", in which "$" means NEs that cannot be replaced.

Second, we also consider the relation between the NEs that appear in the user utterance and the response utterance. In other words, we want to consider topic discourse when selecting the response utterance. Three cases can be considered for this relation and we explain each case with the corresponding example (Table 2) . 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 "Negative" is valid. Example #2 shows the 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". If the term "/New York/city/" in user utterance is replaced with "/Birds/animal/", then the response is not coherent to the user utterance. The response utterance is bound to the NE type. In example #3, the response utterance "He is a famous wrestler and actor" only can appear when the user utterance refers to the term "Hulk Hogan" exactly. Even if the term "Hulk Hogan" is replaced with other NE that is the same type "person", the response would be not adequate. In this case, the response utterance is bound to the NE value. We annotated this binding information as context discourse information

With the two binding relations of NEs to response utterance and context discourse, the system can select more highly related responses to the current topic while maintaining the coherency. When a user inputs an utterance, SLU detects NEs that appear in the utterance. If value V and type T of NE are detected, the system selects dialogue examples that satisfy these conditions:

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

- Matching Example

1) S = (where/WRB, is/VBZ, [korea:COUNTRY]) from the user utterance "Where is Korea?"
2) Search examples that have all of elements of S;
   a) Matched with S_{E1} = (where/WRB, is/VBZ, [COUNTRY]) – "I'm not sure."
   b) Not matched with S_{E2} = (where/WRB, is/VBZ, [Egypt:COUNTRY]) – "Egypt is a country in north east Africa."
   c) Matched with S_{E3} = (where/WRB, is/VBZ, [Korea:COUNTRY]) – "Korea is on the eastern Asia."
3) Both S_{E1} and S_{E3} can be selected as they get the same similarity 1.0, and "I'm not sure" or "Korea is on the eastern Asia" is selected as an system output.

D. Using Back-off Responses for Corresponding Dialogue Acts

We suggested using POS-tagged tokens and NE binding information to match the appropriate dialogue examples as many as possible. However, every utterance cannot be covered by the system. Therefore, when a user inputs utterances that are not similar to any examples in training corpus, the response would not be selected properly. It is very important how to generate proper responses for this case when building a chat-bot system. Even if the user utterance is not matched to proper

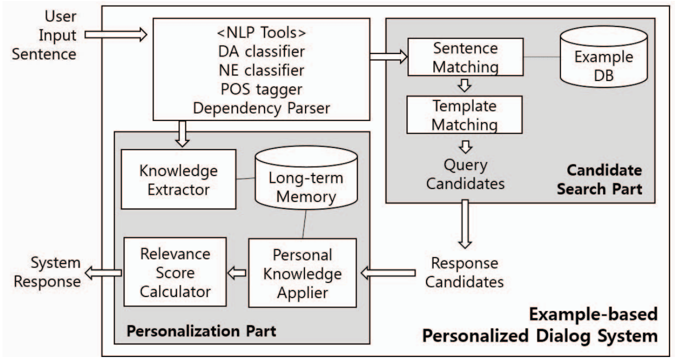

Figure 1. The architecture of the proposed system

examples, the adequate back-off response makes the user think that the system is understanding his/her meaning.

For example, a user says ”Let’s see any funny animation now!” and no examples matched to this. However, the DA classifier notice that the utterance has intention as suggestion, then the system could response to the user like ”OK. I will do at 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 training corpus. Every user utterance in training corpus has its DA, and the corresponding system responses can be clustered by the DA of user utterances. The system responses that can be used for general response of the DA are selected manually. Because we have defined about 30 DAs, selecting less than 100 back-off responses is enough.

