International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 12 (2020), pp. 4251-4254
© International Research Publication House. http://www.irphouse.com

Word Embedding-based Text-to-Scene Conversion

So-Young Park, Bowon Choi, and Jaewook Lee

Game Design and Development, Sangmyung University, Seoul, Republic of Korea.

*ORCIDs: 0000-0003-0746-218X (So-Young Park), 000-0001-8329-5622(Bowon Choi),
0000-0001-8621-2060 (Jaewook Lee)*

Abstract

In this paper, we propose a text-to-scene conversion system using the word embedding, in order to alleviate the date sparse problem. Although some images are not stored in the word-image pairs, the proposed system can generate a scene with the most similar images, recommended by the word embedding. Considering construction cost, the proposed system adopts the natural language toolkit and the word embeddings, that can be automatically learned by machine learning. Experimental results show that the matching performance of the word embedding depends on the size of the training corpus. Specifically, the News word embeddings are learned from 100 billion tokens, while the Wikipedia and the Twitter word embeddings are learned from 6 billion tokens, and 2 billion tokens, respectively. Therefore, the News word embedding achieves 46.7% with the similar matching words in the top 1 while others take 33.3% and 20.0%.

Keywords: Text-to-Scene, Deep Learning, Word Embedding, Natural Language Processing, Word Representation

I. INTRODUCTION

Text-to-scene conversion system automatically generates a scene from an input text with the natural language processing [1]. It considers both a simple sentence and a complex sentence, as the input text. Also, it requires to construct resources such as lexicon and imagines. Many text-to-scene conversion systems have proposed, and they are classified into the following approaches: search-based approaches, rule-based approaches, and machine learning-based approaches.

First, the search-based approaches find a scene to illustrate the input sentence from a database by using keywords extracted from the sentence [2,3]. The database consists of the scenes and their descriptive sentences. These approaches can search the already completed scenes in the database, while they are limited to the scenes stored in the database [3]. When there is no relevant scene in the database, they cannot search the relevant scene [4].

Second, the rule-based approaches generate a scene by analyzing the input sentence according to rules [2,5]. These approaches can provide a much wider range of scene [3]; because they can create a new scene by merging some images, although there is no relevant scene stored in the database. However, it requires to construct elaborate resources, because it is difficult to analyze the input sentence, written in natural language.

Third, the machine learning-based approaches utilize some components, that can be automatically learned by machine learning [3,4,6]. These approaches are useful for natural language processing such as part-of-speech tagging, parsing, and named entity recognition. Therefore, the machine learning-based approaches can analyze various input sentences such as the simple sentence and the complex sentence [4]. Still, these text-to-scene approaches require every image corresponding to each keyword.

On the other hand, the word embedding represents a word into the vector of continuous real numbers with low dimensions, and it can be learned from a large unlabeled corpus [7]. In the word embedding vector space, the words sharing common contexts in the corpus are located close to one another [8,9]. In order to measure the similarity between two words in the word embedding vector space, it can utilize the cosine similarity formula describing the distance of two vectors [10].

In this paper, we propose the text-to-scene conversion system using the word embedding. Although some images are not stored in the word-image pairs, the proposed system can generate a scene with the most similar images, recommended by the word embedding. Section 2 introduces the proposed word embedding-based text-to-scene conversion system. Section 3 shows the experimental results of the proposed system. Section 4 concludes the paper.

II. WORD EMBEDDING-BASED TEXT-TO-SCENE CONVERSION

The proposed text-to-scene conversion system consists of a semantic structure analyzer, a word-image matcher, and a scene generator, as showed in Fig. 1. Given an input sentence composed of the words w_{1n}, the proposed system generates the $Scene$ to merge some images Img corresponding to the keywords in the semantic structure Sem, as described in the equation (3).

$$Scene = \underset{Scene}{argmax}\ Score(Scene \mid w_{1n}) \quad (1)$$

$$\overset{\text{def}}{=} \underset{Scene,Img,Sem}{argmax}\ Score(Scene, Img, Sem | w_{1n}) \quad (2)$$

$$\approx \underset{Scene,Img,Sem}{argmax} \begin{Bmatrix} Score(Sem \mid w_{1n}) \\ \times Score(Img \mid Sem, w_{1n}) \\ \times Score(Scene|Img, Sem, w_{1n}) \end{Bmatrix} \quad (3)$$

