Web2Vec

Korea University COSE461 Final Project

Choi MinHyuk
Department of Computer Science
Team 16
2018320261

Kim SeohYoung
Department of Computer Science
Team 16
2018320259

Abstract

As the number of users of the World Wide Web (WWW) has increased significantly, numerous community sites have emerged. The online community is of great importance because the number of members and loyalty of the site are directly related to attracting advertisements, the most reliable profit model in the Internet business industry, and can be expanded to e-commerce such as shopping and auction. Therefore, it is necessary to know the characteristics of the online community itself, the relational characteristics with community members, and the relational characteristics between community sites. Each Internet community site has a clear feature, but it is not easy to identify this feature at once. However, it can be achieved by analyzing posts on Internet community sites using natural language processing (NLP).

1 Introduction

Until now, there were no attempts to embed various websites to the fixed-sized vectors like Doc2Vec. In this project, we collected data(corpus) from community websites, extracted useful information from each of them, formed a website vector and visualized them in 2D-dimension so that we can analyze the relationship and similarity between each website. Namely in this paper, we propose "Web2Vec". Like Word2Vec, where a word embedding is formed referring to neighbor words or Doc2Vec, where a documentary embedding is formed referring to sentences in the document, Web2Vec works same : It forms a website embedding referring to all corpus in the website, only bigger than Word2Vec and Doc2vec.

2 Related Work

- Text Classification

  Text classification is an important branch of natural language processing (NLP). It aims to learn and analyze classification rules using model algorithms for generalization and then applies the rules to unclassified datasets to achieve automatic classification of massive data.[1] Recently, natural language processing has been performed using language models such as Bidirectional Encoder Representations from Transformer (BERT), and Generative Pre-Trained Transformer 3 (GPT-3)], and these attempts have shown good performance in many areas. As a result of comparing the performance of the models pre-learned in Korean, KoBERT was 86.7%, showing the best performance compared to other models that improved the pretraining method and corpus configuration method.[2]

- Doc2Vec

  Doc2vec is an extended concept in word2vec, a word-based model. doc2vec propose Paragraph Vector, an unsupervised framework that learns continuous distributed vector representations for pieces of texts. The texts can be of variable-length, ranging from sentences to documents. In doc2vec model, the vector representation is trained to be useful for predicting words in a paragraph. each word is represented by a vector which is concatenated or averaged with other word vectors in a context, and the resulting vector is used to predict other words in the context. on sentiment analysis task, doc2vec achieve new state of-the-art results, better than complex methods, yielding a relative improvement of more than 16% in terms of error rate. On a text classification task, our method convincingly beats bag-of-words models, giving a relative improvement of about 30%.[3]

3 Approach

- Data collection using crawler

  We implemented python crawlers for each community site to collect data in this project. We obtained data from a total of five community sites(dcinside, pann, clien, ruliweb, theqoo) using crawlers. And we selected the five most popular bulletin boards in each community site. Data consisting of 100,000 sentences were obtained in the form of corpus on the five selected community sites and bulletin boards.

- Text Tokenization

  We tokenize corpus using kobert and okt. Tokenization is performed in three ways. first, Kobert tokenizes by considering the context of the overall sentence. Second, Okt extracts only non. Third, Okt extracts verb, and adverb to capture the subject or tone.

- Train using Doc2Vec

  Each of the tokenized corpus in three ways is trained as doc2vec. Then, three doc2vec vectors are created per community site, and they are concatenated by multiplying the desired weights.

- Text preprocessing

  The results obtained above are pre-processed to use for other model training. In order to put corpus in other train models, it goes through preprocessing processes such as deleting sentences that are too long or adding <CLS> and <PAD> tokens.

- NLP experiment

  NLP-related experiments are conducted using Pytorch's TransformerDecoder class. This experiment compares the method of using and not using the doc2vec vector obtained above.

