Comparative Analysis of Word Embedding Methods for DSTC6 End-to-End Conversation Modeling Track

Zhuang Bairong, Wang Wenbo, Li Zhiyu, Zheng Chonghui, Takahiro Shinozaki

Tokyo Institute of Technology, Japan

`www.ts.ip.titech.ac.jp`

Abstract

To improve the performance of the seq2seq model for the DSTC6 End-to-End Conversation Modeling Track, we make a comprehensive comparison of different word embedding techniques as the initialization methods of word vectors with the expectation that it helps the model to learn a better word representation for generating response utterances. We perform experiments using the official training and evaluation sets. We obtain the best result when we use fastText for the initialization. We further analyze the learned word embeddings by visualization using the t-SNE algorithm. It has been seen that there is a tendency that similar words in terms of several aspects tend to make a cluster in the embedding space when it is initialized using word2vec, while they scatter when a random initialization is used.

Index Terms: Seq2Seq model, dialogue generation, conversation modeling, word embedding

1. Introduction

Dialogue systems, also called interactive conversational agents or chatbots, can interact with human turns by turns and are applied to many situations, such as technical support services, entertainment, etc. The track 2 of the 6th Dialog System Technology Challenges (DSTC6), which is end-to-end conversational modeling, aims at generating natural and informative sentences in response to a user input using twitter data for model training and evaluation [1]. In the task, a baseline system using LSTM-based Seq2Seq model with a dialog state tracking mechanism as well as an official script to collect Twitter data are released, and the goal is to get higher BLEU scores.

According to our survey, there are 10 papers that used dedicated word embedding tools for word vector initialization in previous DSTC workshops. Most of the papers used word2vec [2, 3], excepting the one submitted by Xu [4] et al., where both word2vec and GloVe [5] were used. Their task was Spoken Language Understanding (SLU) of DSTC5. They used word2vec trained on Google News dataset, and GloVe trained on Wikipedia and Gigaword5 datasets. It is reported that these word embedding methods helped improving the performance. Since word embedding is also an essential part for the end-to-end conversational modeling of the track 2, we make comparisons introducing different word embedding methods to the baseline model, of which the word embedding is simply initialized using random numbers drawn from standard normal distribution.

The compared word embedding methods are word2vec, GloVe, and recently proposed fastText [6, 7]. We train these word embedding models using the Twitter dataset of DSTC6. Additionally, we analyze the learned word embeddings by visualization using the t-Distributed Stochastic Neighbor Embedding(t-SNE) algorithm [8].

The remainder of this paper is organized as follows. We first briefly review word embedding methods in Section 2, and the DSTC6 track 2 baseline system in Section 3. We then explain our system in Section 4. Experimental setup is explained in Section 5, and the results are shown in Section 6. Finally, the conclusions and future work are given in Section 7.

2. Word Embedding Methods

Word embedding has been widely used in natural language processing (NLP) tasks such as bilingual semantic representations [9], machine translation [10], spoken language understanding [11], conversational dialog systems [12, 13], sentiment classification [14] etc. Word embedding has a rather long history. Early in 1984, Hinton [15] put forward a concept called "distributed representations", which was the basic rudiment of word embedding study. Rowies et al. [16] published a new model named locally linear embedding (LLE) in Science in 2000 to computes low-dimensional, neighborhood-preserving embeddings of high-dimensional inputs. In 2003, Bengio et al. [17] tried to learn a distributed representation for words by neural network. In 2013, word2vec [2, 3] has been proposed by Mikolov et al., which has now become the mainstream of word embedding. In 2014, Pennington et al. introduced GloVe [5], which become also very popular. Last year in 2016, based on word2vec, Joulin et al. proposed fastText [7], which can handle subword units and is fast to compute.

The word2vec word embedding is based on skip-gram and continuous bag of words (CBOW). Assuming we have a sequence of training words $w_1, w_2, ..., w_T$ with length T, the objective of the skip-gram model is given by:

$$\arg\max_{\theta} \frac{1}{T} \sum_{t=1}^{T} \sum_{-C \leq j \leq C, j \neq 0} \log P_\theta \left(w_{t+j} | w_t \right), \quad (1)$$

where C is the size of training context, and $P\left(w_{t+j}|w_t\right)$ is a neural network with parameter sets θ taking a word represented by a one-hot vector as input and predicting a set of context words. The embedding word vectors are obtained as rows of a weight matrix of a hidden layer. CBOW is similar to the skip-gram model, but instead of predicting context with current word, CBOW predicts the current word base on context.

