人工知能学会研究会資料
SIG-SLUD-B402-01

Robust Example-based Dialog Retrieval using Distributed Word Representations and Recursive Autoencoders

Lasguido Nio, Sakriani Sakti, Graham Neubig, Tomoki Toda, Satoshi Nakamura

Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan
E-mail: {lasguido.kp9,ssakti,neubig,tomoki,s-nakamura}@is.naist.jp

Abstract: Based on our previous work on example-based chat-oriented dialog systems that utilize a human-to-human conversation. Though promising, our previous simple retrieval techniques resulting a weakness on handling an out of vocabulary (OOV) database queries. In this paper we discuss an approach to increase the robustness of example-based dialog response retrieval. We employ a recursive neural network paraphrase identification technique to achieve a good performance on finding a good response in dialog-pair database. To achieve that, we remodel our previous dialog-pair database into distributed word representation. We apply an recursive autoencoders and dynamic pooling algorithm between user utterance and database to decide whether they have a same meaning or not, even when they composed in different word and structure. These distribute representations and recusive autoencoders have the potential to enhance our retrieval techniques and reduce confusion in example matching, especially when handling an OOV cases. We also present the system performance evaluation based on objective and subjective metrics.

1 Introduction

Recently, an intelligent dialog system that can give a natural response gain more attention. Despite a demand on specific task dialog system, researcher are seeing increasing interest in developing an lightweight data-driven methods to cover broad-coverage chat-oriented dialog systems [1, 2, 3, 4]. It becomes more attractive because, comparing to the rule based technique [5, 6], these method doesn't rely on the complicated hand-made rules and easy to expanded. Data-driven approaches allow for the use of large amounts of data on the Web to efficiently find responses for a large variety of user queries. However, to achieve broad coverage, recording of a large data set of real human-to-human conversation is necessary. Some studies propose constructing dialog examples from available log database like Twitter [8] or Wizzard of Oz (WOZ) [7].

Example-based dialog modeling (EBDM) is one of many approach to data-driven dialog. As one of the data driven approaches, EBDM utilize a data that consists of query-response pairs. Where the query is representative of the user's input to the system, and the response is representative of the system's response. By simply generate an answer that are actually included in the database as an example, EBDM is able to generate an highly natural output when a response is included in the database, especially when the user utterance is included in the database and the example is able to be appropriately retrieved [1, 2, 3]. Despite these advantages, EBDM may response uncorrelated response when the system could not find any example in the database. Most EBDM system rely on canned or template response to handle this problem [9, 10], which may resulting in less than unsatisfactory output.

In most cases, EBDM employs response retrieval techniques to match up the user's utterance with a query in the query-response database. Later, it will outputs a response that correspond to the matching query. The greater number of previous work performs response retrieval by using a simple lexical measure such as TF-IDF weighted cosine similarity [11, 12] or syntactic-semantic similarity based on POS strings and WordNet synsets [13]. Still, these simple response retrieval techniques are not sufficient enough to comprehend a number of situations like retrieving a simple logic manipulation or paraphrase sentence in the example database.

There are two factors that greatly contribute to the accuracy of EBDM systems: (1) the coverage of the dialogue corpus, and (2) the effectiveness of the example retrieval. In this work, we focus on latter these problems, especially on employing more sophisticated method for matching user utterances and queries in the example database.

In this work, we rely on the great improvement in distributional representations of language using vector space word representations [14]. This compositional distributional word representation that built upon a neural networks [15] may have a potential to capture a large number of linguistic phenomenon. Following the successful work of Socher et al. [16], we also employ recursive autoencoders model to match the user utterance and queries in the example database. In the end, we evaluate our proposed method subjectively and objectively by utilizing dialog corpora constructed from movie conversation data as a testbed.

2 Dialog System and Baseline EBDM

Generally, EBDM chooses a response from the examples database by computing similarity measure between the user input and the query part of the query-response pairs. Then, it returns the associated response for the query with the highest similarity. Based on the previous work [17, 18], we utilize TF-IDF based cosine similarity as the baseline similarity measure for use in EBDM.

