Assignment 1: Implementing Word2Vec Skip-gram Model

Aniruddha Bala
Indian Institute of Science
Bangalore, India
aniruddhab@iisc.ac.in

1 Introduction

The primary objective of this assignment is to learn distributed word representations using word2vec skip-gram model as proposed by Mikolov et al. (2013a) and verify the results in the paper to the best possible extent. For this, different experiments are carried out by varying the hyper-parameters of the model such as batch size, number of negative samples, embedding size etc. The best model is chosen by observing the loss on train and validation sets and evaluating the model on Simlex-999 dataset. The best model so obtained is also evaluated on Semantic-Syntactic Word Relationship task (analogy task).

2 Dataset

For this assignment Reuters corpus is used to train the model. The Reuters corpus is a collection of 10788 news stories categorized into 90 different categories. The corpus is accessed through the nltk api. The api provides a default split for train and test sets containing 7769 and 3019 files respectively. For training the word2vec model I have used this default train-test split. 20 percent of training files are kept aside to form the validation set. These train and test files are further processed to obtain the source and target data for the model and the vocabulary. The corpus contains 33591 unique words, but this includes a lot of noise (e.g. numbers, one letter characters etc.). After pre-processing the data, the vocabulary formed is of size 15072. However, it may be a little less than this if subsampling is used. The inputs to the model are word and context word pairs formed from the Reuters corpus. The size of this data depends on the chosen window size and grows as we increase the window size. For model evaluation the Simlex-999 dataset (Hill et al., 2014) is used which contains the word similarity scores between word pairs for different types (nouns, verbs and adjectives). The set contains 333 samples from each type. For evaluating the model on analogy task I have used the the same data used in the word2vec paper by Mikolov et al. (2013b). Each line in the data consists of 4 words where the task is to predict the fourth word in relation to the third word by inferring the relationship between the first two words.

3 Model Architecture

The architecture resembles to the one proposed by Mikolov et al. (2013a) . The input to the skip-gram model is a list of center words and a list of targets i.e. their context words. Essentially it has three parameter tensors the embedding matrix, context weight matrix and context bias vector. The embedding matrix and context weight matrix are randomly initialized from uniform distribution and truncated normal distribution respectively. The bias vector is initialized with zeros. For a given training index an embedding lookup is performed to get the vector of the center word. To calculate the loss negative sampling is used as proposed by Mikolov et al. (2013b) in which for a given center word we maximize the similarity between the center and target context word and minimize the similarity between center word and negative sampled words. The negative samples are drawn from the unigram distribution over the vocabulary words with a distortion factor of 0.75. The embeddings are further normalized and and cosine similarity is calculated between the center word and other words. The whole network is trained using Gradient Descent Optimizer. For performing inference on the analogy task the inputs to the graph are word lists for word1, word2 and word3. The weights used for this task are the trained weights obtained after training the skip-gram model. Here

we find the similarity between the embeddings of the words in vocabulary and the vector vec(word2) - vec(word1) + vec(word3) and return the top k similar words.

4 Methodology

The model has been implemented using Tensorflow. The data for the model is prepared by removing noisy words by regex filtering and tokenizing the corpus into sentences. In addition to this subsampling of the data is done by removing stopwords or alternatively by using the subsampling method as proposed by Mikolov et al. (2013b). Also the words which occur less than a minimum threshold are removed. For the experiments I have used minimum threshold as 2. Using these processed sentences, the center and context word pairs are formed depending on the window size. The network is trained using gradient descent. For the trained network, the training, validation and test losses are reported. Furthermore correlation scores on the Simlex-999 dataset are also calculated by taking word pairs that are common in Simlex-999 data and vocabulary. The correlation is calculated between the word similarity scores obtained from the model and the word pair similarity scores obtained from the Simlex-999 dataset. The correlation is calculated separately for noun, verb and adjective pairs. The best model is selected by observing the training and validation losses and performance on the Simlex-999 task. The selected model is used for word analogy task and accuracy on the task is reported. For calculation of accuracy 1000 most common words are considered which are common in the question set and vocabulary. The fourth word is considered as predicted correct if it occurs in the top 100 predictions. This relaxation is done because the model has been trained on lesser data, as compared to the one proposed in paper and also for lesser epochs and hence the model is not confident enough to rank the correct word as the first prediction.