The fastText word embedding is contributed by the same group of people who established word2vec. It extends word2vec by introducing subword modeling. It represents each word w as a bag of character n-gram.

Given a dictionary of n-grams sized G, $\mathcal{G}_w \subset \{1, ..., G\}$ is the corresponding set of n-grams of w. The word w is then represented by the sum of its n-grams' vector representations. The advantages are that it can group inflected words and is robust for rare words, and the computation is fast.

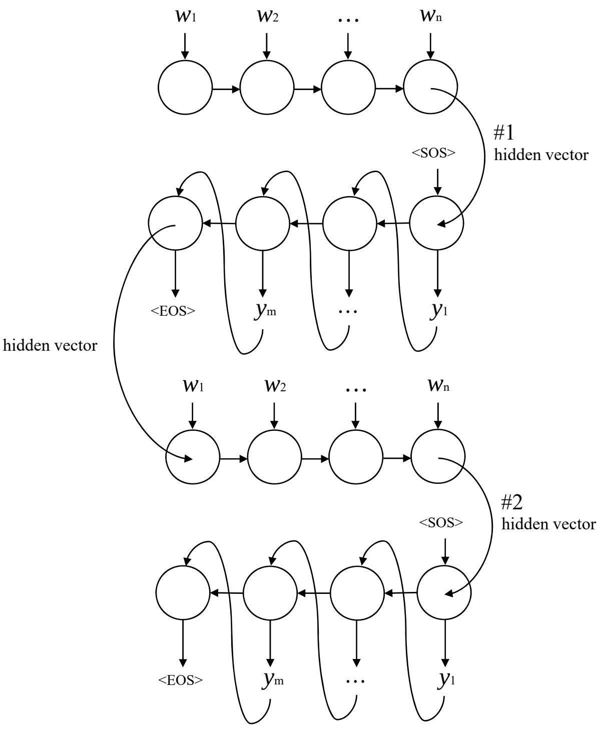

Figure 1: *seq2seq model used for DSTC6 end-to-end conversation modeling.*

GloVe, for Global Vectors, is a global log-bilinear regression model for word representations. The model is defined by:

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right) \left(\omega_i^T \widetilde{\omega}_j + b_i + \widetilde{b}_j - \log X_{ij}\right)^2 , \quad (2)$$

where V is the vocabulary size, $w \in \mathbb{R}^d$ are word vectors, $\widetilde{w} \in \mathbb{R}^d$ are separate context word vectors. X is the co-occurrance matrix, where X_{ij} represents the number of times word j appears in the context of word i. $f\left(X_{ij}\right)$ is a specific weighting function, b_i and $\widetilde{b}_j$ are additional biases. It combines the advantages of global matrix factorization and local context window methods.

3. DSTC6 Baseline System

The DSTC6 Track 2 baseline system is based on Sequence-to-Sequence model using LSTM encoder and decoder networks tracking a dialogue state as shown in Figure 1. Similar to previous works [18, 19], a dialog D is defined as $D = \{U_1, S_1, U_2, S_2, ..., U_n, S_n\}$, where U denotes a user input, and S denotes a system response, n denotes the turns of the conversation. Every U and S are further divided into a series of tokens, i.e. $U_n, S_n = \{w_1, w_2, ..., w_m\}$, where m denotes the m-th token in the n-th utterance. The token vocabulary includes normal words, punctuations, and symbols such as "<URL>" and "<USER>". The response generation is based on a conditional probability of the tokens that depends on the preceding utterances. The probability model of the utterances is factorized as:

$$P_\theta(S_n) = \prod_{n=1}^{N} P_\theta(S_n | U_{<n}, S_{<n}) \quad (3)$$

$$= \prod_{n=1}^{N} \prod_{m=1}^{M_N} P_\theta(w_m^{(n)} | w_{<m}, U_{<n}, S_{<n}), \quad (4)$$

where $U_{<n}$ denotes $\{U_1, U_2, ..., U_{n-1}\}$, $S_{<n}$ is $\{S_1, S_2, ..., S_{n-1}\}$, $w_{<m}$ is $\{w_1, w_2, ..., w_{m-1}\}$, and θ is a set of parameters. The specification of the baseline system is shown in Table 1.

