Embedding Space Creation

March 10, 2023

1 Embedding Space Learning

> "An embedding is a relatively low-dimensional space into which you can translate high-dimensional vectors. Embeddings make it easier to do machine learning on large inputs like sparse vectors representing words. Ideally, an embedding captures some of the semantics of the input by placing semantically similar inputs close together in the embedding space. An embedding can be learned and reused across models." − Machine Learning Crash Course with TensorFlow APIs

1.1 Recreating Word2Vec

1.1.1 What is Word2Vec?

Word2Vec [Mikolov, Tomas, et al. 2013a and Mikolov, Tomas, et al. 2013b] is a popular natural language processing technique that is used to create high-quality vector representations of words from large datasets of text. It is a neural network based model that is capable of capturing the semantic and syntactic meaning of words, and it has been widely used in various downstream NLP tasks such as text classification, sentiment analysis, and machine translation. Word2Vec has revolutionized the field of NLP by providing a more efficient and effective way to analyze and understand natural language text. In this document, we will provide a comprehensive overview of Word2Vec, its architecture, and recreate Word2Vec for our custom dataset.

1.1.2 Most Common Types of Methods for Word2Vec

There are two main types of methods used to create Word2Vec models:

- **Continuous Bag of Words (CBOW)**: In this method, the model predicts the target word based on the context words that surround it. The context words are used as input to the model, and the output is the probability distribution of the target word given the context words.
- **Skip-gram**: In this method, the model predicts the context words given a target word. The target word is used as input to the model, and the output is the probability distribution of the context words given the target word.

Both methods use a neural network architecture with one hidden layer to learn the vector representations of the words. The size of the hidden layer determines the dimensionality of the word vectors, and typically ranges from a few hundred to a few thousand. The Word2Vec models are trained on large corpora of text data using stochastic gradient descent, and the resulting word vectors are used in various NLP applications.

As for our use case in this Assignment, we are interested to create Knowledge Graphs, Topic Modeling and Entity-Relationship extraction as downstream tasks, we find that `Skip-gram` approach will be much suitable for us as underline it is trying to predict the context for a given word, where we can consider context as neighboring words for a given word. This will be very useful to us in establishing the strong relationships between different words.

1.1.3 Skip-gram

Skip-gram is a natural language processing technique used to create vector representations of words. As mentioned earlier, It is a type of Word2Vec model that learns to `predict the context words given a target word`. The basic idea behind Skip-gram is to use the target word as input to a neural network, and then predict the probability distribution of the context words that are likely to appear with the target word in a sentence.

The Skip-gram model takes a corpus of text as input, and creates a vocabulary of all the unique words in the corpus. Each word is represented by a vector of a fixed dimensionality (e.g., 100, 200, or 300). The Skip-gram model then trains a neural network on this vocabulary using a sliding window approach.

In this approach, a window of fixed size (e.g., 5) is moved across the text corpus, and for each target word in the window, the model is trained to predict the surrounding context words. This process is repeated for all target words in the corpus.

During training, the model adjusts the vector representations of each word in the vocabulary based on the prediction errors. After training, the word vectors are used to represent the semantic and syntactic meaning of words, and can be used in various downstream NLP tasks such as sentiment analysis, text classification, and machine translation.

Here are few examples of Skip-grams:

- Consider the sentence **"The quick brown fox jumps over the lazy dog"**. Using a window size of 2, the Skip-gram model would generate training pairs like (quick, The), (quick, brown), (brown, quick), (brown, fox), (fox, brown), and so on. The model learns to predict the context words (e.g., The, brown, fox) given a target word (e.g., quick).
- Let's say we are training a Skip-gram model on a corpus of movie reviews. The model might learn that the word "awesome" tends to appear in the context of positive sentiment words like "great", "fantastic", and "amazing", while it is less likely to appear in the context of negative sentiment words like "bad", "terrible", and "awful". This information can then be used to perform sentiment analysis on new movie reviews.
- Suppose we want to train a Skip-gram model to represent the semantic relationships between different animals. The model might learn that the vector representations of "dog" and "cat" are similar, while the vectors of "dog" and "snake" are dissimilar. This information can then be used to perform tasks such as animal classification or identification. **This example is very close to our use case in this Assignment** .

[ ]: