A Personal Conversation Assistant Based on Seq2seq with Word2vec Cognitive Map

A Personal Conversation Assistant Based on Seq2seq with Word2vec Cognitive Map

Maoyuan Shen
Graduate School of Computer and Information Sciences
Hosei University
Tokyo, Japan
maoyuan.shen.69@stu.hosei.ac.jp

***Abstract*—WeChat is one of social network applications that connects people widely. Huge data is generated when users conduct conversations, which can be used to enhance their lives. This paper will describe how this data is collected, how to develop a personalized chatbot using personal conversation records. Our system will have a cognitive map based on the word2vec model, which is used to learn and store the relationship of each word that appears in the chatting records. Each word will be mapped to a continuous high dimensional vector space. Then the sequence-to-sequence framework (seq2seq) will be adopted to learn the chatting styles from all pairs of chatting sentences. Meanwhile, the traditional one-hot embedding layer will be replaced with our word2vec embedding layer in the seq2seq model. Furthermore, an autoencoder of seq2seq architecture is trained to learn the vector representation of each sentence, then the cosine similarity between model generated response and the pre-existing response in test set can be evaluated , and the distance with principal component analysis (PCA) projection can be also displayed. As a result, our word2vec embedded seq2seq model significantly outperforms the one-hot embedded one.**

***Key Words*—WeChat; Word2vec; Seq2seq; Attention Mechanism; Autoencoder**

I. Introduction

Due to people's busy lifestyles nowadays, a chat assistant who can accurately predict the user's reply intention and chat on behalf of them will be greatly welcomed. This kind of desired AI can have the potential to be not just a chat pet, but also a virtual self that grows up with you in the parallel space. Although human-machine conversational AI has a long way to go through Turing tests [1], research in this field has always been pursued by researchers.

When we consider about Short-Text Conversation (STC), conventional methods can be classified into two categories [2]: Information Retrieval (IR) based method and the Statistical Machine Translation (SMT) based method. It has been shown that when a large corpus of status-response pairs is used in both methods, the SMT performance better than the IR [3]. Neural machine translation (NMT) is a new approach which is widely used in machine translation [4-6]. Meanwhile, it's also a very popular approach to build conversational models [2, 7-9]. With large scale of public social conversation data, those models are totally data-driven (i.e., without any manual rules) to generate plausible responses. Essentially, the NMT is a kind of SMT. It has a single neural network of encoder-decoder to map source sequence to target sequence, and the whole neural network is jointly trained to maximize the conditional probability of the source sequence generating the target sequence [4-7].

Sequence-to-sequence [5] with attention mechanism [6] is the most recently adopted architecture for conversational models [2, 7, 9]. It can capture the semantic and syntactic features between messages and responses in an end-to-end way [7]. Although this kind of model can generate some kind of reasonable responses to different types of questions and even extract knowledge from the training corpus, it has some obvious drawbacks [9]. One is the lack of a coherent personality, and another is the lack of general world knowledge and flexibility of using it. Therefore, I propose improvements on these two aspects.

The reason for the lack of coherent personality is that the training data from the open social conversations disturbed its character, because these conversations come from different participants. So I employ personal chat data decrypted from WeChat database and an open source single role response corpus[1] in model training.

Although seq2seq is a kind of semi-supervised learning, in fact it lacks flexibility in the mapping of sequences to sequences. When we look closely at the implementation details of seq2seq, we can see that it uses one-hot in word embedding. While one-hot embedding can just carry the frequency identity of a word rather than the information revealing its meaning. Such a sequence to sequence is just a monotonous match.

Since Google open source word2vec, it has quickly become a topic of general interest in the industry, because it's fast and efficient [10].