Table 1: *Baseline hyper-parameter setting.*

Hyperparameter	Value
Vocabulary size	20000
# of Encoder Layer	2
Encoder embedding size	100
Encoder hidden size	512
# of Decoder Layer	2
Decoder embedding size	100
Decoder hidden size	512
Decoder projection size	100
Batch size	100
Optimizer	Adam
Dropout	0.5
Max length	30
Beam size	5

In the training, perplexity on validation set is monitored through the epochs, and the best model is selected. For the evaluation, definitions of official training and evaluation sets on Twitter dialogues as well as scripts to obtain BLEU scores are provided.

4. Our extended systems

The baseline system initializes the parameters of the word embedding layer using a standard normal distribution. Although the embedding layer is updated in the training process, the random initialization may not be optimal. To help the embedding layer to capture better knowledge about the tokens, we pre-train the layer by using advanced word embedding techniques. For this purpose, we use word2vec, fastText, and GloVe based word embeddings that are trained on the Twitter data. We also investigate different embedding vector sizes and projection layer sizes.

Additionally, analysis is given for the obtained word embeddings based on visualization using the t-SNE algorithm. The t-SNE algorithm converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data.

5. Experimental Setup

The Dataset of the track 2 task is a collection of Twitter conversations where each of the conversation is between a user and a consumer of various services given in English. The official data set was specified by the track 2 organizer, and each participant gathered the data from Twitter using provided scripts. Some information such as personal name and URL are automatically normalized to special tokens (e.g. <USER>, <URL>) by the provided scripts.

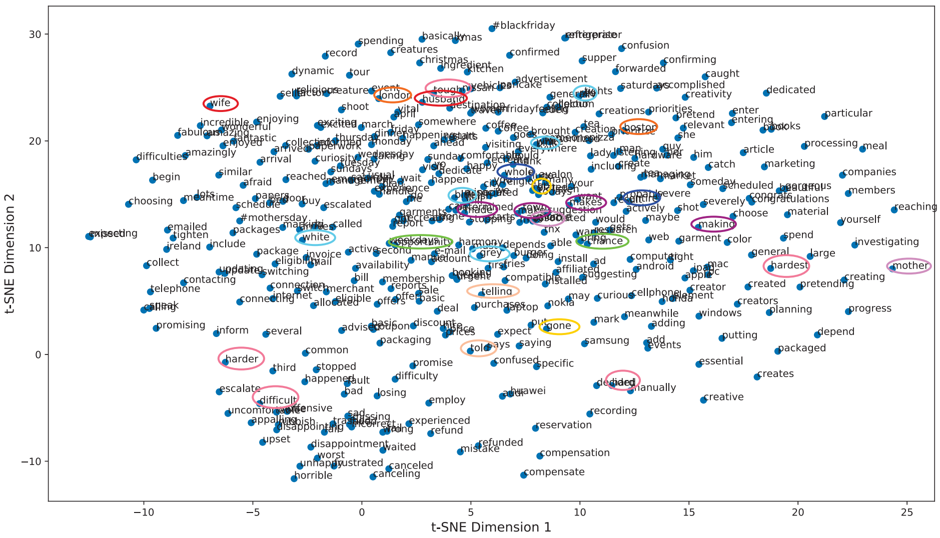

Figure 2: *Visualization of baseline word embedding for relatively frequent 400 words using t-SNE. The circles with the same color represent the words in the same group in some view point.*