TF-IDF based cosine similarity calculates cosine similarity (Equation (1)) over the term vector of two sentences, S_1 and S_2. In order to increase the emphasis on important words, We applied additional TF-IDF weighting to the term vector (Equation (2)) [19]. We set $F_{t,T}$ as a term frequency t in a sentence T, and DF_t as the total number of sentences in the query-response pairs that contain term t.

$$cos_{sim}(S_1, S_2) = \frac{S_1 \cdot S_2}{\parallel S_1 \parallel \parallel S_2 \parallel} \tag{1}$$

$$TFIDF(t, T) = F_{t,T} \log \left(\frac{|T|}{DF_t} \right) \tag{2}$$

3 Neural Network Based Retrieval

Though simple, a cosine similarity method have problems with robustness, which we will discuss in more detail in the following sections. Therefore, our proposed method employ neural network-based retrieval to retrieve more appropriate responses from the example database. In this method, a proper system response is retrieved by modeling our example database using neural word representations and calculating the probability that the user input and a query in the database are paraphrases.

An overview of the neural-network-based retrieval is depicted in Figure 1. To decide weather the sentence is paraphrased or not, we utilize recursive autoencoders (RAE), dynamic pooling, and a dynamic pooling's softmax classifier, adopting the work of [16].

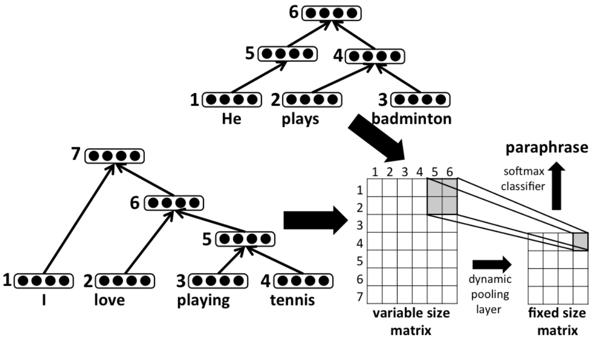


図 1: Overview of neural-network-based retrieval.

3.1 Word Representation

Word representation are obtained with discriminative and non-probabilistic model [14], in which a word in the dictionary ($i \in D$) is embedded into q-dimensional space $S \in \mathbb{R}^{q \times |D|}$. During training, the model concatenate n embedded word $e(i)$, where e is the lookup table and i is a word in dictionary. For every training update, the model read n-gram $x = (i_1, i_2, ..., i_n)$ from the corpus. For every n-gram x, the concatenated model $E(x) = e(i_1) \cdot e(i_2) \cdot ... \cdot e(i_n)$ ("$\cdot$" is concatenation) will be passed through single hidden layer neural network to predict the score $s(x)$. We define $x' = (i_1, i_2, ..., i'_n)$ as a noise n-gram, where the last word of the n-gram $i'_n \neq i_n$ is chosen uniformly from the vocabulary [20].

The training runs based on the criterion that the input n-gram x must have a score at least some margin higher than the noise n-gram x'. By minimizing the loss stochastic $L(x) = max(0, 1 - s(x) + s(x'))$, the training do the gradient descent simultaneously over the neural network parameter and the embedding lookup table. In the end, the embedding lookup table is the output product of the training process. By improving this representation with the word cooccurrence statistics, this representation may capture distributional syntactic and semantic information [14].

One of the word representation advantage is that they allow for soft matching of similar words, especially when there is no exact match. For instance, if we have an input sentence "I like to frequent taverns" that is unavailable in the example database, our existing response generator will fail and response the user input with an uncorrelated response because it could not find any in the database. However, by using distributed word representation, we could match the above sentence with another sentence that semantically similar but has different words such as "visit" for "frequent" and "bars" for "taverns".

3.2 Recursive Autoencoder

Now we have word representation to represent each word in the dictionary. In order to represent a sentence which is a group of sentence, we need to string up the word to each other. We will use the RAE algorithm to do that part. This algorithm combines word representation into vector representations of longer phrases in a syntactic parse tree. By using the syntactic parse tree, this algorithm may capture the compositionality of meaning that is naturally constrained by the tree. To construct the vector representation, this algorithm needs word representations and a binary syntactic tree as input.

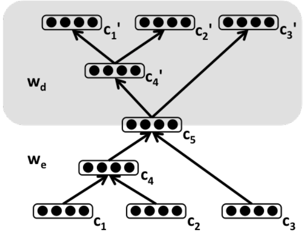


図 2: Recursive autoencoder model.

During construction, every child and non-terminal node in the binary tree is collected as a feature representation of a sentence. A binary tree is composed by parents and its children that form a triplets relationship ($p \to c_1 c_2$). A child could be a word representation vector or another non-terminal nodes. A parent p is calculated through the neural network layer (Equation (3)).

$$p = f(W_e[c_1; c_2] + b), \tag{3}$$

where $[c_1; c_2]$ is concatenation of the vector of two children and f is a tanh activation function.