E. Long-term Memory and Knowledge Extractor

We adopt our previous work on personalization framework [12] to our example-based chatting system. We call our personalized memory a Long-term memory, because the personalized information extracted from user input utterance is stored to the long-term memory, and is used for system response generation. We used a triple extractor to extract personalized information (knowledge units) from user input sentences; the knowledge is a RDF(resource description framework)-style triple. We define a triple as (*arg1*, *rel*, *arg2*), where the *arg*s are noun phrases and *rel* is a textual fragment that indicates the semantic relation between them. The triple extraction uses dependency tree based extraction rules as similar in [13]. All extracted knowledge units are stored in the long-term memory. An extracted triple is examined if it has user-related facts matching with manually generated triple patterns.

Figure 1 shows the architecture of example-based chatting system using long-term memory.

1) Personal Knowledge Applier: The system finds its system response candidates matching the user input utterance *u* with user utterances in the example database **E**. The system response candidates can be modified by applying user-related triples in the long-term memory. We extract a triple trp_c from the system response candidates, then convert second-person pronouns of trp_c to first-person pronouns. To find substitutable triples for trp_c from the long-term memory, two types of queries are generated from trp_c; (*, *rel*, *arg2*) and (*arg1*, *rel*, *), where * is a wildcard character. Matched triples for the queries are retrieved from the long-term memory. For each matched triples, the noun phrase which is represented as *arg* of trp_c in the system response candidate is replaced with the *arg* of a matched triple. We call this technique as triple substitution.

For example, when a user-related triple trp_u = (I, like, blue banana) in the long-term memory is applied to a system response candidate ”I know, you like apples.”, we extract trp_c = (you, like, apple) from the system response candidate. After the personal pronoun is changed, trp_c = (I, like, apple). Because trp_u matches with trp_c except their *arg2*, ”apple” in the system response is replaced with ”blue banana”. The system response is modified to ”I know, you like blue bananas”. Using this technique, our system can generate personalized response.

2) Relevance Score Calculator: Relevance score *rel(s)* measures the appropriateness of a response *s*. *rel(s)* puts weight on responses *s* which include user-related facts, and also puts weight on general, frequently-used responses. *rel(s)* is calculated using statistical information about the example database and the retentions of user-related triples in the long-term memory. *rel(s)* is designed based on the assumption that a general response has many similar responses in the example database. We adopt the relevance score introduced in our previous work [12].

III. Evaluation

A. Experimental Designs

We conducted several experiments for the following purposes:

1) Confirm the proposed utterance matching works well.
2) Confirm the NE information helps selecting the response.
3) Investigate whether our proposed chat-bot system shows competitive performance compared to previous chat-bot algorithm.

To verify the effects of three proposed characteristics on performance, we have built several versions of the systems:

1) The system that does not use proposed policies (The system finds similar examples by only using simple lexical similarity). (S-None)
2) The system that only adopts utterance matching policy with POS-tagged tokens. (S-POS)
3) The system that only adopts NE-related features. (S-NE)
4) The system that adopts back-off responses according to dialogue acts as well as utterance matching policy and NE-related features. (S-All)

Additionally, we prepared an open source ALICE system for system comparison to ours. Generally, dialogue systems, including chat-bot systems, cannot be compared directly with different domains or other environmental settings. Therefore, we have built our system with the same corpus that is used for the ALICE system, so that we can compare the two systems without any biased conditions. We extracted every utterance pairs used in ALICE system and regarded them as training corpus. The training corpus is annotated to train our proposed system.

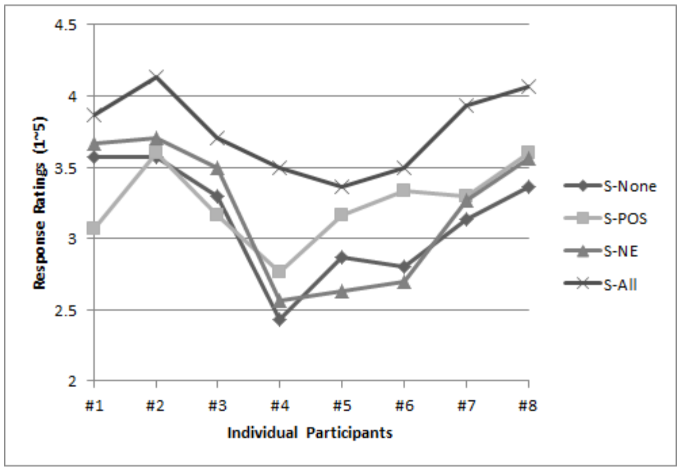

Figure 2. Ratings of various baseline and proposed systems for all participants

For the experiments, 8 participants are employed (aged 20-30). They have made their own 30 utterances as system input; therefore total 240 utterances are used for each experiment. The participants have generated utterances with the following instructions in order to test the systems with various input patterns.

