Deep Learning
CS7015
Programming Assignment 2
Vectorial Representation of Words

Shubhangi Ghosh - EE15B129
Monisha J - CS15B053

March 2018

1 Models

The models used for vectorial representation of words with code sources cited:

1. Continuous Bag of Words [Source]
   (a) Negative Sampling
   (b) Hierarchical Softmax
2. Skipgram [Source]
   (a) Negative Sampling
   (b) Hierarchical Softmax
3. GloVe [Source]

2 Data Sources

The data sources used for collecting data in the Tamil language are:

1. EnTam - English-Tamil Parallel Corpus here
2. Tamil Wikepedia Dump here
3. Tamil NCERT Textbooks here.

4. Tamil News websites like Dinakaran, Dinamani, Virakesari, Theekkathir and Uthayan.

A tool called WebBootCat by SketchEngine was used to crawl tamil text from News websites to simplify the job.

3 Data Preprocessing

The following steps were taken to preprocess the data before feeding them to the models :

1. All tokens of length less than three were removed from the data, which removed a few stopwords and all kinds of punctuation.

2. English words and numerical tokens were removed using a python library *langdetect*.

3. The wikipedia dump was extracted using the wikipedia dump extractor found here. Further, the articles were extracted out and broken down into sentences.

4. The multiple data files were combined into a single data file containing all the data. While concatenating two different documents articles, a sentence consisting of five dummy words was appended so as to mark the contextual distance between articles that are not related.

4 Evaluation Measures

4.1 Semantic Relatedness

To measure the performance of the models with respect to scoring semantic relatedness between words, a test set of 30 pairs of words was crafted. Five persons who are well-versed in Tamil were asked to manually rate the word pairs on their relatedness, on a scale of 0 to 10, 0 being totally unrelated and 10 being very closely related. The average of the five ratings was computed and fixed to be the human rating.
Ratings were calculated for each word pair using cosine similarity scaled to the range of $[0, 10]$ for each model. Correlation between the models' ratings and the human ratings was computed using Spearman's correlation coefficient and the models were compared on this value.

4.2 Synonym Generation

For a given input word, a list of n synonyms is generated, using cosine similarity as the word similarity measure. This is performed by computing the cosine similarity of the input word with all other words in the vocabulary, and the top n words that have the highest similarity are returned.

4.3 Analogy Computation

Semantic analogy is tested by performing semantically interpretable algebraic operations using the word vector representations. For example, $v_{grandson} + v_{sister} - v_{brother}$ must return a vector close to $v_{granddaughter}$. This is implemented by first calculating the results of the vector operations to get the solution vector. Cosine similarities are computed between this vector and the vectors of all the words in the vocabulary and the closest word(s) is returned. In our implementation, we return the 10 closest words.

5 Continuous Bag of Words

Hyperparameters for CBOW were tuned and the following results were obtained:

5.1 Embedding Size:

Embedding Size	Semantic Relatedness
100	0.53
200	0.50
300	0.49

Observations and Interpretation:
Embedding size 100 does better as for higher sizes latent relationships between words may be lost.

5.2 Starting Learning Rate:

Learning Rate	Semantic Relatedness
0.1	0.46
0.01	0.52
0.001	0.51

Observations and Interpretation:
LR 0.01 does best.

5.3 Negative Sampling Rate:

NS Rate	Semantic Relatedness
5	0.51
10	0.53
20	0.55

Observations and Interpretation:
NS rate improves correlation.

5.4 Hierarchical Softmax:

HS	Semantic Relatedness
-	0.53

6 Skipgram

Hyperparameters for Skipgram were tuned and the following results were obtained:

6.1 Embedding Size:

Embedding Size	Semantic Relatedness
100	0.54
200	0.51
300	0.49

Observations and Interpretation:
Embedding size 100 does better as for higher sizes latent relationships between words may be lost.

6.2 Starting Learning Rate:

Learning Rate	Semantic Relatedness
0.1	0.47
0.01	0.53
0.001	0.52

Observations and Interpretation:
LR 0.01 does best.

6.3 Negative Sampling Rate:

NS Rate	Semantic Relatedness
5	0.52
10	0.54
20	0.55

Observations and Interpretation:
NS rate improves correlation.

