Evaluation of a Standalone Language-independent Dialogue Framework

Arnaud Jordan[1] **and Kenji Araki**[1]
[1]Graduate School of Information, Science and Technology, Hokkaido University, Sapporo, Japan

Abstract—*This paper presents an innovative dialogue agent designed for textual casual chatting, which can handle any language. The system acquires knowledge from a non-annotated corpus and then represents all the language aspects into a graph. Using previously acquired knowledge it splits sentences into sub-nodes to proceed to a flexible output generation. Moreover, it uses graph clustering to generate node categories without using any grammar-related tags, and uses these categories to induce new knowledge. The system uses the same processing regardless of the language, that makes the system able to handle any language without any adaptation task. In addition, since the system uses only a limited number of resources, it can be set up as a standalone system in order to preserve the user privacy. We carried out dialogue correctness experiments in Chinese, English and Japanese and obtained results comparable to a more language-specific multilingual system.*

Keywords: natural language processing, multilingual system, spoken dialogue agent, real-time, graph clustering

1. Introduction

Nowadays, a large number of spoken dialogue agents have been proposed, such as ALICE [1], or are still under development. Some of them focus on non-task-oriented dialogues, while others focus on providing information or achieving a particular task. In this paper, we focus on non-task-oriented dialogues because we consider it as the first step to build a complete system which at the end may be able to handle both task and non-task oriented dialogues at the same time.

Many non-task-oriented dialogues systems have already been proposed. With progress in research and systems improve, spoken dialogue agents are able to handle more and more situations, like the Multimodal Multi-domain Spoken Dialogue System [2], for example. However, to reach this objective they use many high-level operations such as word categorization or case grammar. Consequently, to handle these complicated processes most of systems require very language-specific resources such as dictionaries or grammatically tagged corpora. For example, many systems work only in a specific language, such as Japanese [3]. These kinds of systems cannot be easily adapted to another language without a lot of work. A solution would be to create a multi-lingual model that handles all languages [4]. However, this is a hard task since each language has specific aspects that are not used in other languages. Nevertheless, it is possible to try to implement the most common behaviors to cover a maximum of languages. However, the result will be incomplete and not optimal for each specific language. That is why we opt for implementing only very basic processes used universally in all languages.

In this paper, we propose a framework that has been developed with the aim of handling any language, and which consequently uses no language-specific resources to keep a maximal generality. For example, the system must be able to handle a newly constructed language using only some samples of this language. In addition, the proposed framework includes no copyright covered elements and as a result can be easily implemented and adapted in various environments. Moreover, it can be considered as a base framework for a system focusing any specific language.

However, in order to check our algorithm before adding new processing we focus on very simple dialogues. We will improve the system in future, for example, we will increase the speed of the output generation to be able to handle more knowledge.

Developing an algorithm that is not dependent on language is a complicated task, and the results may not be better than the current best language-specific systems. However, it would be useful to achieve many different objectives such as those listed below.

- Handling and acquiring the meaning of new terms, such as words used by young people.
- Minority language support, languages for which specific natural language tools are not available.
- Foreign language learning using casual dialogue as training.

Moreover, since the system uses no external tools, it can be easily distributed or installed on a mobile device and works without any network connection as a standalone system. Consequently, it can also provide a full privacy protection to the user.

In addition, the system could be set up to handle non-verbal input such as sign language too. Since the system can handle any kind of input such as gestures, specials tags or texts, it can be considered naturally multimodal.

2. Outline

The proposed system uses graph traversal to generate and select the optimum responses to the user's inputs. Consequently, the system is composed of two main parts: graph construction which replaces the use of external resources, and graph parsingm, which is used to generate the system's responses.

In order to handle more complex dialogues, we will need to improve the first phase to, for example, automatically acquire knowledge in a similar way as is performed in a lexical database such as WordNet [5].

2.1 Graph representation

All the acquired knowledge is represented in a graph using nodes and directed links, as shown in Figure 1. The system generates many different links, however to keep the figure easily readable we only represent some of them.