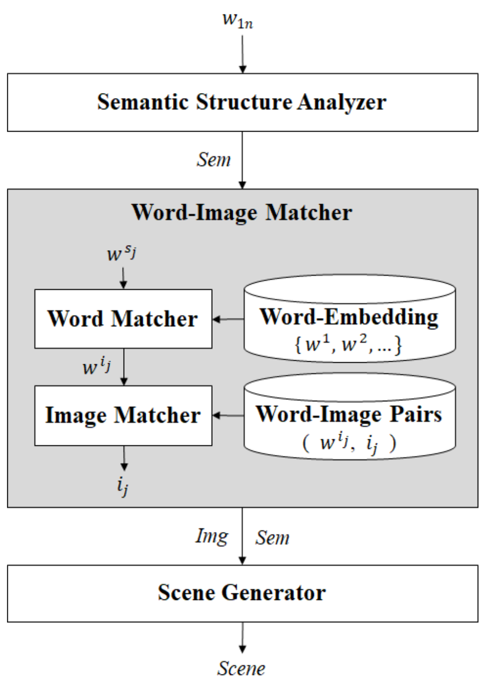


Figure 1. Proposed Text-to-Scene Conversion

First, the semantic structure analyzer fills in the slots of the semantic structure (who, do, what, where, when) with an appropriate word. In order to analyze various sentences, it utilizes some natural language processing tools such as a part-of-speech tagger, a named-entity recognizer, and a dependency parser [4]. Then, it selects the semantic structure of the best suitable clause describing a scene; because a complex sentence contains one or more clauses.

$$Sem = \underset{Sem}{argmax}\ Score(Sem \mid w_{1n}) \tag{4}$$

For example, there are three verbs in the sentence: "Hansel sneaks out of the house and gathers as many white pebbles as he can" [11]. The semantic structure analyzer can generate the semantic structures, and then select the semantic structure (who:Hansel, do:gather, what:pebbles, where:none, when:none), as the best suitable semantic structure.

Second, the word-image matcher finds the image i_j corresponding to every word w^{s_j} in the semantic structure from the word image pairs, where w^{s_j} indicates the word of the j-th slot in the semantic structure, and i_j indicates the image for the word w^{s_j}, as described in the equation (7). The equation (8) represents that both the image i_j and the word where w^{s_j} are independent of the contexts.

$$Img = \underset{Img}{argmax}\ Score(Img \mid Sem, w_{1n}) \tag{5}$$

$$\approx \underset{i_{1,m}}{argmax}\ Score(i_{1,m} \mid w^{s_1}, w^{s_2}, \dots, w^{s_m}) \tag{6}$$

$$\approx \underset{i_{1,m}}{argmax} \prod_{j=1}^{m} Score(i_j \mid w^{s_j}) \tag{7}$$

Considering the word image pairs without the word w^{s_j}, the word-image matcher utilizes the word embedding[7,8,9,10] to search the similar word w^{i_j} included in the word image pairs, as illustrated in Fig. 1 and the equation (9). The equations (10) and (11) describes that it calculates the cosine similarity between the vector of the word w^{s_j} in the semantic structure and the vector of each word w^{i_j} in the word-image pairs.

$$\underset{i_j}{argmax}\ Score(i_j \mid w^{s_j}) \tag{8}$$

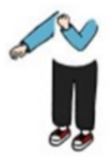
$$\overset{\text{def}}{=} \underset{w^{i_j}}{argmax}\ Score(w^{i_j} \mid w^{s_j}) \tag{9}$$

$$\approx \underset{w^{i_j}}{argmax}\ sim\left(\overrightarrow{w^{i_j}}, \overrightarrow{w^{s_j}}\right) \tag{10}$$

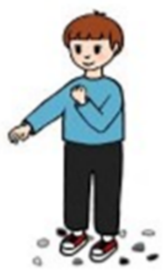
$$sim\left(\overrightarrow{w^{i_j}}, \overrightarrow{w^{s_j}}\right) = \frac{\overrightarrow{w^{i_j}} \cdot \overrightarrow{w^{s_j}}}{\left\|\overrightarrow{w^{i_j}}\right\| \left\|\overrightarrow{w^{s_j}}\right\|}$$

$$= \frac{\sum_{k=1}^{d}\left(v_k^{i_j} \cdot v_k^{s_j}\right)}{\sqrt{\sum_{k=1}^{d}\left(v_k^{i_j}\right)^2}\ \sqrt{\sum_{k=1}^{d}\left(v_k^{s_j}\right)^2}} \tag{11}$$

Given the semantic structure (who:Hansel, do:gather, what:pebbles, where:none, when:none) and the word-image pairs including the singular noun 'boy', the verb 'gather', and the plural noun 'stones', for example, the word-image matcher can find the images such as (a) of Fig. 2, although there is neither the Hansel image nor the pebble image in the word-image pairs.

who	do	what	where	when
Hansel	gather	pebbles	none	none
boy	gather 	stones	-	-

(a)



(b)