Actually, word2vec is a concise language model [11], which uses one layer neural network to map the one-hot form of word vector to the distributed form of the word vector. In order to accelerate the training, it uses some techniques like the hierarchical softmax, negative sampling, etc. [12] Word2vec representation has a number of excellent features. First of all, the word2vec representations of words carry the syntactic and semantic information [11], which makes the words of similar

Supervisor: Prof. Runhe Huang

[1] https://github.com/fateleak/dgk_lost_conv/blob/master/results/xiaohuangji50w_nofenci.conv.zip

meaning can be projected to a close distance in the word2vec space. This feature is much like a cognitive map of language built by humans through learning. On the other hand, it can compress the dimensionality of one-hot vector representation, which will be a very good feature for seq2seq model, because that will reduce the input units of the RNN. Due to its good performance, word2vec has been applied to comments sentiment classification [13], named entity recognition [14], and the visually grounded word embeddings (vis-w2v) to capture visual notions of semantic relatedness [15].

II. Related Work

Our proposed model is based on word2vec [11], seq2seq [5] and attention mechanism [6], so let's have a brief review of them.

A. Word2vec

Before the advent of word2vec, DNN has already been used to train word vector which is used to deal with the relationship between words and words. That method is a three-layer neural network structure (which can also be multilayered), which can be divided into input layer, hidden layer and output layer (softmax layer) [16].

Word2vec is usually divided into CBOW (Continuous Bag-of-Words) and Skip-gram models depending on how it defines the input and output of data.

The training input of the CBOW model is the corresponding word vectors for the context of a particular word, and the output is the word vector of this particular word. As the example shown in Fig. 1, our context values of 4 size, and this particular word is Learning, that is the term vector what we need to output. There are 8 words in the context, 4 in front and back, and these 8 words are input to the model. Since CBOW uses the word bag model, these eight words are all equal, that means the distance between them and the word we care about is not considered, as long as it's within our context.

The idea of the Skip-Gram model is reversed to CBOW, that means the input is the word vector of a particular word, and the output is the context vectors of the words corresponding to the particular word. For the example above, the context size is 4, the specific word Learning is the input, and these 8 words before and after are the output.

Word2vec uses CBOW and Skip-gram to train the model and get the word vector. It proposes an optimization method to replace the hidden layer and the neuron in the output layer with the data structure of the Huffman tree. The leaf node of the Huffman tree acts as the output layer neurons, and the number of leaf nodes is the size of the vocabulary. The inner node acts as a hidden layer of neurons.

B. Seq2seq and Attention Mechanism

This structure of seq2seq is also called the encoder-decoder mode [5]. We use $x = \{x_1, x_2, \ldots, x_{n_x}\}$ represents the input statement, $y = \{y_1, y_2, \ldots, y_{n_y}\}$ represents the output

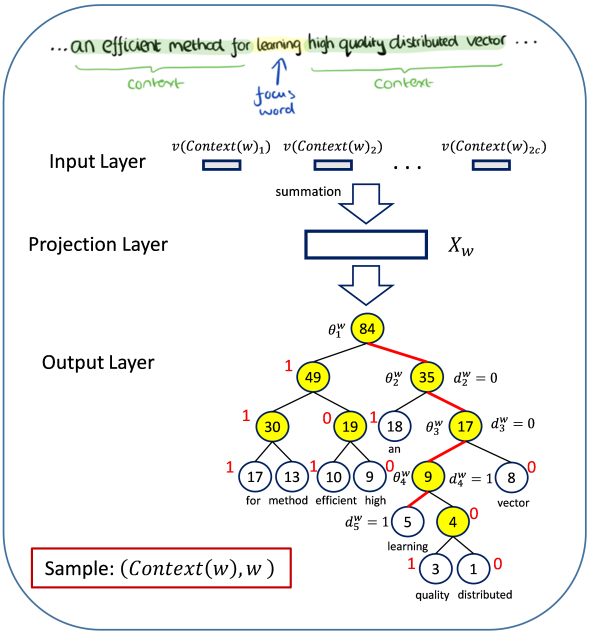


Fig. 1. Structure of word2vec (CBOW).

statement, and y_t represents the current output word. The goal is described by fomula (1):

$$p(y|x) = \prod_{t=1}^{n_y} p(y_t|y_1, y_2, \ldots, y_{t-1}, x) \tag{1}$$

As shown in fomula (2), our training objective is to minimize the negative log-likelihood of the conditional probability of the target y given source x:

$$J = \sum_{x,y \in D} -\log(p(y|x)) \tag{2}$$

Where D is our training set.