1) Generate 10 utterances without any restrictions.
2) Given every possible NE types, generate 10 utterances that contain at least one of the NEs.
3) Given 5 random sentences from training corpus, generate two paraphrases for each random sentence.

The utterances were inputted into the ALICE and our three systems equally, and the responses were evaluated. All responses were rated from 1 to 5 according to the appropriateness of the input utterances. Because the ratings are evaluated by human, the ratings cannot be qualified quantitatively. To investigate whether the rating values are valuable for judging the performance of systems, we observed relation between the ratings and user utterance similarities. As we described early, the system selects the most similar example to the user input according to the similarity score. We calculated the correlation between the similarity scores and ratings from experimental results as 0.397, which means the ratings for responses are proportional to the matching score for user utterances. The positive value of correlation implies the appropriateness of the ratings as the measurement of chatting system indirectly.

B. Experimental Results & Discussion

We designed overall experiments focusing on verifying two aspects: 1) how to find similar sentences to the user utterance with proposed features, and 2) how to generate the proper response while maintaining competitive performance to previous chat-bot systems, such as ALICE. We set a baseline system, S-None, for investigating the effect of utterance matching policy, and compared S-POS, S-NE, and S-All to the baseline. The baseline system and the proposed systems generate responses for the user utterances and average ratings are calculated.

Figure 2 presents individual ratings of baseline and proposed systems for each participant. As expected, the S-All shows the highest performance for every participant with the proposed features. On the other hand, S-POS or S-NE cannot improve the performance from S-None for every participant. It can be analogized that both POS and NE information is simultaneously needed for covering various sentence patterns of user utterances, but they are not significantly effective if used separately because using only one feature can result in wrong example match.

Table III. Average ratings for various systems and input types

	All (30)	Rephrased (10)	Free uttrance (10)	NE-included (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**

For example, a user input ”Can you speak some French words?” from participant #1 matched to the example sentence ”Can you learn new words?” in the system S POS because the word ”French” was not used properly as NE information. Another user input ”Did you smoke a pepper?” from participant #5 matched to the example sentence ”Can you smoke a pipe?” in the system S-NE because the system did not use POS-tag information. If POS-tag information was used properly, that example sentence would not match to the user utterance because the difference of highly weighted words ”pepper” and ”pipe” would be reflected. 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 match exactly to any sentences in training corpus. However, using proposed utterance matching policy and NE type information, the system find the example sentence ”How do you make a sandwich?” as the most similar sentence, which results in the response ”Bread, cheese, meat, condiments, cut, serve”.

To analyze the effect of each feature to various types of user inputs in detail and compare to the ALICE system, average ratings were calculated by combinations of the processed test sets and systems (Table 3). The proposed system (S-ALL) has the significantly improved performance compared to ALICE. Moreover, S-POS and S-NE also present higher performances than baseline although they are not significant when each feature is applied separately. ALICE shows relatively low ratings on rephrased inputs and high ratings on free inputs. We can assume that numerous heuristic patterns and rules in ALICE make advantages for processing many typical sentences. For example, ALICE has rules like ”WHICH ONE IS *” for processing the input ”which one is preferred?”, so that ALICE can generate proper responses even if similar sentences do not exist in training corpus. On the other hand, many rephrased inputs cannot be processed by ALICE because ALICE could not have all sentence structure as rules. We can also verify that S-POS and S-NE improve performance significantly when processing free utterances and NE-included utterances, respectively.