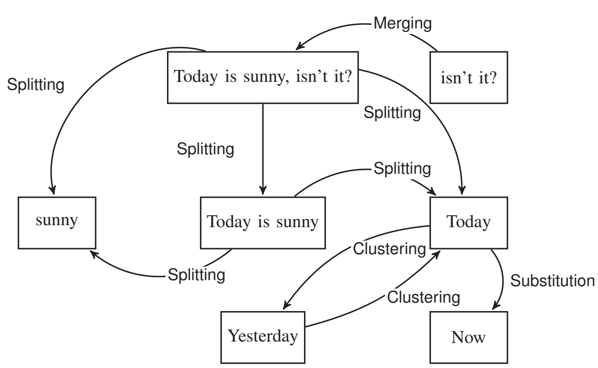

Fig. 1: A simple graph

2.2 Basic links

We used only basic links, which are necessary for output generation [6], in order to preserve the generality of the system.

2.2.1 Splitting

A *splitting link* is generated between a node and its sub-node. Here, sub-node refers to a node whose value is contained in another node value. For example, the node "I like pears" contains the node "pears" and is consequently related to it.

2.2.2 Merging

A *merging link* is the opposite of a splitting link. It is set between a node and a super-node. For example, there is a merging link between "How" and "How are you?".

Concretely, a *merging link* is set between a sentence's part sub-node and all the complete sentences which include this sub-node. Using these links, the system can retrieve complete sentences that are eligible for output.

2.2.3 Substitution

A *substitution link* is provided between a *node A* and a *node B*, if the *node B* can be used instead of the *node A* in the output. This substitution can be considered as a similar process to association in psychology [7].

For example, when the user inputs "Hello", the system can answer "Hello", "How are you?" or "How do you feel?". Consequently, they are *substitution links* from "Hello" to "How are you?" and to the other possible responses.

However, if the input is "How are you?" the system may reply "I am fine" and "I am tired" at the same time, which would not be coherent behavior. To avoid this kind of unexpected action it is possible in a future version of the system to implement emotional concepts; the node "I am fine" can be connected to a good emotion, i.e. a node representing this emotion, and the node "I am tired" to a bad one, and then the system can be set up to output only nodes related to the same emotion when the user inputs a question. These emotional nodes are not related to language, since the same basic emotions are used by all humans [8].

2.2.4 Clustering

The system uses the MaxMax algorithm [9] to create nodes clusters and generate *cluster links* in order to be able to generate more various responses to the user's input. The MaxMax algorithm has been made to suit tasks such as Word Sense Induction (WSI). It is a non-parametrized and graph applicable algorithm which is very easy to implement. However, other clustering algorithms that work on graphs can be easily adapted to be used in the system.

Concretely, for example, "apples" and "oranges" can be related by a *cluster link*. The system attempts to replace the nodes of the sentences with others from the same cluster to generate a new sentence. If "apples" and "oranges" are in the same cluster and if the system learns the sentence "I eat apples", then it will generate the sentence "I eat oranges".

2.3 Node generation

The system uses training samples (cf. 2.6) to generate nodes in the graph before the dialogue starts. Firstly, each sentence of the samples is converted into a node called an *input node*. For example, the sentences "Hello" and "How are you?" are converted into two distinct nodes. Then, the system proceeds to generate the sub-node.

2.3.1 Sub-node generation

In natural language processing, one of the most common tasks is to identify words present in a sentence. However, in the context of a multilingual system we cannot use a morphological analysis tools such as JUMAN [10] which are only available in specific languages, such as Japanese.

A solution would be to use unsupervised word segmentation [11]. However, we need a real-time and fast adaptive algorithm. This is why we develop our own algorithm to identify parts of the sentence. We use already existing nodes to try to split new ones. For example, the system uses the node "I like" to split the node "I like peaches" into "I like" and " peaches". The generated sub-nodes can represent several words, e.g. "like peaches", as well as a single word like "peaches" or a part of a word like "ach".

This method will generate a lot of noise, i.e. many nodes that are not useful for output generation, as well as useful ones. However, as has been proven for stochastic resonance [12], it could also help the system to generate many correct and useful responses. Concretely, the system may access many ineffective nodes which will not be used to generate the output of the system, because they are regularly related to all the other nodes. As a result, their influence on the choice of the output is limited.

2.4 Link characteristics

Each kind of link between nodes has its own characteristics. These are used during the graph traversal to calculate the node's score and the link's cost.

- **Node score** denotes the importance of a node.
- **Link cost** refers to how much power is needed to take the link and go to the pointed node. This value is used to limit the graph traversal.

All links that have a link cost exceeding a defined value (arbitrarily set to 5) are ignored by the system.

