COMP6714 Information Retrieval and Web Search

Project 2

Author: Xinhong Wu

ZID: z5089853

Abstract

This Project is the implementing of Word-Embeddings for adjectives. The goal of this project is try to train a embedding model used to find synonym of a give adjectives. It is implemented spaCy for data processing and Tensorflow for model training.

Introduction

1. Data Processing and Training Data Generation

The data used for this project is BBC_Data.zip. By using the spaCy, the data from this file will be proceed, transformed and saved in a new file in the present working directory, which can later be used to train the embeddings.

2. Model Training:

Model training part is supposed to read out the data processed by part 1, and come up with best possible settings to train the embeddings. The model training is based on Tensorflow. The training part is basically like the training in word2Vec model. Just use a different batch generation.

3. Compute Top K

The Compute top k method read the embedding file by using genism. And then try to find top K most similar adjectives based on the embedding model. Return a list of top K most similar adjectives.

Methodology

The project constitutes three major parts listed below:

1: Data Processing and Training Data Generation

The data used for this project is BBC_Data.zip. By using the spaCy, the data from this file will be proceed, transformed and saved in a new file in the present working directory, which can later be used to train the embeddings.

The method used for processing the data:

1. Read the data from the file and put all documents in a string

2. Proceed the string by spaCy

3. Transform the data and write to a new file:

   - Read the string sentence by sentence, and then word by word

   For each sentence:

   - Skip no meaning part such as: "\n"
   - For word of adj., make no transformation.
   - For word of noun. or verb., lemmatization.
   - For word of some specific entities, replace the word with its entity type.
   - For word of some specific POS tags, replace the word with its POS tag.

   - After proceed each sentence, add a "EOS" into the output file.

2: Model Training:

Model training part is supposed to read out the data processed by part 1, and come up with best possible settings to train the embeddings.

The model training is based on Tensorflow. The training part is basically like the training in word2Vec model. Just use a different batch generation.

The batch generation:

If the centre word is adjective:

- Try to find the first verb. before centre word in buffer. If find, add to the word_to_use list.
- Try to find the first noun. after centre word in buffer. If find, add to the word_to_use list.

Use weighted random method to add other context word to word_to_use list based on its count. That is, the rare word will has higher probability to be choose and add to word_to_use list.

3: Compute Top K

The Compute_topk method read the embedding file by using genism. And then try to find top K most similar adjectives based on the embedding model. Return a list of top K most similar adjectives.

Results

The final tunable parameters:

1. Batch size: 120
2. Skip window: 2
3. Number of samples: 2
4. Vocabulary size: 4000
5. Learning rate: 0.002
6. Number of negative samples: 64

The average hits test on the dev data set: 10.55

Conclusion and Discussion

The processing part and batch generation part make a lot improvement of the result. The processing part greatly reduce the vocabulary noise of the data. And by transformation, we can get more general case of the sentence. The batch generation part which designed to generate batch more suitable for adjective.

The final parameters debugging part shows that the parameters have a great influence of the final embedding model. The test result become sound in low vocabulary size. The result gets better at learning rate of 0.002 than that of 0.001. That may because the number of iteration is not large enough to get convergence.