IV. Conclusion

In this paper, we presented a new chat-bot framework that is originated from EBDM, a task oriented dialogue system. EBDM is relatively free of manual labelling and annotation, and our proposed chat-bot has inherited the advantages. To improve performance while minimizing the human efforts, we suggested three features: POS-tagged token matching, using NE-related information, and using back-off responses according to DA type. We expect that POS-tagged token and NE information help extending the scope of user input patterns and selecting more appropriate responses. If proper examples are not selected, back-off responses are provided to users, so that they can talk with the system without awkwardness. Also, we proposed a personalization framework for dialog systems to build rapport with the user. Personalized dialog system remembered user-related facts, and applied them to the system response.

The experimental results verify that the proposed features are useful for improving the performance of the system. Even though each feature alone cannot improve the performance significantly from baseline system, the system shows dramatically improved performance when the features are adopted simultaneously. Other experimental results show that our system has competitive performance to ALICE on the same training corpus

As a future development of our research, several goals exist. Despite our effort to reduce the human labour, the NE information should be annotated manually. Detecting important topic keywords automatically will enables our system to extent to other language easily. Additionally, more complex features such as parsing information can be used for sentence matching. We may consider the opportunity cost between the precision and execution time while using those features. We are also planning to implement long-term memory into our chatting system. The system could find useful information from user utterance automatically, and it will be exploited by the system for future conversations.

Acknowledgment

This work was partly supported by ICT R&D program of MSIP/IITP. [10044508, Development of Non-Symbolic Approach-based Human-LikeSelf-Taught Learning Intelligence Technology]. This work was also partly supported by National Research Foundation of Korean (NRF) [NRF-2014R1A2A1A01003041, Development of multi-party anticipatory knowledge-intensive natural language dialog system].

References

[1] B. A. Shawar and E. Atwell, "Using the Corpus of Spoken Afrikaans to generate an Afrikaans chatbot," Southern African Linguistics and Applied Language Studies, vol. 21, pp. 283-294, 2003.

[2] J. Weizenbaum, "ELIZA—a computer program for the study of natural language communication between man and machine," Communications of the ACM, vol. 9, pp. 36-45, 1966.

[3] R. S. Wallace, The anatomy of ALICE: Springer, 2009.

[4] B. Wilcox, "Beyond Facade: Pattern matching for natural language applications," GamaSutra. com, 2011.

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

[6] C. Lee, S. Jung, S. Kim, and G. G. Lee, "Example-based dialog modeling for practical multi-domain dialog system," Speech Communication, vol. 51, pp. 466-484, 2009.

[7] X. Shen, B. Tan, and C. Zhai, "Implicit user modeling for personalized search," in Proceedings of the 14th ACM international conference on Information and knowledge management, 2005, pp. 824-831.

[8] F. Qiu and J. Cho, "Automatic identification of user interest for personalized search," in Proceedings of the 15th international conference on World Wide Web, 2006, pp. 727-736.

[9] L. Ardissono, C. Gena, P. Torasso, F. Bellifemine, A. Difino, and B. Negro, User modeling and recommendation techniques for personalized electronic program guides: Springer, 2004.

[10] Y. Jiang, J. Liu, M. Tang, and X. Liu, "An effective web service recommendation method based on personalized collaborative filtering," in Web Services (ICWS), 2011 IEEE International Conference on, 2011, pp. 211-218.

[11] S. J. DeRose, "Grammatical category disambiguation by statistical optimization," Computational Linguistics, vol. 14, pp. 31-39, 1988.

[12] Y. Kim, J. Bang, J. Choi, S. Ryu, and G. G. Lee, "Acquisition and use of long-term memory for personalized dialog systems," in Proceedings of the 2014 Workshop on Multimodal Analyses enabling Artificial Agents in Human-Machine Interaction, 2014.

[13] F. Wu and D. S. Weld, "Open information extraction using Wikipedia," in Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 2010, pp. 118-127.