Each kind of link has the following three characteristics.

- **Weight** denotes the value of the linked node; links that bring a lot of information such as a substitution link have a high value.
- **Distance** denotes the information difference. Splitting links only remove a part of the information; consequently their distance is small.
- **Base cost** is used to calculate the cost of the link.

Changing these characteristics will change the system's behavior. For example, we can make the system generate more sentences[1], but these will not be all correct or make the system take a more careful behavior[2] and only output sentences that are definitely correct.

Equation (1) is used to calculate the node score, and the weight and distance are calculated by aggregating the total values of the links used to arrive at this node from the user's input.

We use the exponential function, to limit the number of parsed nodes. For example, we want to avoid a path which uses many small distance links.

$$S_n = \frac{\sum weight}{e^{\sum distance}} \tag{1}$$

- S_n is the score of a specific node.

[1]Decrease the *distance value* or the *base cost*
[2]Increase the *base cost*

In addition, we use Equation (2) to calculate the link cost.

We use the number of links to decrease value of very frequent nodes in a similar way to the tf-idf method [13]. This is often the case of nodes resulting from the noise of the splitting algorithm. In addition, we use a logarithm to reduce the difference between two nodes that only have a small difference in number of links, and consequently can be considered similar.

$$C_l = c \times (1 + log(n_{link})) \tag{2}$$

- C_l is the cost of the link.
- c is the base cost of the target link type.
- n_{link} is the number of links of the corresponding type from the same node.

The clustering links are used to create new nodes, but are not used during the graph traversal.

Table 1 contains the empirically defined values for each type of link.

Table 1: Links' characteristic

Link	Cost	Distance	Weight
Splitting	1.5	0.75	2
Merging	0.99	1	3
Substitution	2.5	2	5

2.5 Output generation

As shown in Figure 2, the system checks each input node of the graph to look for all the nodes that match, include or are included in the user's input[3]; all the matching nodes' score is increased.

```
ALGORITHM visitingGraph(input)
    FOR EACH inputNode OF inputNodes
        IF inputNode = input
            inputNode.increaseScore()
            folowLink(inputNode, 0)
        ELSE IF inputNode contains input
            inputNode.increaseScore()
            folowLink(inputNode, 0)
        ELSE IF input contains inputNode
            inputNode.increaseScore()
            folowLink(inputNode, 0)
```

Fig. 2: Algorithm used to find node related to the input

Then, as shown in Figure 3, using the previously-acquired links, the score of all the nodes that are related to a matching input node will be increases too in function of their links' characteristics. All the nodes will be accessed until the link cost exceeds a defined value. The link cost of each link is added to the previous link cost, and as a result the cost used in the comparison increases each time the system follows a link.

[3]For example, the input "I like eating" includes "eating" and is included in "I like chocolate". As a result these two nodes' scores are increased.

```
ALGORITHM folowLink(node, currentCost)
    FOR EACH link ELEMENT OF node.links()
        cost:=currentCost + link.cost()
        IF cost < MAXCOST
            linkedNode:= link.getNode()
            linkedNode.updateScore()
            folowLink(linkedNode, cost)
```

Fig. 3: Algorithm used for graph traversal

Finally, the node that has the best score, which exceeds the trigger value (cf. 2.5.1), is selected and output.

For example, if the input sentence is "I like making cookies", the nodes "I like" and "cookies" are included and their score will be increased. Both of these are related to the node "I like eating cookies" by a merging link, and its score will be increased too. If the score of the node exceeds the trigger value, the system will output "I like eating cookies".

If no node's score exceeds the trigger value after all of the graph has been traversed, the system will output an apology sentence such as "I am sorry, I cannot reply". Those apology sentences are present in the training samples (cf. 2.6).

2.5.1 The output trigger value

In the aim to create a real-time system, the system has to reply in a minimum amount of time like a human would, but with maximum relevance, i.e. the best possible response. To produce this kind of behavior, the system uses a dynamic trigger, which value decrease in function of the time spent, using Equation 3.

$$V_t = V_i - t \times k \tag{3}$$

- V_t is the value of the trigger at t.
- V_i is the initial value.
- t is the time since the initial value.
- k is a defined coefficient.

This equation makes the trigger value decrease continuously using a single parameter that is empirically set. The system periodically checks if a node score exceed the trigger value. Then, it will output the node which has the higher score to the user. After each iteration, the score of the outputted node is set to 0 and the score of all the remaining nodes is decreased.