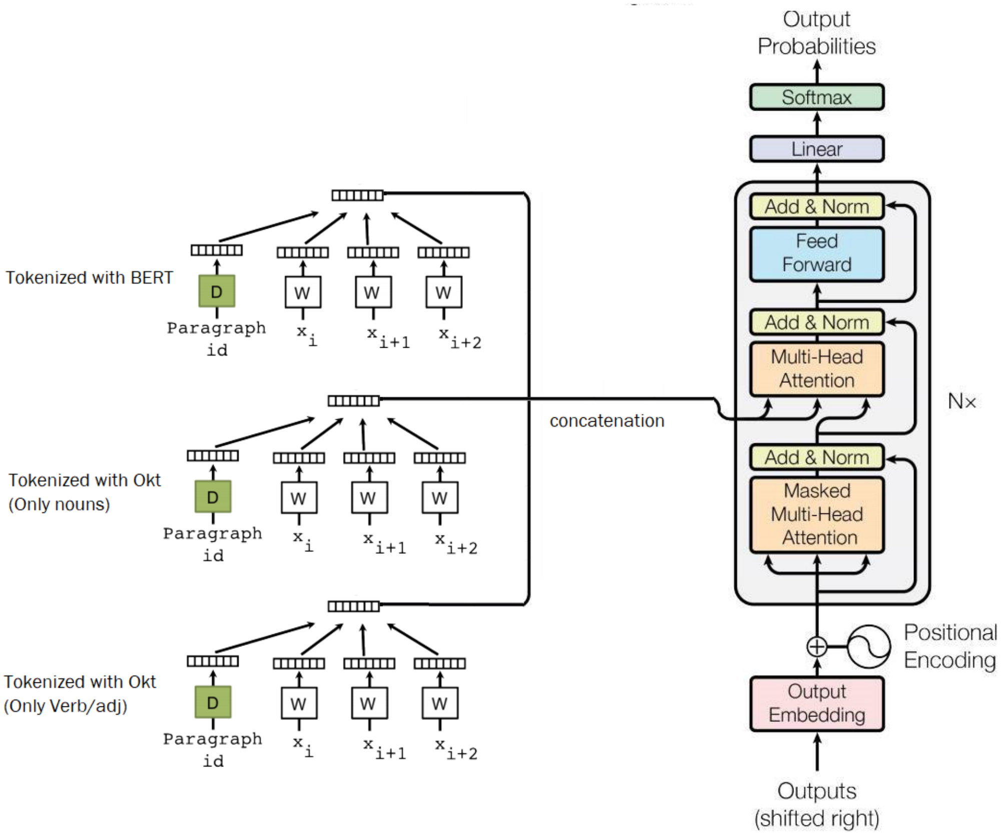


Figure 1: The flow graph of Project system

4 Experiments

4.1 Data

We collect a bunch of data from 5 famous websites(dcinside, pann, clien, ruliweb, theqoo), from which 20000 corpuses each are collected. And then, we tokenized those corpuses using KoBERT and Okt to extract contextual meaning, topic and general parlance of a sentence.

4.2 Evaluation method

To evaluate our Web2Vec vectors more clearly, we visualize vectors in 2d dimension reducing dimension using PCA and see and judge if similar websites are grouped. They seem well grouped by their style, topic or etc but we wanted to get more quantitative result and that's why we decied to adopt TransformerDecoder for language modeling or matching each article to corresponding websites.

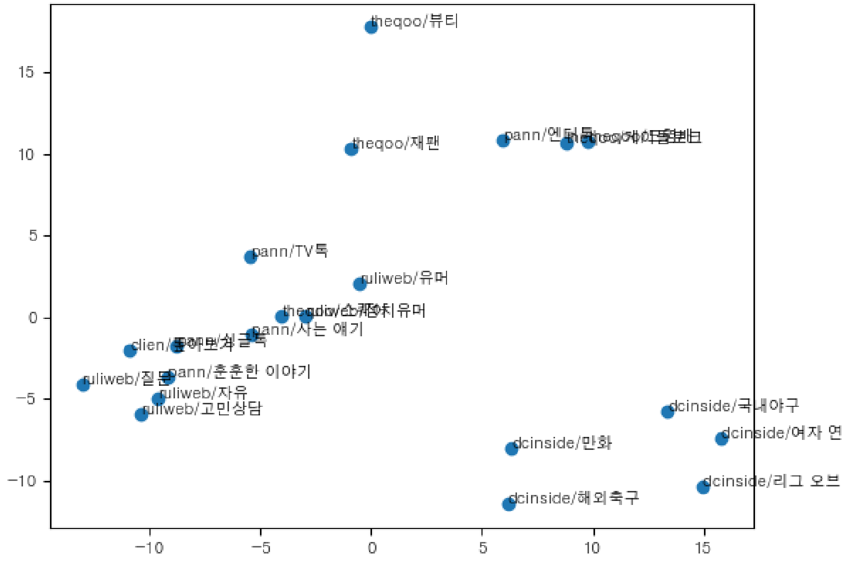


Figure 2: Web2Vec vectors in 2D space

4.3 Experimental details

We first set learning rate to 1e-3, but shows a very rapid decrease of Loss and of course, leads to overfitting so we changed it to 1e-6. Also we add regularization term (weight-decay parameter in Pytorch) to 1e-5 and increase dropout rate of TransformerDecoder from 0.1 to 0.5 to prevent overfitting because the loss graph clearly shows overfitting problem. For doc2vec training, it takes almost 6.5 hours for one group of tokenized data, so total about 20 hours are spent for the training. This result were derived from the fact that Gensim doesn't support GPU. On the contrary, TransformerDecoder of Pytorch supports GPU so it takes only 1.5 hours for training.

