Assignment 2

CS 565

Due date: 10th March, 2017

This assignment is about learning vector representation for words. Throughout this assignment we will be using a small processed text corpus available at http://mattmahoney.net/dc/text8.zip. This contain a single file with tokens $w_1, w_2 \ldots, w_{17005208}$ separated by space. Consider only top 10K most frequent words and replace rest other words with a unique word "UNK".

1 Neural Probabilistic Language Model (NPLM)

Learn distributed vector representation using NPLM [BDVJ03]. Take following hyper-parameters in model:

- Word embedding size 50, which means you have to learn 50 dimension vectors for each words
- Context window size 5, which means you have to predict next word based on five past context words.
- The hidden layer size would be 100.
- Learning rate of around 0.01 would be good enough for this experiment.
- Run at-least one epoch over training dataset.

2 Latent Semantic Analysis (LSA)

Consider obtaining distributed word embeddings by applying SVD (singular value decomposition) on word-word co-occurrence matrix obtained on the provided corpus. You may build the co-occurrence matrix based on neighborhood of words (5 words left and right of the present/target word). Obtain 50 dimensional vector embedding for each word.

2.1 Bonus marks

Instead of using SVD implementation of the chosen library, implement SVD and compare the two set of vectors (one obtained from library and another from your own implementation) while evaluating word embedding in the next section. In your report, summarize the steps of SVD implementation.

3 word2vec

Implement the CBOW model with negative sampling [Ron14] and obtain 50 dimensional vector embedding for each word. Use negative sampling size 10, context window size 5, initial learning rate 0.01 and run at-least one epoch over training dataset.

4 Evaluations of Word Embedding

1. Use 100 words of your interest from dictionary and their learned vectors through each one of trained models and visualize it through t-SNE projection[1]. You may use scikit learn[2]

[1] https://lvdmaaten.github.io/tsne/
[2] http://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

package for t-SNE projection. Did you observe any pattern in the projections? Add your observation and images (vectorized images will be nice) of t-SNE projections in your report.

2. Do an extrinsic evaluation of the word vectors obtained via the three methods mentioned above on the Named Entity Recognition task. Implement a window based Neural Network model (window size 5) to predict named entity tag for each word. You can download the named entity dataset from https://github.com/glample/tagger/tree/master/dataset. Consider *eng.train* as training data, *eng.testa* as validation data and *eng.testb* as testing data. Report performance of each method for this task, with and without updating the word vectors while training for NER task.

References

[BDVJ03] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003.

[Ron14] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.