Figure 4 shows an example of the trigger evolution.

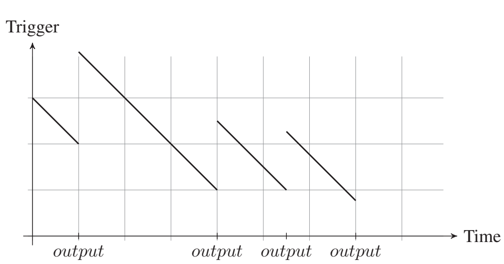

Fig. 4: Example of trigger evolution

After the system selects an output, the trigger value is reinitialized. This new value is calculated using Equation 4.

$$V_i = (\sum_{k=i-5}^{i-1} S_k)/5 \times 2 \tag{4}$$

- S_k is the score of the output k.
- i is the output number.

Concretely, the system uses the mean of the last five outputs' score corresponding to the output node score to calculate the new trigger value. This method allows the system to adapt to the nodes' scores automatically.

2.5.2 Example of output generation

Using the graph of the Figure 5 the system can generate several responses.

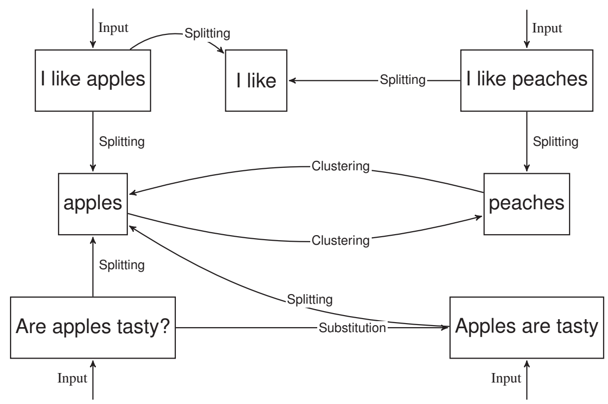

Fig. 5: Example of graph used to generate an output

Some of possible responses are listed below.

- If the input is "Are apples tasty?" the system can directly output "Apples are tasty".
- If the input is "I like apples?" the node "I like apples" will be selected as output, since it is included in the input.
- If the input is "Are peaches tasty?" the system can use the cluster link and output "Peaches are tasty".

The system will output all the nodes which are complete sentence and which score exceed the trigger value, for one input several outputs are possible.

In addition, a dialogue is a real-time process [14], to make the system enable to receive inputs at any time we implement each operation in a different thread executed in parallel. Since, there are no blocking operations, the system can continue to receive inputs while it is generating an output. Moreover, the new input will influence the current output generation.

2.6 Training samples

To generate the graph, the system uses two kinds of basic resources, which contain no grammar information and need no complex creation processes. They can be, for example, extracted from a dialogue between two humans or from any kind of text such as books or screenplays.

Compared to a common system based on AIML [15] corpus, the corpus of the presented system contains no tags and all the rules are automatically acquired from the samples. For example, wild-cards, which are often present in corpus-based chat-bots, are not present in the samples. They have to be statistically induced[4] by the system.

2.6.1 Dialogue samples

The dialogue samples contain some very simple dialogues such as those shown in Figure 6, used to acquire substitution and splitting links in the target language. Concretely, a substitution link is set between an utterance and its response and between all the sub-nodes of the utterance and all the nodes of the response which are not present in the utterance.

U1: what do you drink? U2: I drink milk

Fig. 6: Example of a dialogue sample

2.6.2 Knowledge samples

The knowledge samples are a list of simple sentences such as "I like cookies" or "The president of the USA is Obama".

The knowledge samples are used to increase the possible outputs of the system. These samples can be collected easily, since they consist of a list of simple sentences that are not contextually related. They can be collected from a text such a Wikipedia article or from users' dialogues.

3. Experiments

We used the same protocol as the evaluation of generality of SeGA-ILSD [16]. However, for our system we do not use an ELIZA-type system to generate part of the responses. We also do not use morphological analysis tools as the baseline system does.

In order to fit the baseline experiment process, we used a speech input tool. However, this kind of speech recognition tool uses a lot of language-dependent resources and they are only provided for a limited number of languages. That is why for the experiment we only consider the speech recognition as an input tool which replace the keyboard and which is not a part of the presented system itself.