5 Experimental Setup

First set of experiments were carried out with 60 dimensional embeddings. Keeping the embedding size fixed, the other hyperparameters were varied and observations were made. The observations were made by varying the number of negative samples as 64, 128, and 200; batch size as 1, 16 and 64; window radius as 2, 3 and 5; and with and without subsampling. For subsampling the subsampling threshold was set to 1e-5 [3]. One experiment was carried out for two different learning rates 0.1 and 0.5, out of which the network was found to converge faster for learning rate of 0.5. Henceforth, all further experiments were carried out keeping the learning rate fixed at 0.5. Next, the embedding size was increased to 100 keeping the batch size as 1 and number of negative samples as 128 for which the performance was found to be better. Finally the best setting of hyperparameters was applied to 100 dimensional embeddings and results were obtained. The models have been evaluated based on the training, validation and test set losses and also the final correlation scores on Simlex-999 for noun, verb and adjective pairs and also the overall score considering all three.

6 Results

The result section contains tables showing the model performance for different hyperparameter combinations. For each experiment I have reported the training, validation and test losses and also the correlation scores on the simlex data (for Nouns, Adjectives, Verbs and Overall). The figure below shows the decreasing training and validation loss as a function of number of samples seen so far. The figure below corresponds to the setting batch size=1, window radius=5, number of negative samples=128, and embedding dimension=60, with stopwords removed.

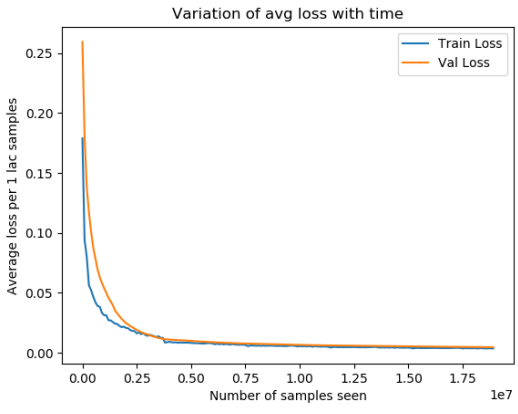

Figure 1: Variation of training and validation loss with time

7 Observations and Conclusions

From the first table we see that as we increase the batch size the number of epochs required to train

Batch Size	Epochs	Avg Loss			Simlex Correlation Scores			
		Train	Val	Test	Nouns	Adjectives	Verbs	Overall
1	3	0.005112	0.006115	0.006041	0.0351	0.0598	0.0562	0.0481
16	30	0.006462	0.008337	0.008271	0.0049	-0.0329	-0.0787	-0.0202
64	50	0.011240	0.013663	0.013700	-0.0035	0.0296	0.0649	0.0232

Table 1: Effect of varying the batch size keeping other hyperparameters fixed (win_radius=5, neg_samples=128, embed_dims=60).

Sampling	No. of words	Avg Loss			Simlex Correlation Scores			
		Train	Val	Test	Nouns	Adjectives	Verbs	Overall
No	6.3 M	0.004483	0.004844	0.004914	0.1234	-0.0407	-0.0374	0.0539
Yes	6.1 M	0.005112	0.006115	0.006041	-0.0173	0.1084	0.0614	0.0273
Stopwords removed	3.8 M	0.005112	0.006115	0.006041	0.0351	0.0598	0.0562	0.0481

Table 2: Effect of sampling keeping other hyperparameters fixed (epochs=3, batch_size=1, win_radius=5, neg_samples=128, embed_dims=60). Yes implies the sampling technique proposed in [3] was used

Epochs	Avg Loss			Simlex Correlation Scores			
	Train	Val	Test	Nouns	Adjectives	Verbs	Overall
3	0.005112	0.006115	0.006041	0.0351	0.0598	0.0562	0.0481
5	0.003642	0.004673	0.004615	0.0279	0.0940	0.0409	0.0425

Table 3: Effect of increasing number of epochs keeping other hyperparameters fixed (batch_size=1, win_radius=5, neg_samples=128, embed_dims=60).

Window radius	Avg Loss			Simlex Correlation Scores			
	Train	Val	Test	Nouns	Adjectives	Verbs	Overall
2	0.007577	0.010272	0.010330	0.0439	-0.0071	-0.0065	0.0246
3	0.006350	0.008007	0.007939	-0.0130	-0.0968	0.1417	0.0199
5	0.005112	0.006115	0.006041	0.0351	0.0598	0.0562	0.0481

Table 4: Effect of varying the window radius keeping other hyperparameters fixed (epochs=3, batch_size=1, neg_samples=128, embed_dims=60).

