Technology Review on Word2Vec

Fengyi Zhang

Introduction

The word2vec model and application proposed by Mikolov et al.[1,2] have attracted much attention in the past decades. The vector representations of words learned by word2vec models have been shown to carry semantic meanings and are useful in various natural language processing tasks. As the beginner of natural language processing, I would like to write a technical review on it to help other beginners in neutral network or non-experts in natural language processing to understand the mechanism of the model in simplicity and detail.

Overview

The use of vectors to represent words is not the first creation of Word2Vec.The earliest word vectors used One-Hot encoding, also known as one-bit valid encoding, where the size of each word vector dimension is the size of the entire vocabulary, and for each specific word in the vocabulary, the corresponding position is set to 1. The One-Hot encoding approach to representing word vectors is very simple, but the disadvantages are obvious: On the one hand, the vocabularies we use are large, often in the millions or more, and processing such high-dimensional data consumes a lot of computational resources and time. On the other hand, all the word vectors in One-Hot encoding are orthogonal to each other, and do not reflect the similarity relationship between words. Therefore, a distributed representation is proposed, which is based on the idea that each word of the original One-Hot coding is mapped to a shorter word vector through training. An interesting study shows that when representing our words with the word vector in the figure below, we can find:

$$\overrightarrow{King} - \overrightarrow{Man} + \overrightarrow{Woman} \approx \overrightarrow{Queen}$$

The reason is that the word vectors we use for the distributed representation contain contextual information. The word of the king and the queen have some semantic similarity and we often use them in similar context, and the way we train the distribution representation is based on such contextual information, which will be discussed in the following part. Thus, when we try to train a model of word representation, it evidently brings the contextual information into our model.

The training model of Word2Vec is essentially a neuronal network with only one hidden layer. Its input and output are both encoded with One-Hot. The weights from the input layer to the hidden layer are the word vectors using distributed representation. If the input is represented in $1 \times V$, where V is vocabulary size, the weight matrix is $V \times D$, where D is the dimension of distributed representation, after we perform a matrix

multiplication, we can get a new vector of 1 × D, which is a our dedicated word vector.

Continuous Bag-of-Words (CBOW)

Mikolov presents two algorithm to train the vector model. The CBOW architecture is a deep learning classification model in which we take in context words as input and try to predict the target word. Consider this example– "I love CS410", we will have pairs like ([I love], CS410) when the window size is set to 2. The deep learning model would try to predict the target word CS410 based on the context words. After we multiply the one-hot representation by the contextual and center weight matrix respectively, we pass these embeddings to a dense SoftMax layer that predicts the target word. We match this with our target word and compute the loss and then we perform backpropagation with each epoch to update the embedding layer in the process.

Skip-Gram

The skip-gram model is very similar to the CBOW model, the difference is given a target word, the context words are predicted. So, considering the same sentence – "I love CS410." and a context window size of 2, given the center word CS410, the model tries to predict [I, love]. Since the skip-gram model predicts multiple words from a single given word, we need several pairs of samples including both positive one and negative ones. Positive input pairs will have the training form of [(CS410, (I,love)),1] where label 1 indicates a relevant pair whereas negative samples will have a label of 0. However, instead of the actual surrounding words, we feed random contextual words along with the target words like [(CS410, BurgerKing), 0] to indicate an irrelevant pair.

Conclusion

In this review, we briefly talk about how the Word2Vec works. Given a large corpus of text, word2vec produces an embedding vector associated with each word in the corpus. These embeddings are well structured so that words with similar characteristics have fair cosine similarity to each other. The CBOW and the skip-gram model are the two main architectures used in Word2Vec, where given an input word, skip-gram will try to predict the contextual words to the input while the CBOW model will take the surrounding words and try to predict the missing one. In fact, in the realm of word embedding, we also have GloVe[3] based on co-occurrence matrix decomposition and BERT[4] based on transformer, an attention mechanism that learns contextual relations between words in a text. Word embeddings are always an essential part of solving many problems in NLP, and till now, the application of word embedding is very extended, including, information retrieval, sentiment analysis, topic classification and so on. I will continue to learn and research on such techniques so that we can make computer better understand our world and serve our life.

Reference:

[1] Mikolov, Tomas & Sutskever, Ilya & Chen, Kai & Corrado, G.s & Dean, Jeffrey. (2013). Distributed Representations of Words and Phrases and their Compositionality. Advances in Neural Information Processing Systems. 26.

[2] Mikolov, Tomas & Chen, Kai & Corrado, G.s & Dean, Jeffrey. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of Workshop at ICLR. 2013.

[3] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

[4] Devlin et al (2019) .BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding NAACL 2019