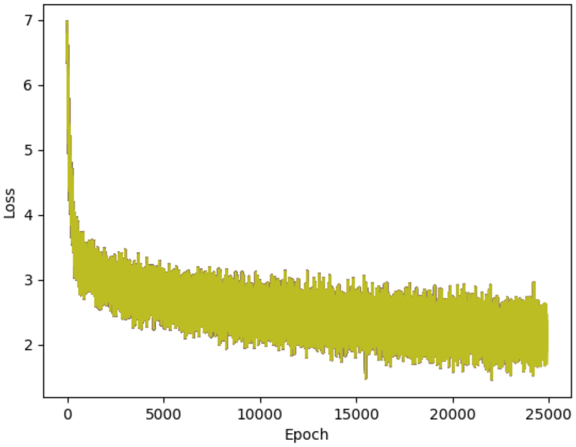


Figure 3: Loss graph without Web2Vec

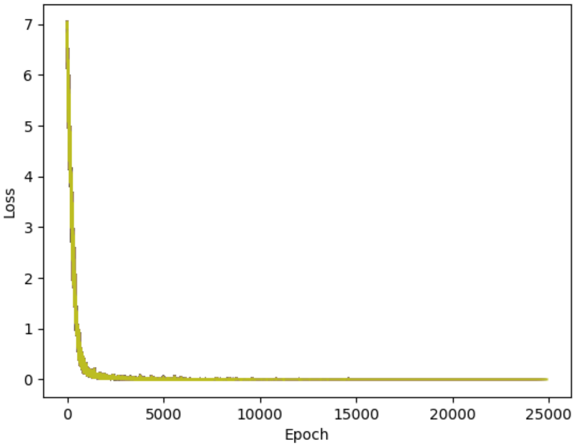
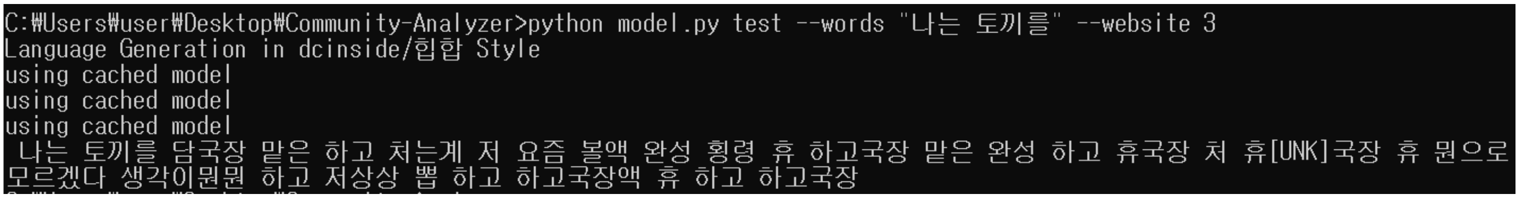


Figure 4: Loss graph with Web2Vec

4.4 Results

- Using Web2Vec vector as a hidden vector for TransformerDecoder caused an overfitting problem. We tried to solve this problem, like increading dropout, adding regularization term or modifying the model itself, but sadly we couldn't get the result where the accuracy of the model using Webvec is higher than the other one. The closest gap of accuracy was 0.245(with Web2Vec) and 0.275(without Web2Vec)
- We also tried to use it in Language Modeling, but we couldn't get a satisfactory result due to the lack of data. For example, we had only 100,000 corpus, but more than nearly 100,000,000 corpus are needed to train a proper and robust model.



```
C:₩Users₩user₩Desktop₩Community-Analyzer>python model.py test --words "나는 토끼를" --website 3
Language Generation in dcinside/힙합 Style
using cached model
using cached model
using cached model
 나는 토끼를 담국장 맡은 하고 처는게 저 요즘 볼액 완성 횡령 휴 하고국장 맡은 완성 하고 휴국장 처 휴[UNK]국장 휴 뭔으로
모르겠다 생각이뭔뭔 하고 저상상 뽑 하고 하고국장액 휴 하고 하고국장
```

Figure 5: Failed language modeling

5 Analysis

Web2Vec, which means project website information to a fixed sized vector, seems successful because similar websites are grouped well as you can see in Figure 2. However, two kinds of application of the vector failed owing to various reasons, for instance, lack of data.

6 Conclusion

Even thought we failed two experiments of application of vector to other NLP tasks, it has a potential to become more robust and precise if we collect much more corpus data and using more delicate tokenizer and training model. And we expect that certainly it can be applied to other various domains if it is trained in more data-affluent environment.

References

[1] Ding Weijie, Li Yunyi, Zhang Jing, and Shen Xuchen. Long text classification based on bert. *2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC), Information Technology,Networking,Electronic and Automation Control Conference (ITNEC), 2021 IEEE 5th*, 5:1147 – 1151, 2021.

[2] Jeiyoon Park, Mingyu Kim, Yerim Oh, Sangwon Lee, Jiung Min, and Youngdae Oh. An empirical study of topic classification for korean newspaper headlines. In *Annual Conference on Human and Language Technology*, pages 287–292. Human and Language Technology, 2021.

[3] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

A Appendix: Team contributions

- Choi minhyuk Managed general training process and models (corpus preprocessing, Doc2Vec, Transformer)
- Kim SeohYoung Wrote crawler modules for each websites and training models using GPU