the model are more and performance on the simlex task degrades. One possible reason for this can be that we use the same k negative samples for all the training samples in the batch so there exists some probability that the sampled candidate belongs to the true classes. This problem escalates with the increasing batch size and hence the updates become noisy. However, this problem can be solved by obtaining k different negative samples for each sample in the batch and ensuring that it does not match with the true labels, but this comes with an added computation cost. If batch size is one there is smaller chance of the single label getting sampled from the unigram distribution. Hence we proceed with batch size 1. Table 2 shows the effect of sampling. The effect of sampling is not so prominent, but removal of stopwords results in relatively lesser number of tokens and perfomance is also better. Therefore for further experiments stopwords were removed. In table 3 I have tried to increase the number of epochs by 2 for single batch size. Though there is a decrease in train-

No. of negative samples	Avg Loss			Simlex Correlation Scores			
	Train	Val	Test	Nouns	Adjectives	Verbs	Overall
64	0.004178	0.005013	0.004975	-0.0750	-0.1069	-0.0144	-0.0560
128	0.005112	0.006115	0.006041	0.0351	0.0598	0.0562	0.0481
200	0.004069	0.005307	0.005261	-0.0207	0.1073	0.0228	0.0162

Table 5: Effect of varying the number of negative samples keeping other hyperparameters fixed (epochs=3, batch_size=1, window_radius=5, embed_dims=60).

Embedding size	Avg Loss			Simlex Correlation Scores			
	Train	Val	Test	Nouns	Adjectives	Verbs	Overall
60	0.005115	0.006081	0.006067	0.0351	0.0598	0.0562	0.0481
100	0.005039	0.006541	0.006493	0.1126	0.1586	0.1306	0.1264

Table 6: Effect of varying the embedding size keeping other hyperparameters fixed (epochs=3, batch_size=1, win_radius=5, neg_samples=128).

Model	**Accuracy**
60-d	0.21
100-d	0.08

Table 7: Accuracies on the analogy task considering the 1000 most common words in the question set and vocabulary and considering top 100 words for the calculation of accuracy.This accuracy is on 52 questions obtained after filtering.

ing loss but there is no substantial improvement in the Simlex-999 scores. In table 4 we see that as we increase the window radius the performance improves in general. This is expected because as we include more context the word representations learnt should be better. In table 5 we see the effect of number of negative samples on model performance. It is seen that we get better performance by setting the number of negative samples as 128. In [3] the authors train their models on 5-30 negative samples because their training set size is larger than the one being used here. The training set used by the authors consists of around billion tokens whereas here we have trained our models on million token corpus. Also in the paper they suggest to use smaller number of negative samples around 5-20 for large training corpus and larger number of negative samples for smaller corpuses. Next in table 6 we see the effect of increasing the embedding dimension to 100. Doing this results in significant improvement in the Simlex-999 scores. Finally I take the models listed in table 6 and apply them to the word analogy task. It is observed that contrary to our expectation the 60-d model outperforms the 100-d model.

8 Visualizations

The visualization section contains few images that were visualized using pca. The images were visualized online on (http://projector.tensorflow.org/) by uploading the learnt embeddings and labels. The scatter plot for the images represents words in 2D space that are closer to the given word. The word that is larger in font is the chosen word. More red a point is the closer it is in terms of cosine distance to the chosen word. The visualizations show that the there is an inherent bias in the learnt word vectors. If we see the nearest neighbours of company and one (figure 2 and 3) the words are well separated from the center black cluster and the closest words predicted also make sense. But if we oberve that of king and queen (figure 4 and 5) the words are clustered near the black cluster and the model is not able to learn good representations for these. The reason for this can be attributed to the skewness in the dataset. The words king and queen occur very infrequently in the data whereas the words like one occur very frequently (figure 6). To visualize the learnt relationships for the word analogy task I have plotted

few of the country names along with their capitals. The lines are parallel, or make acute angles to each other; from this we can say that there is high similarity between the three word pairs. Finally I have added some outputs obtained while testing and running the analogy task.