The parameters W_e and b are trained using recursive autoencoders as shown in Figure 2. The RAE performance is evaluated through the Euclidean distance between original input and its estimated reconstruction node (Equation (4))

$$E(p) = ||[c_1; c_2] - [c_1'; c_2']||^2 \tag{4}$$

where

$$[c_1', c_2'] = f(W_d p + b_d). \tag{5}$$

This process will be repeated recursively for all non-terminal nodes. In the course of RAE training, we want to minimize the total error of all inputs pairs on every non-terminal node. The total error can be obtain by adding up all the calculated error from a single parse tree T

$$E_{tree}(T) = \sum_{p \in T} E(p). \tag{6}$$

One advantage of using recursive autoencoder is that it can capture the compositional structure of phrases and their similarity within the two given sentences. For example, a sentence "tons of stuff to throw away" and "a lot of junk to dispose" there are relationship between words and phrases such as "tons of stuff" with "a lot" and "throw away" with "dispose". Using the recursive autoencoder, we can not only capture the word paraphrase similarity, but also the phrase similarity.

3.3 Dynamic Pooling and Softmax Classifier

Once we obtained the RAE-derived representation of the sentence, we would like to calculate the similarity of two sentences. To achieve that, we need to deal with the arbitrary length of the sentence. Thus, normalizing the RAE word representations into a fixed length of vector become necessary. Given the binary tree representation of a sentence, we can define a matrix M, where rows and columns in this matrix represent two sentences with the different lengths i and j. This row and column represents the non-terminal nodes and leaves in the binary tree, therefore the matrix M's size is $2i - 1 \times 2j - 1$.

In dynamic pooling, we aim to takes a matrix M as an input and output a matrix M' with the fixed size $n \times n$. In order to do so, matrix M will be divided into n roughly equal parts. Every minimal value in the rectangular window is selected to form a $n \times n$ grid. During this process, matrix M' will lose some part of the information compared to the original matrix M [16]. But this approach manages to capture the matrix M's global structure.

By obtaining the uniform size of matrix M' from the given two sentence, next we classify matrix M' using a softmax classifier. Softmax classifier takes the matrix M' as an input, and outputs a confidence score that decided whether a user utterance and dialog database is a paraphrase or not.

4 Experimental Setup

4.1 Dialog Corpora

A dialog corpus used in this experiment is collected from several movie script source such as Friends TV show scripts[1], The Internet Movie Script Database[2], and The Daily Script[3]. The total number of gathered movie scripts is 1,786 with 1,042,288 dialog pairs. More details on the data can be found in [18]. In general, we define two basic types of information for each dialog: actor and utterances. The utterances are the actual content of each dialog turn in the movie scripts. The actor refers to the character name in the movies. Later, this information will be utilized in constructing the dialog corpus.

4.2 Filtering

In our previous study, we have introduce a *tri-turn* unit to find the candidate of dialog-pairs [17]. Given the HTML format of movie script as input, we construct a dialog corpus by perform a filtering as follows,

1 **The extraction of a dialog-pairs**, which ensures that the conversation is done between two people talking each other. To obtain two-way conversation script, we perform dialog (*tri-turn*) extraction to find the candidate of dialog-pairs. A tri-turn is a three conversation turns between two actors X and Y that has the pattern X-Y-X. In tri-turn, the first and last dialog turn are performed by the same actor and the second dialog turn is performed by the other actor. Later the query-response pairs are made by separating the tri-turn pattern X-Y-X into two pairs, X-Y and Y-X.

2 **Semantic similarity calculation**, which ensure that each query-response pair is semantically related. This process employs semantic similarity [21] as shown in Equation (7). The similarity of sentence X and Y can be obtained by calculating the relation between X_{syn} and Y_{syn}, where X_{syn} and Y_{syn} respectively is a group of WordNet[4] synsets for each word in the sentence X and Y. Next, every high similarity dialog pair will be included as an example database.

$$sem_{sim}(X,Y) = \frac{2 \times |X_{syn} \cap Y_{syn}|}{|X_{syn}| + |Y_{syn}|} \quad (7)$$

4.3 EBDM Setup

We perform the filtering on the EBDM example database by using natural language tools and Wordnet synsets provided by NLTK toolkit[5]. We also use Apache Lucene[6] to calculate the TF-IDF based cosine similarity. During experiment, we define Q_{test} as a query from the test set. In order to get the appropriate response similar to the actual response R_{test}, we use two retrieval techniques, TF-IDF based cosine similarity retrieval $cos_{sim}(Q_{test}, Q_{train})$ and NN-based retrieval $rnn(Q_{test}, Q_{train})$. They are used to obtain the closest query Q_{train} in the train database. The response R_{train} that trails the closest query is given as an output response.