Figure 2. Scene Generated by Merging Images

Table 1: Word Embedding

Name	Method	Data Set	#Words	Dimension	File Size
News	word2vec	Google News (100 billion tokens)	3,000,000	300	1,662 MB
Wikipedia	Glove	Wikipedia 2014+Gigaword 5(6 billion tokens)	400,000	200	252 MB
Twitter	Glove	Twitter (27 billion tokens, 2 billion tweets)	1,193,514	200	758 MB

$$Scene = \underset{Scene}{argmax}\, Score(Scene|Img, Sem, w_{1n}) \approx \underset{Scene}{argmax}\; Score(\,Scene \mid Img, Sem\,) \qquad (12)$$

Third, the scene generator determinately yields the scene based on both the semantic structure and the images, as described in the equation (12); because the semantic structure specifies the location of each image in the scene. Given both the semantic structure and the images such as (a) of Fig. 2, for example, the scene generator yields the scene (b).

III. EXPERIMENTS

In order to examine the characteristics of the proposed text-to-scene conversion system, we implement the proposed system on Google CoLaboratory[12]. Specifically, the semantic structure analyzer utilizes the natural language toolkit NLTK [13], and the word-image matcher utilizes the Gensim library [14] in Python to load the word embedding. Besides, we extract the 15 keywords from the fairy tale "Hansel and Gretel", and construct the 60 word-image pairs previously constructed for 17 fairy tales such as "the three little pigs" and "the wolf and the seven young goats". Then, we evaluate them on Intel® Core™ i7-8565U CPU 1.80GHz 64bit and 16GB RAM.

For the purpose of analyzing the performance difference according to each word embedding, the word embeddings are prepared as shown in Table 1. First, the News word embedding indicates the 300-dimensional vectors learned from Google News including about 100 billion tokens for 3 million distinct words [14,15]. Second, the Wikipedia word embedding indicates the 200-dimensional vectors learned from Wikipedia 2014 and Gigaword 5 including approximately 6 billion tokens for 400,000 most frequent words [15,16]. Third, the Twitter word embedding indicates the 200-dimensional vectors learned from 2 billion Tweets including roughly 27 billion tokens for about 1.2 million distinct words [15,16].

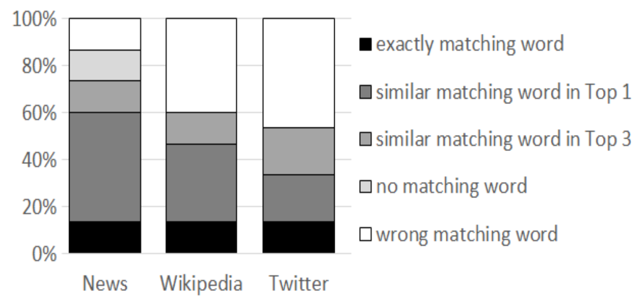


Figure 3. Matching Performance per Word Embedding

In the Fig. 3, the exact matching words indicate the number of exactly matching between the word in the semantic structure and the word in the word-image pairs. When, for example, the word 'bread' occurs in the clause "Hansel takes a slice of bread"[11], there is the word 'bread' in the word-image pairs. The matching result 13.33% is fair to all word embeddings; because the exact matching word results are independent of the word embedding to search the similar word.

The similar matching words in the top 1 indicate the number of correct matching between the word in the semantic structure and the most similar word recommended by the word matcher, where the similar word is one of the words in the word-image pairs. Because there is no image of the word 'pebbles' in the word-image pairs, the News word embedding recommends the word 'stones' in the word-image pairs, as the most similar word. The News word embedding achieves 46.7% while other two word embeddings obtain 33.3% and 20.0%, respectively. It describes that the large-scale dataset has a good effect on the quality of the word embedding.

The similar matching words in the top 3 indicate the number of correct matching between the word in the semantic and the top 3 similar words recommended by the word matcher. While there is no image of the word 'stepmother' in the word-image pairs, the News word embedding recommends the words 'woman' in the word-image pairs, as the third most similar word. For the word 'stepmother', the words 'girl' and 'boy' are more similar than the word 'woman'; because the word 'stepmother' shares common contexts with the words 'girl' and 'boy', rather than the word 'woman'. The News word embedding achieves 13.3% while other two word embeddings obtain 20.0%.

The no matching words indicate the number of no matching between the two words. The word matcher cannot recommend the similar word; because the recommend threshold is higher than the similarity between the word in the semantic structure and every word in the word-image pairs. For example, the word matcher with the News word embedding recommends no word for 'Hansel'; because the proper name 'Hansel' rarely appears in news articles.

The wrong matching words indicate the number of incorrect matching between two words. When the word 'witch' occurs in the clause "the witch cleans out the cage in the garden"[11], the Twitter word embedding recommends the words 'woman', 'girl', and 'cat', rather than the correct word 'crone' in the word-image pairs. The News word embedding takes 13.3% while other two word embeddings obtain 33.3% and 46.67%, respectively.