Figure 2: Closest words for company

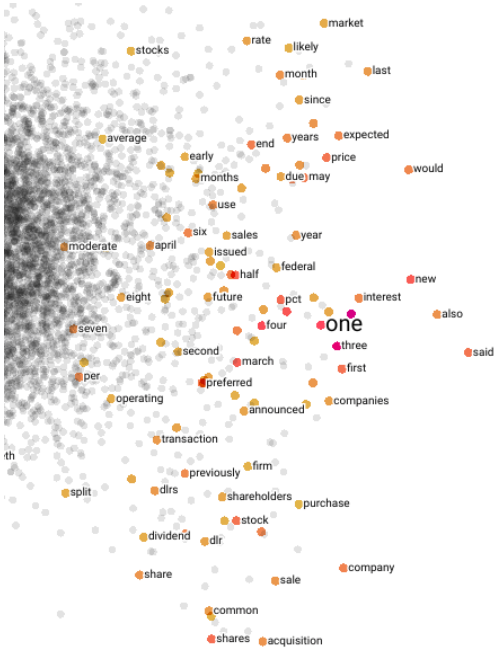

Figure 3: Closest words for one

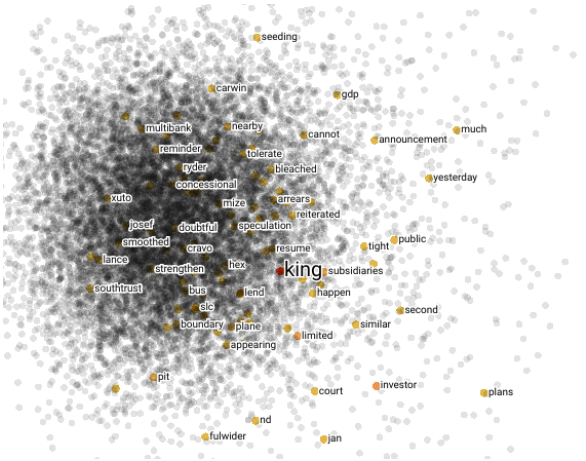

Figure 4: Closest words for king

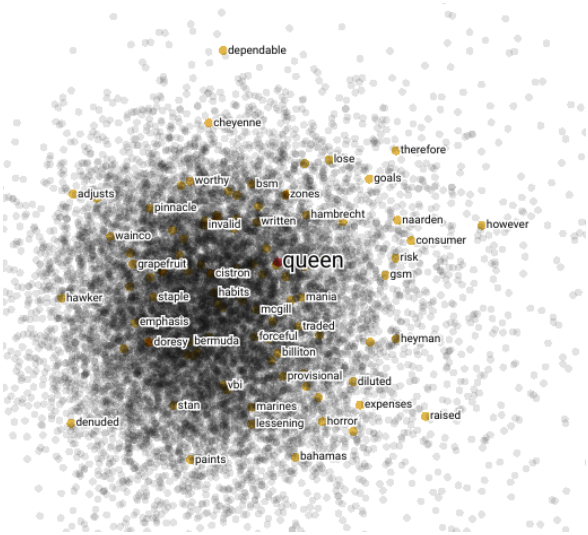

Figure 5: Closest words for queen

```
>>> vocab['king']
23
>>> vocab['queen']
3
>>> vocab['man']
21
>>> vocab['women']
7
>>> vocab['one']
1791
>>> vocab['two']
1592
>>> vocab['u.s.']
3199
```

Figure 6: Counts of some words in the corpus

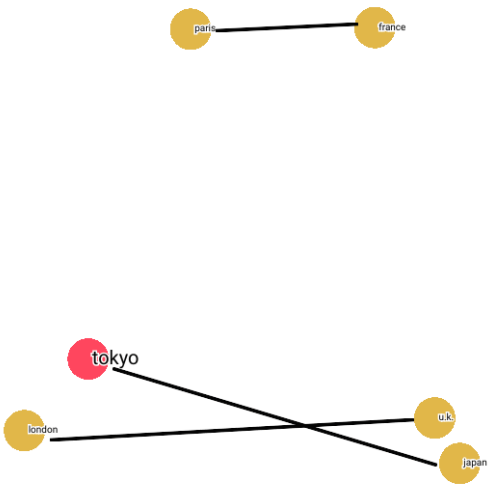

Figure 7: 2D pca projection of 100-d skipgram vectors for the countries and their capitals

```
Words closest to billion
mln dlrs rose year total pct fell compared stg rise sales net share february quarter
Words closest to oil
prices production last said also crude gas price would u.s. two exports new government may
Words closest to shares
common stock share company lt corp inc offer stake acquisition sale outstanding dlrs cash ltd
```

Figure 8: Outputs for test words billion, oil, shares

```
Top 20 predictions:
sees increase higher expects annual losses inc half end first said year rd pct corp ausinont earnings feb jan expected
see:sees::say:says
Top 20 predictions:
say sees products completed inc rcpts said previously limited explanation recession rd expects greek resources computer
Total questions: 52
Accuracy for the analogy task: 0.21
```

Figure 9: Some outputs for the analogy task

References

Felix Hill, Roi Reichart, and Anna Korhonen. 2014. Simlex-999: Evaluating semantic models with (genuine) similarity estimation.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality.