Word2Vec Skip-Gram Model Implementation

Ankur Debnath
M.tech (EE)
ankurdebnath@iisc.ac.in

Abstract

The implementation of **word2vec** using **skip gram model** was done on the Reuters corpus which is a prediction-based model for learning word embeddings. The model was fine-tuned by experimenting on different hyperparameters and model evaluation was done by calculating **Spearman's Correlation** ρ between cosine similarity and **SimLex 999** similarity task. The State-of-the-art performance of word2vec on Spearman's Correlation is 0.15-0.25. Also [1] Mikolov et. al. was implemented and claims about capturing relationships between words through the analogical reasoning task was verified.

1 Introduction

Word2Vec using skip-gram model is a shallow, two-layer neural network that is trained to reconstruct linguistic contexts of words. It takes inputs as skip-gram pairs of words in the entire corpus and produces a vector space, typically of several hundreds of dimensions, with each unique word in the corpus being assigned a corresponding vector in the space.Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.

The inputs to the system were the input embeddings $\boldsymbol{v}$ of dimensions $\boldsymbol{N}$ for $\boldsymbol{d}$ words in the dictionary from the corpus which were random at initialization and the hidden layer in the neural network generated embedding matrix W of size $d \times N$ where each row corresponds to the final word embedding to the corresponding word in the dictionary.

2 Implementation and Model Evaluation

A total of 2 tasks were performed, Task1 comprising of model generation,training and evaluation, whereas Task2 comprised of implementing [1] Mikolov et. al. using the best model from Task1.

2.1 Task 1 : Model Evaluation

In this, word2vec skipgram model was implemented on Reuters corpus. The model is a 2-layer neural network which takes inputs as word embeddings randomly initialised and hidden layer produces word embeddings for each word in the corpus and the output layer produces word embedding for the predicted word. Before training the data, preprocessing of raw data from the corpus was done including tokenization, removal of symbols, rare words, stopwords for better and clean training data.

Since the data was large, creating one-hot vectors was a computationally expensive task, hence inbuilt functions in Tensorflow was used to implement look-up table task. Experiments were done on different hyperparameters such as Batch Size, Embedding Size, Number of Negative Samples, epochs for better training on the data.The Loss funtion used in training was *Noise Contrastive Estimation(NCE)* Loss which inherently incorporates Softmax function and the optimizer was *GradientDescentOptimizer*

Model Evaluation was done on **SimLex-999** similarity task. After training the model and learning the word embeddings, word pairs common in both SimLex-999 and corpus were taken and Spearman's Correlation Coefficient was calcculated between the cosine similarity of word pairs from the corpus and SimLex scores and was tabulated in Table 1.

2.2 Task 2 : Paper implementation

[1] Mikolov et. al. was implemented using the trained model in Task 1 on the dataset ***question-words.txt***[2] which contains analogy tasks as word

Batch Size	Embedding Size	Negative Samples	Spearman's ρ
64	128	32	0.0684
64	128	64	0.1031
64	256	32	0.1456
64	256	64	0.0763
128	128	32	0.0913
128	128	64	0.1165
128	256	32	0.1129
128	256	64	0.0914

Table 1: Model parameters on various hyperparameters

pairs of different contexts such as country-capital, male-female etc. The task was to capture relationships between words. The model in Task-1 was trained on [2] and word embeddings were learnt by the model and then using euclidean distance between query word and all the words in the dictionary was computed to give top-k words, k being a hyperparameter.

Also the bias in the learning was tried to capture through considering nearest words around a given pair of words which was wagely conclusive. Theoretically, neural networks is a biased learning which when trained on [2] should result in biases such as bias on gender etc.

Figure 1: Biases on words 'boy' and 'girl'

```
In [36]: nearest_k('boy',10)
top-10 closest words

boy
aunt
grandson
sons
his
policeman
father
stepfather
prince
princess

In [37]: nearest_k('girl',10)
top-10 closest words

girl
mom
prince
bride
grandmother
aunt
princess
son
stepbrother
policeman
```

3 Experimental Results

In Task 1, the model was trained by varying hyperparameters and performance on SimLex-999 similarity task is tabulated in Table 1. According to the results the best ρ was 0.1456 for batch size 64, embedding size 256 and negative samples 32.

The same model has been used in Task 2 and trained on the [2] to get the biases as shown in Fig. 1. The results clearly show the biases on 'boy' and 'girl' where surrounding words belong to the same gender

Conclusion

The experiments verify the fact that word2vec skip gram model is an efficient representation for word embeddings which in turn not only captures the context but also able to capture the relationships with others words. The results also show that the model is consistent with the gold-standard Simlex-999 similarity task and hence is a rich and dense representation of words.

References

1.Mikolov et. al.
2.GitHub Link