4.4 NN-Based Retrieval Setup

During the experiment, we employ the 100-dimensional word representation computed and provided by Turian et al. [20]. We also use the trained RAE with 150,000 sentences from NYT and AP section of Gigaword corpus, provided by Socher et al. [16]. As an input for RAE, we use Stanford parser [22] to generate the parse tree for our conversation dialog database.

After performing pre-processing, filtering, and choosing dialog pairs that can be transformed into a vector of word representations, we finally use 10,033 dialog pairs as our training and test data. We randomly separate our dialog pair data into 1,000 dialog pairs for test and 9,033 dialog pairs for train.

In order to provide a balance amount of similar and non-similar queries during training, we do cross product to all training dialogues (9,033 pairs) with each other, and calculate the the syntactic-semantic similarity [18]

$$sim(S_1, S_2) = \alpha[sem_{sim}(S_1, S_2)] + (1-\alpha)[cos_{sim}(S_1, S_2)].$$

We assume that a similar query is obtained when the syntactic-semantic score is exclusively between 0.7 and 0.9, and a non-similar query is obtained when the syntactic-semantic score is exclusively between 0.2

[1] http://ufwebsite.tripod.com/scripts/scripts.htm

[2] http://imsdb.com/

[3] http://dailyscript.com/

[4] http://wordnet.princeton.edu/

[5] http://nltk.org

[6] http://lucene.apache.org/

and 0.4[7]. In the end, we obtained 1,421,338 pairs of training data with the ratio between similar and non-similar sentences being 50:50.

5 Evaluation

5.1 Performance of NN-Based Retrieval

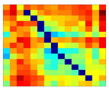
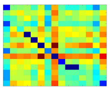
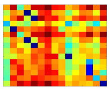
sim	Sentences	Matrix
0.94	S_1) Captain, we can not keep going fast on these icy roads. S_2) We can not keep going fast on these icy roads!	
0.60	S_1) Hold your fire! He's got a girl. S_2) Looks like he's got a hostage.	
0.50	S_1) I've been careful, I've been waiting my chance. S_2) Oh, you've been under a lot of stress.	

表 1: Sentence pairs.

The correlation between user input and example database can be seen in the Table 1. For each utterance pair ($S1$ and $S2$) we calculate syntactic-semantic score *sim*. The rightmost column contain a matrix representation that depict the paraphrase relations between two utterances. As seen in the matrix representation, when a utterance pair have a high similarity score (a similar pair), it will generates a clear diagonal structure of dark line. This diagonal structure was a result from the Euclidean distance computation. When this diagonal structure is clearly seen, NN-based retrieval is manage to find a close or paraphrased sentence to the input query.

5.2 Objective Evaluation

	CSM > threshold	RNN > threshold
CEF	O	
OOV	X	O
OOV-NP	X	X

表 2: Evaluation case.

We evaluate the system response objectively by calculating the system output response R_{output} similarity with the actual expected output R_{test}. We compare the NN-based retrieval (RNN) to the baseline system using TF-IDF cosine similarity retrieval (CSM). In order to do so we score the system by using the same scoring model (RNN and CSM) that we use for retrieving the system response.

In our observation, we found out that the two methods have roughly similar results. In order to look more closely at the evaluation results, we split all testing data into three cases (Table 2) based on each system's retrieval score.

1 Case 1 (closest example found; CEF) when the CSM retrieval score is more than a threshold. During this case, the CSM has managed to find a close example in the database.

2 Case 2 (out of vocabulary; OOV) when the CSM retrieval score is less than or equal to threshold and RNN retrieval score is more than the threshold. In this group, the CSM does not manage to find any close examples in the database, but the paraphrase identification model did.

3 Case 3 (out of vocabulary - non paraphrase; OOV-NP) when both the CSM and RNN retrieval scores are less than or equal to the threshold. For this case, there is no match with the given input for either method.