IV. CONCLUSION

In this paper, we propose the text-to-scene conversion system using the word embedding, in order to alleviate the date sparse problem. It consists of the semantic structure analyzer, the word-image matcher, and the scene generator, and it has the following characteristics.

First, the proposed text-to-scene conversion system employs some components, that can be automatically learned by machine learning. Experiments show that the semantic structure analyzer utilizes the natural language toolkit, and the word-image matcher adopts the word embeddings learned from Google News, Wikipedia, and Twitter.

Second, the proposed text-to-scene conversion system can create a scene with the most similar images, recommended by the word-image matcher, although some images are not stored in the word-image pairs. Experimental results show that the News word embedding achieves 46.67% with the similar matching words in the top 1 while it takes 13.3% with the similar matching words in the top 3.

Third, the matching performance of the word embedding depends on the size of the training corpus. Experiments show that the News word embeddings are learned from 100 billion tokens, while others are learned from 6 billion tokens, and 2 billion tokens, respectively. Therefore, the News word embedding achieves 46.7% with the similar matching words in the top 1 while others take 33.3% and 20.0%, respectively.

For the future works, we will improve the pretrained word-embeddings by using domain specific information. Because the pretrained word-embeddings are learned from the large-scale general corpus, it can be insufficient to solve the problem in the specific domain. Also, we will compare the characteristics of the word embedding according to languages.

ACKNOWLEDGEMENTS

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2018R1D1A1B07048122)

REFERENCES

[1] F. Yang, Z. Yuan, and X. Cheng, A Scene Division Method Based on Theme., *Proceedings of the International Conference on Advances in Computer Technology, Information Science and Communications*, 2019, 44-48.

[2] Baek, Seung-Cheol, Hee-Jin Lee, and Jong-C. Park, On the Automatic Generation of Illustrations for Events in Storybooks: Representation of Illustrative Events., *Proceedings of Korea HCI Conference*, 2008, 390-396.

[3] Ulinski, Morgan, Bob Coyne, and Julia Hirschberg. Evaluating the WordsEye Text-to-Scene System: Imaginative and Realistic Sentences. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, 2018, 1493-1499.

[4] Ji-Un Jeon, Do-Heon Choi, Soo-Hwan Jung, and So-Young Park, Illustration Generation System Using Deep Learning For Complex Sentences in Fairy Tale. *Journal of The Korean Society for Computer Game*, 32(2), 2019, 73-81.

[5] Jung-Ho Hong, Seung-Hee Cho, Ji-Un Jeon, So-Young Park. Development and Evaluation of Text-to-Scene Model for Korean Language Writing Education as a Foreign Language, *Journal of The Korean Society for Computer Game*, 31(3), 2018, 63-70.

[6] R. Johansson, A. Berglund, M. Danielsson, and P. Nugues, Automatic text-to-scene conversion in the traffic accident domain. *Proceedings of International Joint Conferences on Artificial Intelligence*. 2005. 1073-1078.

[7] Al-Saqqa, Samar, and Arafat Awajan. The Use of Word2vec Model in Sentiment Analysis: A Survey. *Proceedings of International Conference on Artificial Intelligence, Robotics and Control,* 2019, 39-43.

[8] Ma, Changyi, and Wenye Li, Sparse Binary Optimization for Text Classification via Frank Wolfe Algorithm. *Proceedings of the 12th International Conference on Machine Learning and Computing*. 2020. 171-176.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*. 2013, 3111-3119.

[10] Jatnika, Derry, Moch Arif Bijaksana, and Arie Ardiyanti Suryani. "Word2Vec Model Analysis for Semantic Similarities in English Words." *Procedia Computer Science,* 157, 2019, 160-167.

[11] Bushra Nafees Abad, Hansel and Gretel, *The Silver Book of Classic Tales: Every Child Should Read*, Notion Press, 2019, 19-22.

[12] Tock, Kalée. "Google CoLaboratory as a Platform for Python Coding with Students." *Proceedings of Robotic Telescopes, Student Research and Education*, 2(1), 2019.

[13] Steven Bird and Edward Loper. NLTK: The Natural Language Toolkit. *Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, 2004, 214–217.

[14] R. Rehurek, P. Sojka, Gensim-statistical semantics in python. Retrieved from genism.org, 2011.

[15] Kravchenko, Dmitry, and Lidia Pivovarova. "DL Team at SemEval-2018 task 1: tweet affect detection using sentiment lexicons and embeddings." *Proceedings of the 12th International Workshop on Semantic Evaluation*. 2018, 172-176.

[16] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." *Proceedings of the 2014 conference on empirical methods in natural language processing*. 2014, 1532-1543.