It's easy to model the distribution of the conditional probability $p(y|x)$ with seq2seq. The model includes encoder and decoder. First, in the encoder section, the input is transmitted to the encoder, and then the hidden state C of the last time step t is obtained, which is the basic function of Long Short-Term Memory (LSTM) cell. The C can be regarded as the context vector of the source sentence. Second, the context vector C from the encoder is entered as an initial state to the LSTM of the decoder. Then the decoder will calculate the output sequence, and the output from the previous moment y_{t-1} will be the input for this moment to calculate y_t.

In seq2seq model, the closer to the first cell of the decoder, the more influence it has on decoder, but it is not reasonable. The attention mechanism was then proposed [6], and for each cell in the output, the importance of each word in the input sequence will be detected. For each y_t in the output y, it's affected by a context vector c_t. The context vector c_t is an

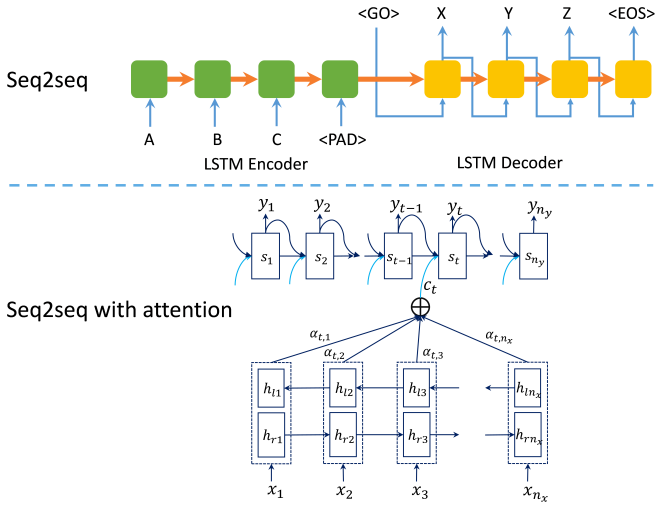


Fig. 2. Seq2seq and attention mechanism.

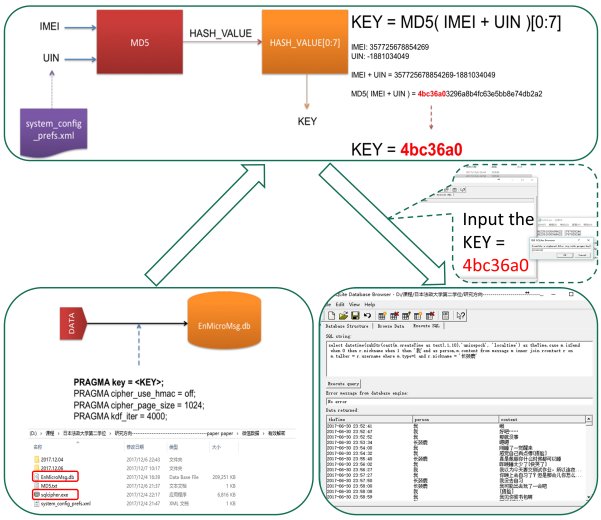


Fig. 3. Decryption of EnMicroMsg.db with SQLCipher.

information filter of all hidden states $h = \{h_1, h_2, \ldots, h_{n_x}\}$ of the encoder, it can be calculated as fomula (3):

$$c_t = \sum_{i=1}^{n_x} \alpha_{ti} h_i \tag{3}$$

Where α_{ti} is computed by fomula (4) and (5):

$$\alpha_{ti} = \frac{e^{e_{ti}}}{\sum_{j=1}^{n_x} e^{e_{tj}}} \tag{4}$$

$$e_{ti} = \text{align}(s_{t-1}, h_i) = v_a^T \tanh(W_a s_{t-1} + U_a h_i) \tag{5}$$

Where $v_a^T \in R^{n'}$, and $W_a \in R^{n' \times n}$, $U_a \in R^{n' \times 2n}$ are weight matrices, the n is the number of units in the LSTM cell, and the n' is the number of neurons in the hidden layer of alignment model. The align is an alignment model to evaluate he correlation between the input of position i and the output of position t.