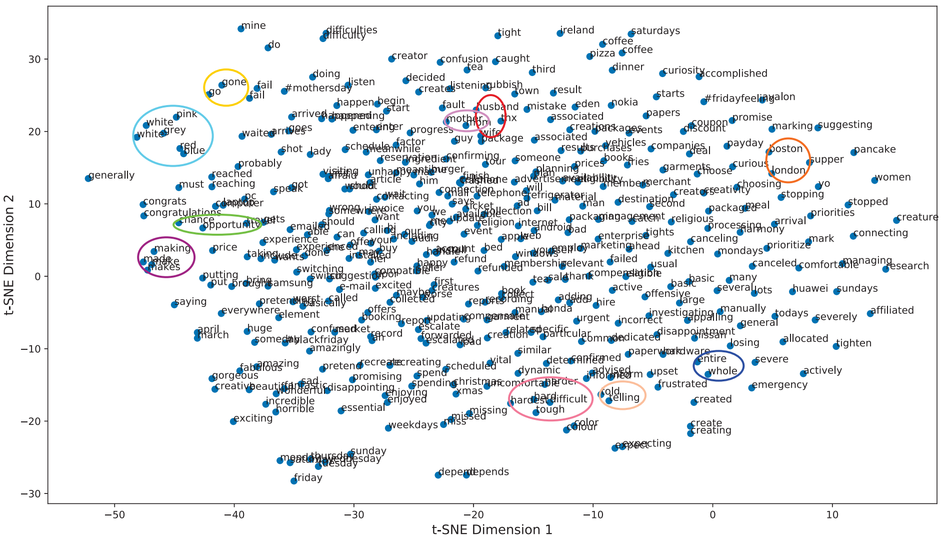

Figure 3: *Visualization of word embedding initialized by word2vec for relatively frequent 400 words using t-SNE. The circles with the same color represent the words in the same group in some view point.*

The official dataset that we have collected amounted to 22 GB. It contained about 887 thousand dialogues, and 1.08 million turns of conversations. All our experiments were based on this dataset excepting the one for the entry submission, which was performed using a subset that included 19 GB of the training data since the data was middle of collection. The results of our entry submission will be published from the track 2 organizer as team 4. The Seq2Seq neural networks were implemented using the Chainer Toolkit [20].

6. Results

Table 2 shows BLEU score results. In the table, the scores of the "Official Baseline" are those announced from the track organizer. The scores of "Team 4 Official Score" are the results of our entry submission evaluated by the organizer, where we used the subset of the official training set. "Self Baseline" indicates the results of the baseline system trained at our side using the full official training set. The rest of the results are based on the full official training set and the initialization of the word embedding layer by one of word2vec, fastText and GloVe.

Our official results gave better scores than the official baseline. When we trained the baseline system with varying conditions, we confirmed that choosing the embedding size to 100 gave generally the best BLEU scores. The tendency of the development set perplexity and the evaluation set BLEU was not consistent, where the perplexity was the smallest when the BLUE was the lowest. The same tendency was observed when we applied the word embedding initializations. This was maybe because of over-tuning of the network to the validation set. The perplexity tended to become small when the network size was

Table 2: *Comparison between different word-embedding techniques. ”Embd-Size” means the vector size of word embedding, and ”Prj-Size” indicates the dimension of the decoder’s projection layer. ”Val-PP” means perplexity on validation set. ”BLEU1˜4” means the BLEU scores for each model.*

Word Embedding	CBOW/SKIP	Embd-Size	Prj-Size	Val-PP	BLEU1	BLEU2	BLEU3	BLEU4	HumanRating
Official Baseline	-	100	100	-	0.230	0.109	0.064	0.039	3.364
Team4 Official Scoring	SKIP	300	100	-	0.234	0.112	0.065	0.040	3.443
SelfBaseline	-	100	100	20.2	0.234	0.112	0.066	0.040	-
	-	200	100	20.4	0.231	0.110	0.064	0.039	-
	-	300	100	20.1	0.226	0.108	0.063	0.039	-
word2vec	SKIP	200	100	20.0	**0.237**	0.112	0.065	0.039	-
	SKIP	200	200	20.0	0.230	0.109	0.064	0.039	-
	SKIP	300	100	20.1	0.232	0.109	0.062	0.037	-
	SKIP	300	300	20.0	0.230	0.110	0.064	0.039	-
	CBOW	200	100	20.1	0.233	0.113	0.066	0.041	-
	CBOW	200	200	19.7	0.233	0.110	0.064	0.038	-
	CBOW	300	100	20.2	0.226	0.106	0.062	0.037	-
	CBOW	300	300	19.6	0.232	0.111	0.065	0.039	-
fastText	SKIP	200	100	20.0	0.235	**0.115**	**0.068**	**0.042**	-
	SKIP	200	200	**19.5**	0.230	0.108	0.062	0.036	-
	SKIP	300	100	21.2	0.223	0.103	0.058	0.034	-
	SKIP	300	300	**19.5**	0.229	0.109	0.064	0.039	-
	CBOW	200	100	20.7	0.224	0.105	0.060	0.036	-
	CBOW	200	200	19.6	0.226	0.108	0.062	0.037	-
	CBOW	300	100	21.2	0.227	0.107	0.062	0.038	-
	CBOW	300	300	19.7	0.226	0.105	0.060	0.035	-
GloVe	-	200	100	20.0	0.232	0.111	0.065	0.039	-
	-	200	200	19.7	0.231	0.110	0.064	0.038	-
	-	300	100	20.0	0.230	0.108	0.062	0.037	-
	-	300	300	**19.5**	0.229	0.111	0.065	0.039	-

large.