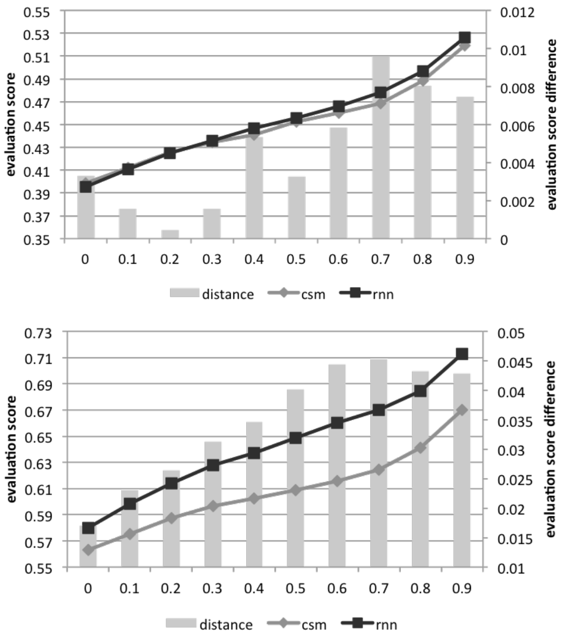


図 3: Objective evaluation results for each threshold over TF-IDF based cosine similarity metrics (upper) and NN-based similarity metrics (lower).

As can be seen in the Figure 3, we measured the objective evaluation score at different thresholds to determine the threshold value. We obtain a threshold score 0.7 by observing the longest evaluation score difference between RNN and CSM. With this thresh-

[7] Note that it would be better to manually create a corpus of similar and non-similar utterances, but this is extremely time consuming, so we take the more light-weight automatic approach in this paper.

old, the amount of data in CEF, OOV, and OOV-NP respectively are 587, 206, and 207.

The details of the objective evaluation results for each of these groups is shown in Figure 4. The left figure shows evaluation calculated by TF-IDF based cosine similarity (*cos-tf-idf*), the right figure shows evaluation calculated by NN-based similarity (*paraphrase*). Furthermore, by using the best threshold score in Figure 4, we can see that the NN-based retrieval RNN approach performs better compared to the baseline method in the OOV data. However, the results is dropped in the case OOV-NP data.

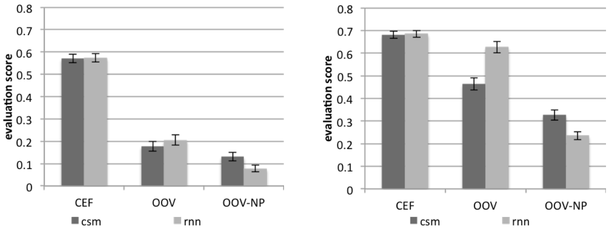


図 4: Objective evaluation results. The left figure shows evaluation over TF-IDF based cosine similarity, the right figure shows evaluation over NN-based similarity.

5.3 Subjective Evaluation

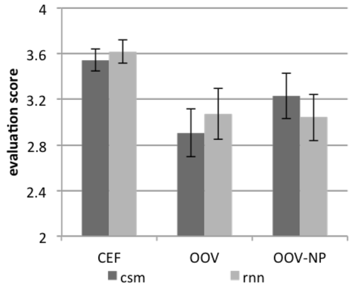


図 5: Subjective evaluation results.

During the subjective evaluation, we asked 10 human annotators to give a naturalness score between 1-5 to the system response R_{output} given the input query Q_{test}. A higher score indicates that the system is giving a natural and relevant system response to the user input, otherwise lower scores indicate that the system doesn't give a related or natural response. In this evaluation, each person assesses 50 randomly selected query-response pairs that were evenly distribtued over all systems. The results of this evaluation are shown in Figure 5. We can see in the figure that the RNN perform slightly better compared to the baseline approach for the CEF and OOV cases. However, when both of the systems can not find any good response (OOV-NP), the RNN response may not be related to the user utterance topic. In this situation, the user tends to choose the baseline approach response over the RNN response.

6 Conclusion

In this work, we employed distributed word representation and recursive autoencoders as an paraphrase identification model. We investigated this model to retrieve responses in a data-driven chat-oriented dialog system, and compared it with our previous work in TF-IDF based cosine similarity retrieval. The experiment shows that the NN based retrieval is able to capture the correlation between user input and example database especially when the user input is not available in the example database (OOV case). Furthermore, objective evaluation shows that that the NN-based retrieval approach performs slightly better compared to the TF-IDF based cosine similarity retrieval approach when both methods find a response, or when only the RNN method finds a response.