III. Preliminary

A. Static Data Processing

The chatting records of WeChat is encrypted in the cell phone. Luckily, there is an effective method to decrypt it should be got. First, we should get the IMEI code of the smart phone, and find the UIN code in the file [system_config_prefs.xml] in the directory of WeChat APP. Second, a MD5 computation of IMEI+UIN should be made to get a KEY. Finally, the database can be decrypted with SQLCipher and the computated KEY. Fig. 3 illustrates the process of decryption.

B. Real-time Data Acquisition and Response with Itchat

Itchat[2] is a Python open source API to login the WeChat account(see Fig. 4). It can immediately get new WeChat messages and respond the message with the chatting engine.

[2] http://itchat.readthedocs.io/zh/latest/

Fig. 4. Login the WeChat account with itchat.

C. Dataset Preprocessing

Although the WeChat records can be obtained, the number of them is too small. So our experiments employ the open conversation corpus[1]. Our original conversation corpus has 500k one-round conversations. After removing duplicate and invalid data, the data is divided into the training set and the test set for each model. TABLE I gives the details.

TABLE I. Dataset preprocessing

Models	Training set	Test set
Word2vec	500k pairs of dialogs	
Autoencoder	395293 different sentences	
Word2vec embedded seq2seq One-hot embedded seq2seq	260047 pairs of dialogs	2000 pairs of dialogs

D. One-hot Embedded Seq2seq

One-hot Embedded Seq2seq is a common implementation by Tensorflow. One-hot embedding is a quantitative expression of a word. With it, any word can be transfered to a vector of the dimensionality of the vocabulary size, the values of the

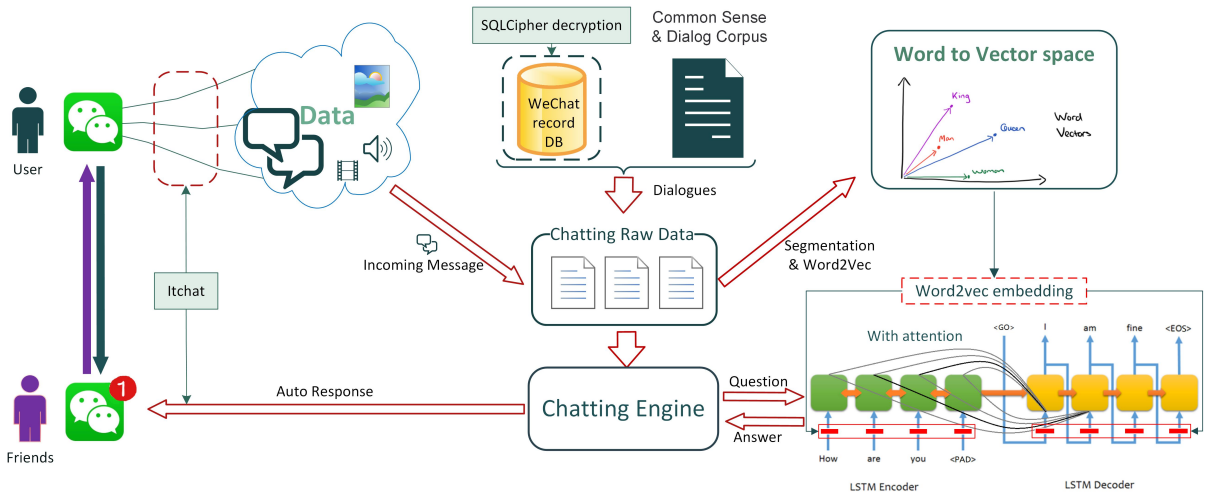


Fig. 5. Overview of coversational system based on seq2seq with word2vec cognitive map.

vector are a single high (1) and all the others low (0), and the position of (1) is determined by the unique identity of the specific word. The Fig. 6 is the structure of one-hot embedded seq2seq.