We use the Google speech recognition implemented in an Android[5] application to get the user's inputs and evaluations.

We asked subjects to evaluate each response of the system as below.

- **Correct reply** Meaning is correct, and expression is natural.
- **Semi-correct reply** Meaning is correct, but expression is not natural.
- **Erroneous reply** Meaning is not correct.

The aim of the evaluation is simply to check whether the system's responses are grammatically correct and correspond to the user input. Nevertheless, we asked the subjects to evaluate the system's response as erroneous if the system does not reply to the input question. For example, if the input is "What will you do tomorrow?" and the response is "I don't know", it is considered to be erroneous even if the output is grammatically correct and a human could reply in a such way.

3.1 Baseline

We used the SeGA-ILSD system as a baseline for this experiment. This spoken dialogue system uses an inductive learning method based on genetic algorithms with sexual selection. Concretely, it acquires rules automatically from pairs consisting of an utterance and its associated reply, and attempts to crossover two rules to create a new one. Rules that generate erroneous output are progressively removed from the system using user feedback.

In order to crossover two rules, the system needs to identify each word in the sentences and in consequence, for which it uses a morphological analysis tool[6].

In addition, when no rules are found to reply to the input, the system uses an ELIZA-type system to generate the output. The ELIZA-type system contains manually created rules that are different for each language.

Moreover, the baseline uses Microsoft Japanese recognizer (Version 6.1), Microsoft English Recognizer (Version 5.1) and Microsoft Simplified Chinese Recognizer (Version 5.1) as speech recognition tools[7].

3.2 Preparation of the experiment

We asked three native speakers each of Chinese, English and Japanese to imagine each one a simple casual dialogue of about 40 sentences in order to create the dialogue samples.

The Japanese knowledge samples were directly extracted from our previous research. For this research we asked subjects to teach some common knowledge to train a spoken dialogue agent. The same samples were also manually translated into the two other languages by a native speaker.

For Chinese (Mandarin) we used simplified Chinese characters. We did not make any distinction between different kinds of English. The used corpus can be considered as small, however in order to be able to compare the system comportment in the three languages we prefer to favor the corpus unity than the corpus size to carry out first experiments.

[4] A word that has many substitution links can be considered to be a kind of wild-card.

[5] http://developer.android.com

[6] JUMAN Version 5. for Japanese, Apple Pie Parser Version 5.9 [17] for English and ICTCLAS for Chinese [18]

[7] The version 6.1 stems from Microsoft Office 2003 and the version 5.1 is extracted from the package Microsoft Speech SDK 5.1: http://www.microsoft.com/en-us/download/details.aspx?id=10121

3.2.1 Splitting parameter

To avoid the generation of too many nodes in languages using Latin characters, we set a minimal character length of four to split a string in English.

We also did not use the sentence starting capital letters to increase the node matching rate. For example, in the sentences "Cats are cute" and "I like cats", "Cats" and "cats" are the same word, but they will be considered different words by the system because of the capital letter. However, we kept meaningful capital characters, such as in the case of proper nouns.

It is important to note that the knowledge required to know if a word needs a capital or not depends of the language. With a bigger corpus we think this task can be avoid without an important impact to the system since the number of nodes will be sufficient to split all the words with and without a capital letter.

Characters depend on the language, but they do not make the system language-dependent. The user can input any character into the system; the output generation process will not be affected. For example, a word such as "t%&3=f" can be learnt by the system in the same way as all other words.

3.3 Experiment settings

Table 2 shows details about the subjects who participated in the evaluation of the proposed system.

Table 2: Subject's information

	Chinese	**English**	**Japanese**
Subject	7	4	13
Male	2	2	5
Female	5	2	8
Student	7	2	9
Worker	0	2	4
Age [year]	21.9	21.5	23.0

Table 3 summarizes information about the system knowledge. The number of nodes refers to the number of nodes created before the user starts the dialogue. To count the number of words, we split sentences using spaces for English, for Japanese we used the JUMAN tool [10], and for Chinese we use the Google Translator's[8] word segmentation.

Table 3: Corpus' information

Number	Chinese	English	Japanese
of nodes	1410	715	1623
of words	1246	1029	1268
of characters	1818	4853	2514
of sentences (knowledge)	100	100	100
of sentences (dialogues)	88	69	85

We can see that more a language has words more generated nodes are numerous. In addition, the number of nodes in the system is small. However, we think that bigger graphs can be easily handling as well using parallel processing.