参考文献

[1] S. Jung, C. Lee, and G.G. Lee, “Dialog studio: An example based spoken dialog system development workbench,” in *Proc. of the Dialogs on dialog: Multidisciplinary Evaluation of Advanced Speech-based Interactive Systems. Interspeech 2006-ICSLP satellite workshop*, Pittsburgh, USA, 2006.

[2] C. Lee, S. Lee, S. Jung, K. Kim, D. Lee, and G.G. Lee, “Correlation-based query relaxation for example-based dialog modeling,” in *Proc. of ASRU*, Merano, Italy, 2009, pp. 474–478.

[3] K. Kim, C. Lee, D. Lee, J. Choi, S. Jung, and G. G. Lee, “Modeling confirmations for example-based dialog management,” in *Proc. of SLT*, Berkeley, California, USA, 2010, pp. 324-329.

[4] A. Ritter, C. Cherry, and W. B. Dolan, “Data-driven response generation in social media,” in *Proc. of EMNLP*, Edinburgh, Scotland, UK., July 2011, pp. 583-593.

[5] J. Weizenbaum, “Eliza - a computer program for the study of natural language communication between man and machine,” *Commun. ACM*, vol. 9, no. 1, pp. 36-45, 1966.

[6] R. Wallace, *Be Your Own Botmaster*, A.L.I.C.E A.I. Foundation, 2003.

[7] H. Murao, N. Kawaguchi, S. Matsubara, Y. Yamaguchi, and Y. Inagaki, “Example-based spoken dialogue system using WOZ system log,” in *Proc. of SIGDIAL*, Saporo, Japan, 2003, pp. 140-148.

[8] F. Bessho, T. Harada, and Y. Kuniyoshi, “Dialog system using real-rime crowdsourcing and Twitter large-scale corpus,” in *Proc. of SIGDIAL*, Seoul, South Korea, 2012, pp. 227-231.

[9] C. Lee, S. Jung, S. Kim, and G. G. Lee, “Example-based dialog modeling for practical multi-domain dialog system,” *Speech Commun.*, vol. 51, no. 5, pp. 466-484, May 2009.

[10] N. Chambers and J. Allen, “Stochastic language generation in a dialogue system: Toward a domain independent generator.,” in *Proc. of SIGDIAL*, Cambridge, Massachusetts, USA, 2004, pp. 9-18.

[11] R. E. Banchs and H. Li, “IRIS: a chat-oriented dialogue system based on the vector space model,” in *Proc. of ACL (System Demonstrations)*, 2012, pp. 37-42.

[12] R. E. Banchs, “Movie-dic: a movie dialogue corpus for research and development,” in *Proc. of ACL (2)*, 2012, pp. 203-207.

[13] L. Nio, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, “Combination of example-based and smt-based approaches in a chat-oriented dialog system,” in *Proc. of ICE-ID*, Bali, Indonesia, 2013.

[14] R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” in *Proc. of ICML*, New York, NY, USA, 2008, pp. 160-167.

[15] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, “Semantic compositionality through recursive matrix-vector spaces,” in *Proc. of EMNLP*, 2012.

[16] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning, “Dynamic pooling and unfolding recursive autoencoders for paraphrase detection,” in *Advances in Neural Information Processing Systems 24*. 2011.

[17] L. Nio, S. Sakti, G. Neubig, T. Toda, M. Adriani, and S. Nakamura, “Developing non-goal dialog system based on examples of drama television,” in *Proc. of IWSDS*, Paris, France, 12 2012.

[18] L. Nio, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, “Utilizing human-to-human conversation examples for a multi domain chat-oriented dialog system,” *IEICE Transactions on Information and Systems*, June 2014.

[19] G. Salton, A. Wong, and C. S. Yang, “A vector space model for automatic indexing,” *Commun. ACM*, vol. 18, no. 11, pp. 613-620, Nov. 1975.

[20] J. Turian, L. Ratinov, and Y. Bengio, “Word representations: A simple and general method for semi-supervised learning,” in *Proc. of ACL*, 2010.

[21] D.Liu, Z.Liu, and Q.Dong, “A dependency grammar and wordnet based sentence similarity measure,” *Journal of Computational Information Systems*, vol. 8, no. 3, pp. 1027-1035, 2012.

[22] D. Klein and C. D. Manning, “Accurate unlexicalized parsing,” in *Proc. of ACL*, Stroudsburg, PA, USA, 2003, pp. 423-430.