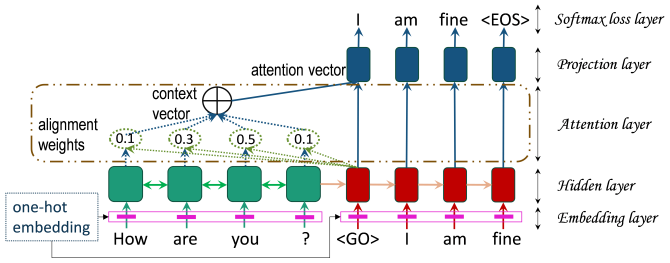


Fig. 6. Structure of one-hot embedded seq2seq.

IV. Proposed Method

Due to the excellent features of word2vec, it's proposed to build the embedding layer of seq2seq model.

A. Coversational System Based on Seq2seq with Word2vec Cognitive Map

In this paper, the seq2seq is combined with word2vec embedding to form a continuous learning and responding system. The system will consist of four autonomous modules whose operations ensure a continual learning from new data and auto responding with the trained seq2seq model. The four modules of the system are: real-time data acquisition and response, static data processing, word2vec and seq2seq. The first two modules are about Wechat data access, and the last two modules are about how to build the chatting engine, Fig. 5 shows the system diagram that has been built.

B. Word2vec Embedded Seq2seq

Word2vec embedded word will carry the semantic and syntactic information of the word. But word2vec embedded seq2seq will be harder to converge than the one-hot embedded seq2seq. For this reason, I build this structure with 2-layers encoder and decoder. As you can see in the Fig. 7.

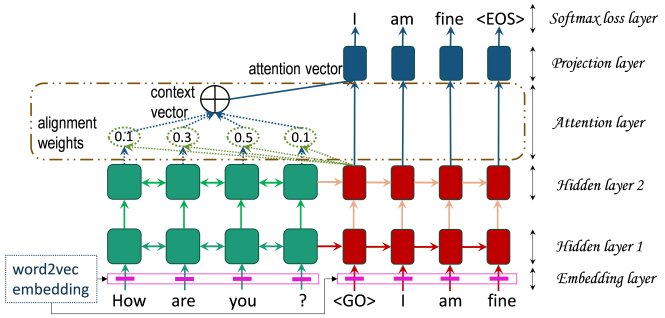


Fig. 7. Structure of word2vec embedded seq2seq.

C. Word2vec Embedded Autoencoder

Autocoder is a powerful tool to evaluate our results. It's built with the architecture of word2vec embedded seq2seq, but not with attention. In addition, it will trained with the same inputs and outputs, rather than the conversations. With this autoencoder, any sentence can be maped to a vector of a fixed dimensionality.

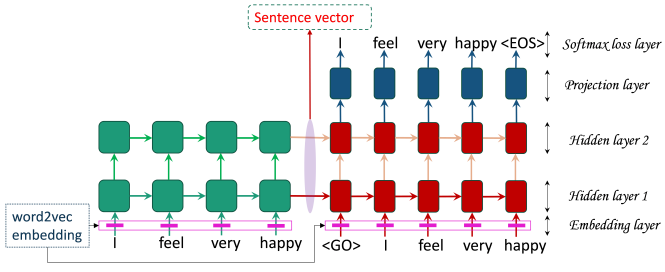


Fig. 8. Structure of the autoencoder.

V. Experiments and Result Analysis

In order to compare the models of the two structures on the task of generating response, they will be trained with the same dataset. In addition, a word2vec and an autoencoder are trained for embedding and evaluation.