[8] `https://translate.google.com/`

3.4 Experiment results

3.4.1 Speech recognition precision

The speech recognition tool does not recognize punctuation. Consequently, the subjects was not able to input point or question mark in their inputs.

The precision is calculated using Equation 5. This formula is the same as that used in the baseline.

$$P = \frac{n_c}{n_c + n_i} \tag{5}$$

- P is the precision.
- n_c is the number of correct recognitions.
- n_i is the number of incorrect recognitions.

Table 4 shows the precision for each language.

Table 4: Speech recognition precision

	Chinese	**English**	**Japanese**
Precision	93.4%	86.7%	89.3%

3.4.2 Evaluation of responses

Figure 7 shows an example of dialogue with the proposed system. Evaluation by the subjects is presented in parentheses. The sentence starting capital letters are added after the output generation.

User: What do you like?
System: I like studying *(Correct reply)*
System: Do you like studying? *(Correct reply)*

Fig. 7: Example of a dialogue with the system

The system firstly outputs "I like studying", then because the score of the nodes "studying" and "like" are high, it also outputs the sentence "Do you like studying?". This comportment can generate natural dialogues as well as erroneous outputs. That is why we will try to enhance it in further research using more parameters such as emotions and contextual nodes for the output generation.

Table 5 contains the experiment results compared to the baseline [19]. For the baseline, the values in parentheses correspond to the evaluation results for the ELIZA-like output.

Table 5: Experimentation results [%]

Language	**Correct**	**Semi-correct**	**Erroneous**
Chinese	25.9	17.0	57.1
English	39.1	14.2	46.7
Japanese	31.7	16.2	52.1
Baseline* (ELIZA's responses)			
Chinese	25.6 (16.0)	13.6 (30.4)	4.4 (10.0)
English	4.0 (8.4)	16.0 (53.2)	15.2 (3.2)
Japanese	14.6 (38.2)	2.5 (13.5)	8.9 (22.3)

* Results without the ELIZA's responses.

However, both parts of the baseline are evaluated in a single run. ELIZA's responses are used when no other response is available.

3.5 Results analysis

We can see that the results of the three languages are similar. In addition, they exceed the baseline's results if we exclude the ELIZA's responses. We consider ELIZA's responses as language dependent, since they are manually inserted in the system for each target language. The proposed system is able to answer most of the greetings and some questions of the user. It does not simply look for a matching rule, but it decomposes the input and analyses the nodes related to each parts in order to output the best responses.

3.5.1 Used resources

The baseline uses a morphological analysis tool and an ELIZA-type system, both of which are language-specific. However, the other parts of the output generation do not depend on language. Consequently, the system can be adapted to other languages with a minimal amount of work for any language for which such tools are provided. However, if one of these tools is not available, the adaptation task becomes much more complicated.

In comparison, the proposed approach only needs language samples to be trained and then be able to handle a dialogue in the targeted language. These samples can be simply extracted from the user's own chat logs.

To achieve a fully end-to-end language-agnostic dialogue system, it is possible to start the system without any knowledge and to allow it to acquire knowledge from the users' inputs. However, in this case the teaching process will be very annoying for the user. A better method would be to make the system assist with a dialogue between two humans and acquire knowledge in a similar way to a child hearing people around him and finally becoming able to speak. The dialogue samples used in this paper can be considered to be dialogue heard by the system during its "childhood".

4. Conclusion

In this paper, we used an unique language free algorithm to provide a real-time spoken dialogue agent to the user. We carried out experiment in Chinese, English and Japanese, and obtained similar results in all these languages. Moreover, the precision obtained exceed the baseline if we exclude ELIZA's responses.

The SeGA-ILSD system handles several languages; however it needs to be adapted to each one. In contrast, the proposed system needs no special work to be adapted to another language. For example, we can input into the system both Chinese and Japanese training samples at the same time, and the system will be able to output Chinese as well as Japanese sentences. However, it cannot preserve contextual information from one language to the other.

In our future research, we will add emotional nodes [20] to the graph in order to enable the generation of more outputs using more parameters. In addition, sharing knowledge between users [21] would help the system to acquire many different kinds of knowledge directly from the users.