For the word2vec initialization with skip-gram and CBOW, the fastText initialization with skip-gram, and the GloVe initialization, choosing embedding size to 200 and projection size to 100 was generally optimal. For BLEU 2, 3 and 4, the best results were obtained by fastText with skip-gram, which gave some improvement over the baseline.

For the Official Baseline and the Team 4 Official Scoring, human evaluation was also performed by the organizer where 10 different humans rated system responses in the range of 1 (very poor) to 5 (very good) and their scores were averaged. As shown in the table, our official submission results achieved higher human rating than the official baseline.

For an intuitive understanding of the learned word embeddings by the baseline model and the one initialized by the pre-trained word embeddings, we visualized the trained word embedding layer using t-SNE. Figure 2 is the result of the baseline system, and Figure 3 is the result using word2vec for the initialization. These figures are made using the parameters of the word embedding layers after training. For this analysis, we manually chose relatively frequent 400 words from the vocabulary. From the figures, it is observed that related words tended to aggregate in the model initialized by using word2vec, while they scattered in the baseline model. The followings are such examples that we observed.

- **Words within the same category**
  white/grey/pink/red/blue | wife/husband | boston/london | mother/mom
- **Words with different tenses**
  make/makes/made/making | told/telling | go/gone
- **Synonyms**
  hard/harder/hardest/tough/difficult | chance/opportunity | entire/whole

These results show that the initialization has a large influence on the distributions of words in the embedding space even after training.

7. Conclusion and Future Work

We have made a comprehensive comparison of using different word embedding techniques to initialize a word embedding layer of the seq2seq model for the conversation modeling track of DSTC6. While the difference from the baseline was not very large, the best result was obtained when fastText was used with the embedding size of 200 and the projection size of 100. The human rating score of our official submission results was also better than the baseline. When we visualized the word embedding layer after training, it was observed that related words in some senses (e.g., categories on meanings, tenses, synonyms) tend to form a cluster when word2vec was used for the initialization, whereas they were scattered when the random initialization was used.

Future work includes investigating more advance model structures (e.g. [18, 19, 21]) for better BLEU scores. We are also interested in automatically optimizing the structures based on evolutionary algorithms.

8. Acknowledgement

This work was supported by JSPS KAKENHI Grant Numbers JP26280055 and JP17K20001.

9. References

[1] C. Hori and T. Hori, "End-to-end conversation modeling track in dstc6," *arXiv preprint arXiv:1706.07440*, 2017.

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

[3] 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.

[4] G. Xu, H. Lee, M.-W. Koo, and J. Seo, "Convolutional neural network using a threshold predictor for multi-label speech act classification," in *Big Data and Smart Computing (BigComp), 2017 IEEE International Conference on*. IEEE, 2017, pp. 126–130.

[5] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[6] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.

[7] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[8] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[9] L. Wolf, Y. Hanani, K. Bar, and N. Dershowitz, "Joint word2vec networks for bilingual semantic representations," *Int. J. Comput. Linguistics Appl.*, vol. 5, no. 1, pp. 27–42, 2014.

[10] S. Liu, N. Yang, M. Li, and M. Zhou, "A recursive recurrent neural network for statistical machine translation," *Microsoft*, 2014.

[11] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *Interspeech*, 2013, pp. 3771–3775.

[12] A. Celikyilmaz, D. Hakkani-Tur, P. Pasupat, and R. Sarikaya, "Enriching word embeddings using knowledge graph for semantic tagging in conversational dialog systems," *genre*, 2010.

[13] J. Dodge, A. Gane, X. Zhang, A. Bordes, S. Chopra, A. Miller, A. Szlam, and J. Weston, "Evaluating prerequisite qualities for learning end-to-end dialog systems," *arXiv preprint arXiv:1511.06931*, 2015.

[14] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," in *ACL (1)*, 2014, pp. 1555–1565.

[15] G. E. Hinton, "Distributed representations," 1984.

[16] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[17] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[18] I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models," in *AAAI*, 2016, pp. 3776–3784.

[19] I. V. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. C. Courville, and Y. Bengio, "A hierarchical latent variable encoder-decoder model for generating dialogues," in *AAAI*, 2017, pp. 3295–3301.

[20] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a next-generation open source framework for deep learning," in *Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS)*, vol. 5, 2015.

[21] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," *arXiv preprint arXiv:1705.03122*, 2017.