A. Training Details

1) Word2vec: The key parameters and explanations for our training of word2vec are listed in TABLE II.

TABLE II. Word2vec parameters

Prameters	Explanations	Values setting
-cbow	Choice of training model (0: Skip-gram model 1: CBOW model)	1
-size	Dimensionality of the feature vectors	200
-window	Size of training window	8
-negative	Choice of training method (0: Hierarchical Softmax method ; > 0: Negative Sampling method)	25
-hs	Choice of training method (0: Negative Sampling method 1: Hierarchical Softmax method)	0
-sample	Threshold of sampling	1e-5
-threads	Number of running threads	20
-binary	Mode of storage (0: Common format 1: Binary format)	1
-iter	Number of iterations (epochs) over the corpus	15

2) Autoencoer: Each cell is set to 512 units and 204 dimensional word2vec embedding. The vocabulary number is 27411. Then the sentence can be mapped to a vector of 2048 dimensionalities.

The performance of autoencoder is tested with 4 pairs of similar sentences. As can be seen from Fig. 9, it can cluster sentences very well by the meaning.

3) One-hot Embedded Seq2seq: Because of the attention mechanism, one-hot embedded seq2seq is easy to converge. So one-layer bidirectional LSTM is used as the encoder. The vocabulary number is 22632.

4) Word2vec Embedded Seq2seq: When the word2vec embedding is introduced, the model becomes difficult to converge. By increasing the number of units in the LSTM cell to 1024 and using 2-layers bidirectional encoder, the model barely converges. In Fig. 10, it illustrates that word2vec embedded seq2seq requires longer training time, and the final loss is higher than the other two models.

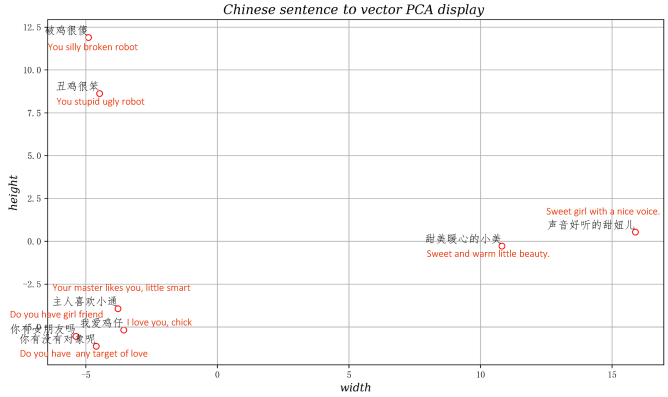
B. Model Analysis with Test Set

Our models are able to generate answers for questions from the 2,000 test samples. Fig. 12 shows the average cosine similarity between generated answers and the pre-existing answers of test samples.

The deviation vector is the difference of sentence vectors between the generated answer and the pre-existing answer. Fig. 11 shows the distribution of the deviation vectors from 2,000 test samples. It obviously shows the deviation vectors of word2vec model is more concentrated in (0,0). That means it has a lower margin of error with the standard answer.

Cosine similarity	破鸡很傻 You silly broken robot	丑鸡很笨 You stupid ugly robot	主人喜欢小通 Your master likes you, little smart	我爱鸡仔 I love you, chick	你有女朋友吗 Do you have girl friend	你有没有对象呢 Do you have any target of love	甜美暖心的小美 Sweet and warm little beauty.	声音好听的甜妞儿 Sweet girl with a nice voice.
破鸡很傻 You silly broken robot		0.4702260	0.1715385	0.1104738	0.2046397	0.1945348	0.1653126	0.1596077
丑鸡很笨 You stupid ugly robot	0.4702260		0.3512953	0.3061962	0.3090403	0.2805672	0.2689343	0.2851223
主人喜欢小通 Your master likes you, little smart	0.1715385	0.3512953		0.4556556	0.3805911	0.3020046	0.2749970	0.2758557
我爱鸡仔 I love you, chick	0.1104738	0.3061962	0.4556556		0.4193270	0.3525118	0.2704793	0.3212784
你有女朋友吗 Do you have girl friend	0.2046397	0.3090403	0.3805911	0.4193270		0.6771919	0.2552519	0.2603397
你有没有对象呢 Do you have any target of love	0.1945348	0.2805672	0.3020045	0.3525118	0.6771919		0.2617601	0.3103476
甜美暖心的小美 Sweet and warm little beauty.	0.1653126	0.2689343	0.2749970	0.2704793	0.2552519	0.2617601		0.5691882
声音好听的甜妞儿 Sweet girl with a nice voice.	0.1596077	0.2851223	0.2758556	0.3212783	0.2603397	0.3103476	0.5691882	

