Homework 1

Aminul Huq
Student ID: 2019280161

1 Word2Vec Gradients Calculation

Given that,

$$\hat{y}_i = P(i|c) = \frac{exp(u_i^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)} \tag{1}$$

and we know Cross Entropy loss:

$$J = -\sum_{i=1}^{W} y_i log(\hat{y}_i) \tag{2}$$

Hence, we can say

$$J = -\sum_{i=1}^{W} y_i log(\frac{exp(u_i^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}) \tag{3}$$

$$J = -\sum_{i=1}^{W} y_i [u_i^T v_c - log(\sum_{w=1}^{W} exp(u_w^T v_c))] \tag{4}$$

As y is a one-hot encoded vector so it only contains 1 at the k^th index for which all the other indexes summation will be 0. So, the modified cost function will be.

$$J = -y_k [u_k^T v_c - log(\sum_{w=1}^{W} exp(u_w^T v_c))] \tag{5}$$

for $y_k = 1$
Now solving it for $\frac{\partial J}{\partial v_c}$

$$\frac{\partial J}{\partial v_c} = -[u_k - \frac{\sum_{w=1}^{W} exp(u_w^T v_c) v_w}{\sum_{w=1}^{W} exp(u_x^T v_c)}] \tag{6}$$

which can be arranged as:

$$\frac{\partial J}{\partial v_c} = \sum_{w=1}^{W} (\frac{exp(u_w^T v_c)}{\sum_{x=1}^{W} exp(u_x^T v_c)} u_w) - u_k \tag{7}$$

Using value of $\hat{y}_i$, equation (7) can be written as:

$$\frac{\partial J}{\partial v_c} = \sum_{w=1}^{W} (\hat{y}_w u_w) - u_k \tag{8}$$

Now we can u_k as Matrix vector multiplication: $U.y$
Also

$$\sum_{w=1}^{W} (\hat{y}_w u_w) \tag{9}$$

this is a linear transformation of vectors U scaled by $\hat{y}_w$. This can be written as $U.\hat{y}_w$.Therefore with the help of these two equations we can say equation (8) is:

$$U[\hat{y} - y] \tag{10}$$

u_i is assumed to as a column vector. For row vector it can be transposed and written as.

$$U^T[\hat{y} - y] \tag{11}$$

2 Word2Vec Implementation

In this task I have implemented skip-gram model with negative sampling for word2vec. Due to unavailability of computational resources I was unable to use the full 10GB dataset. I partitioned it into several parts and used 1GB of data from that. The total number of words and unique words present were 195621508 and 342505 respectively. Several preprocessing of the data was performed before putting it into skip-gram. Finally a CSV file is submitted where the spearman's co-efficient value converted from -1 to 1 to the range 0-10. In separate columns MAE error is also shown for each of the three different types of embedding's. Average difference score for 100,200 and 300 embeddings are 1.501,1.506 and 1.581 respectively. Attached .ipynb file has been used for generating all three embeddings. This same file was used and only the parameters were changed to get the results.

3 Word2Vec Improvement

In order to improve the performance of word2vec several additional components can be added to the embeddings. Parts of Speech Tagging or POS2Vec and Lexicon2Vec are among them.

POS2vec is the process of assigning to each word of a text the proper POS tag. The Part-of-speech gives large amount of information about a word and its neighbors, syntactic categories of words (nouns, verbs, adjectives, adverbs, etc.) and similarities and dissimilarities between them.

Lexicon2Vec are lists of phrases and words which have polarity scores and can be used to analyze texts. Each lexicon contains of words and their values which are the sentiment scores for that words. There are various sentiment and emotion lexicons that can be used, but choosing the proper lexicon is very important. According to Seyed Mahdi Rezaeinia, Ali Ghodsi and Rouhollah

Rahmani in their paper “Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis” they performed the same and got better results than only using word2vec embeddings.

I was unable to implement such methods and so I am mentioning the theoritical prospects of the same.