References

[1] Richard S. Wallace. 2009. *The Anatomy of A.L.I.C.E.*, pp. 181-210. Parsing the Turing Test.

[2] Ridong Jiang, Rafael E. Banchs, Seokhwan Kim, Kheng Hui Yeo, Arthur Niswar and Haizhou Li. 2014. *Web-based Multimodal Multi-domain Spoken Dialogue System*. Proceedings of 5th International Workshop on Spoken Dialog Systems.

[3] Arnaud Jordan and Kenji Araki. 2013. *Spoken Dialog Processing for Acquiring Taste and Knowledge*. Proceedings of PACLING2013.

[4] Ryan T. McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B. Hall, Slav Petrov, Hao Zhang, Oscar Täckström and others. 2013. *Universal Dependency Annotation for Multilingual Parsing*, pp. 92-97. Proceedings of ACL2013.

[5] George A. Miller. nov. 1995. *WordNet: A Lexical Database for English*, Vol. 38, No. 11, pp. 39-41. Communication of the ACM. Association for Computing Machinery.

[6] Arnaud Jordan and Kenji Araki. 2014. *A Framework for Multilingual Real-time Spoken Dialogue Agents*, pp. 24-29. Proceedings of iCAST2014.

[7] Howard C. Warren. 1921. *A History Of The Association Psychology*. Charles Scribner's Sons.

[8] Paul Ekman. 1972. *Universals and Cultural Differences in Facial Expression of Emotion*, pp. 207–283. J. Cole ed. Nebraska Symposium on Motivation. University of Nebraska Press

[9] David Hope and Bill Keller. 2013. *MaxMax: a graph-based soft clustering algorithm applied to word sense induction*, pp. 368-381. Computational Linguistics and Intelligent Text Processing. Springer.

[10] Kurohashi and Kawahara lab. 2012. *Japanese Morphological Analysis System JUMAN version 7*. Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University.

[11] Daichi Mochihashi, Takeshi Yamada and Naonori Ueda. mar. 2009. *Bayesian Unsupervised Word Segmentation with Hierarchical Language Modeling*, pp. 49-56. IPSJ SIG Notes 2009(36). Information Processing Society of Japan (IPSJ).

[12] Luca Gammaitoni, Peter Hänggi, Peter Jung and Fabio Marchesoni. 1998. *Stochastic resonance*, volume 70, pp. 223. Reviews of modern physics. American Physical Society (APS).

[13] Gerard Salton and Michael J. McGill. 1986. *Introduction to modern information retrieval*. PMcGraw-Hill, Inc.

[14] Masashi Takeuchi, Norihide Kitaoka and Seiichi Nakagawa. 2004. *A spoken dialog system activating the natural response timing using prosodic and linguistic information for chat-like conversation*, pp. 87-92. IPSJ SIG Notes 2004(15). Information Processing Society of Japan.

[15] Richard S. Wallace. 2003. *The elements of AIML style*. Alice AI Foundation.

[16] Kenji Araki and Michitomo Kuroda. 2006. *Generality of spoken dialogue system using SeGA-IL for different languages*, pp. 72-77. Proceedings of the Second IASTED. Computational Intelligence.

[17] S. Sekine and R. Grishman. 1995. *A Corpus-based Probalistic Grammar with Only Two Non-Terminals*. Proceedings of the Fourth International Workshop on Parsing Technologies.

[18] Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong and Qun Liu. 2003. *Q.HHMM-base Chinese lexical analyzer ICTCLAS*, volume 17, pp. 184-187. Proceedings of the Second SIGHAN Workshop on Chinese Language Processing.

[19] Kenji Araki and Michitomo Kuroda. Jan. 2007. *Evaluation of Generality of SeGA-ILSD for a Chat Using Different Languages*, volume 2007, pp. 79-85. IPSJ SIG Notes 2007(7). Information Processing Society of Japan.

[20] M. Ptaszynski, P. Dybala, R. Rzepka and K. Araki. 2008. *Effective analysis of emotiveness in utterances based on features of lexical and non-lexical layer of speech*, pp. 171-174. Proceedings of the Fourteenth Annual Meeting of the Association for Natural Language Processing.

[21] Arnaud Jordan and Kenji Araki. 2013. *Comparison of two Knowledge Treatments for Questions Answering*, pp. 55-62. Proceedings of SNLP2013.