(a)



(b)

Fig. 9. (a) The cosine similarity between evaluation sentences; (b) The PCA projection reserves 47.75% information of original sentence vectors.

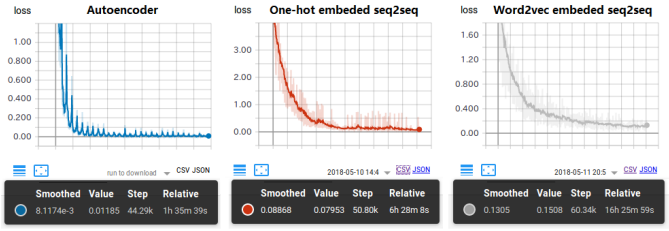


Fig. 10. Training process of three models.

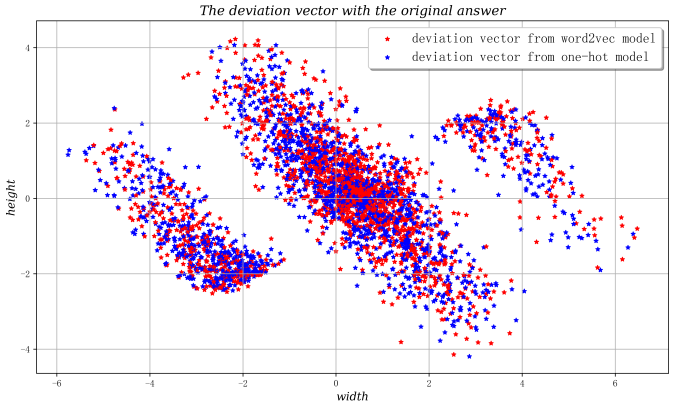


Fig. 11. The 2-dimensional PCA projection shows 18.02% information of original deviation vectors.

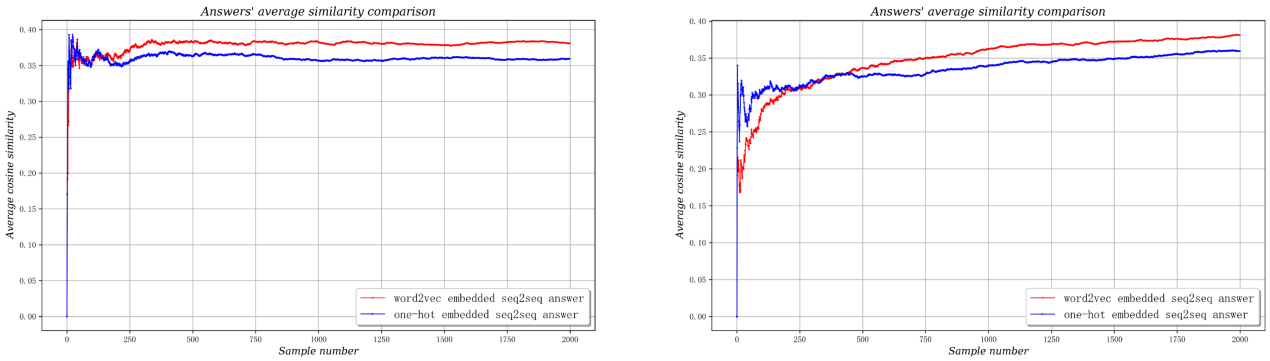


Fig. 12. In different order, the changes of the average cosine similarity between the answers generated by the 2000 test samples and the original answers of the test samples are shown . As shown in the figure, the word2vec embedded seq2seq outperforms the one-hot embedded seq2seq, and it also have a better performance in the longer questions. The average cosine similarity of word2vec is 0.381151, while one-hot is 0.359540.