6.4 Hierarchical Softmax:

HS	Semantic Relatedness
-	0.54

7 GloVe

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. Hyperparameters for GloVe were tuned and the following results were obtained:

7.1 Embedding Size:

Embedding Size	Initial Learning Rate	Semantic Relatedness
50	0.1	0.51
100	0.05	0.47
100	0.1	0.46
200	0.1	0.42

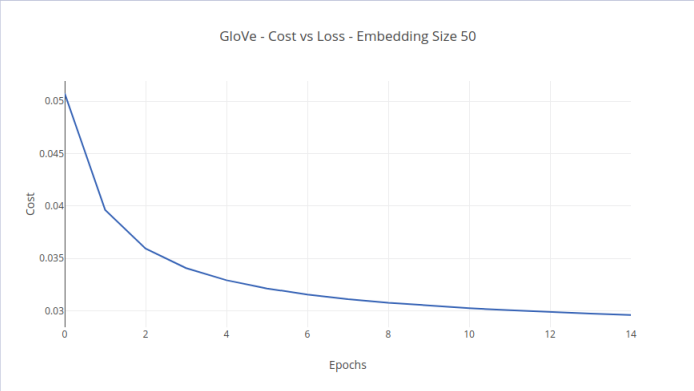

Figure 1: GloVe - Cost vs Loss - Embedding Size 50

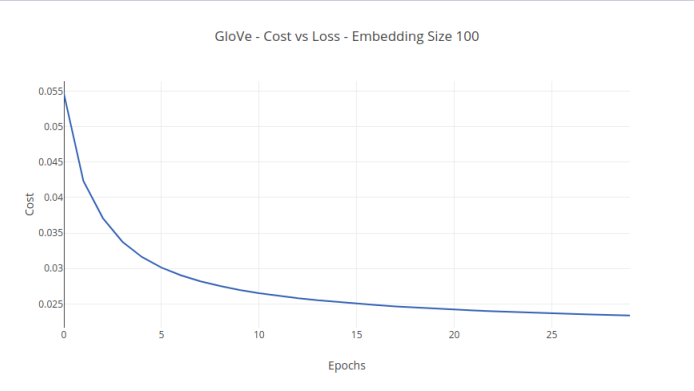

Figure 2: GloVe - Cost vs Loss - Embedding Size 100

Observations and Interpretation:

A plot of loss vs epochs can be seen in Figure 1 (Embedding size 100), Figure 2 (Embedding size 100), and Figure 3 (Embedding size 300).

8 Observations and Conclusions

The best model provided around 0.56 correlation by Spearman's measure.
Model : Skipgram
Hyperparameters :
- Initial Learning Rate : 0.025
- Embedding Size : 100

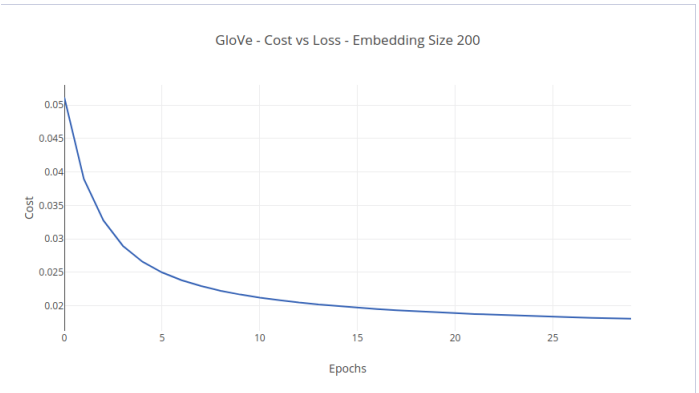

Figure 3: GloVe - Cost vs Loss - Embedding Size 200

- Window : 5
- Negative Sampling : Yes, 20

Some ways to improve the word embeddings could be:

1. Stemming or Lemmatization could be helpful as words in the Tamil language display many inflectional forms.

2. Increasing amount of data would give more meaningful vectorial representation of words. Varying dataset size hasn't been tried due ti time constraints as well as lack of sources for vernacular language data.