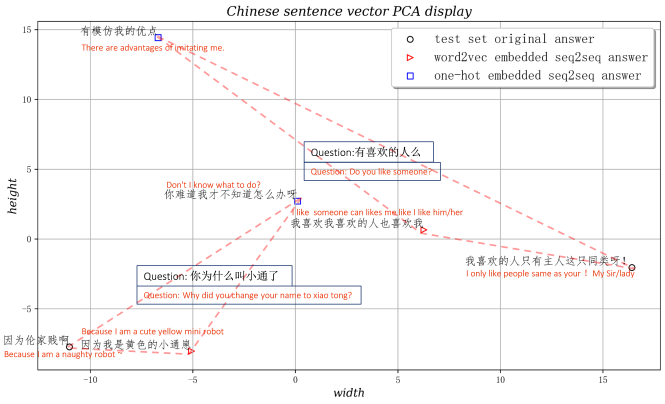


Fig. 13. The 2-dimensional PCA projection shows 55.67% information of original sentence vectors.

To get a more intuitive view of the results, two samples are shown as example. As shown in Fig. 13, the distance from answer of word2vec to the original answer is always the shortest edge of a triangle, and the two sentences also have the most similar meanings according to human understanding.

Thus, it can be concluded that word2vec embedded seq2seq is better than one-hot embedded seq2seq in conversation task.

VI. Conclusion & Future Work

With the empirical study of this paper, the proposed model performed well on the test dataset. It can generate appropriate and meaningful responses to questions it has never encountered, although it's harder to train. It can be speculated that word2vec provides better generalization capability for the seq2seq model, it provides a more continuous state variable space for the input, rather than fixed and isolated state changes of one-hot. That's why one-hot tends to converge more easily but deviates more from new questions. Generative Adversarial Networks (GANs) performs well in modeling generalization. I plan to use GANs to generalize the seq2seq in the future.

References

[1] C. Machinery, "Computing machinery and intelligence-am turing," *Mind*, vol. 59, no. 236, p. 433, 1950.

[2] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," *arXiv preprint arXiv:1503.02364*, 2015.

[3] A. Ritter, C. Cherry, and W. B. Dolan, "Data-driven response generation in social media," in *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2011, pp. 583–593.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[7] O. Vinyals and Q. Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869*, 2015.

[8] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan, "A neural network approach to context-sensitive generation of conversational responses," *arXiv preprint arXiv:1506.06714*, 2015.

[9] C. Xing, W. Wu, Y. Wu, J. Liu, Y. Huang, M. Zhou, and W.-Y. Ma, "Topic aware neural response generation." in *AAAI*, vol. 17, 2017, pp. 3351–3357.

[10] S. Lai, K. Liu, S. He, and J. Zhao, "How to generate a good word embedding," *IEEE Intelligent Systems*, vol. 31, no. 6, pp. 5–14, 2016.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[13] D. Zhang, H. Xu, Z. Su, and Y. Xu, "Chinese comments sentiment classification based on word2vec and svmperf," *Expert Systems with Applications*, vol. 42, no. 4, pp. 1857–1863, 2015.

[14] S. K. Sienčnik, "Adapting word2vec to named entity recognition," in *Proceedings of the 20th nordic conference of computational linguistics, nodalida 2015, may 11-13, 2015, vilnius, lithuania*, no. 109. Linköping University Electronic Press, 2015, pp. 239–243.

[15] S. Kottur, R. Vedantam, J. M. Moura, and D. Parikh, "Visualword2vec (vis-w2v): Learning visually grounded word embeddings using abstract scenes," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 2016, pp. 4985–4994.

[16] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5528–5531.