Natural Language Processing Assignment #1
word2vec

March 9, 2019

Due date: 4/8 11:59 PM. These questions require thought, but do not require long answers. Please be as concise as possible. We encourage students to discuss in groups for assignments. You should final submit a package which consists: a report (Latex) and source code with the necessary annotation. You should put your implementation ideas and observation in one report. And the report needs your name and student number. Your basic word2vec and its improvement version should be put into two folds and each part contains two parts: train and test. All related resources can be found at `https://cloud.tsinghua.edu.cn/d/d0086178a1c24ebd93d0/`. We provide a word2vec code based on Tensorflow, Wikipedia corpus and test file WordSim353. If you have any questions, you can discuss on `https://github.com/thunlp/NLP-THU`.

1 Gradients Calculation

Assume you are given a predicted word vector v_c corresponding to the centre word c for skip-gram, and word prediction is made with the softmax function found in word2vec models:

$$y_o = p(o|c) = \frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$$

where w denotes the w-th word and u_w $(w = 1, ..., W)$ are the output word vectors for all words in the vocabulary. Assume cross entropy cost is applied to this prediction and word o is the expected word (the o-th element of the one-hot label vector is one), derive the gradients with respect to v_c.

2 Word2vec Implementation

In this part, you will implement the word2vec models and train your word vectors with deep learning framework Tensorflow. First, you should process a large English corpus, such as Wikipedia. Then, choose cbow or skip-gram architecture to train word2vec with NCE/Negtive Sampling. Finally, you should evaluate word embedding quality on WordSim353 with Spearman's correlation coefficient, embedding performance with different dimensions (100, 200, 300). Note:

Some python package can help you to process corpus such as gensim and nltk. You should utilize `nltk.metrics.spearman` to calculate Spearman's correlation coefficient. The observation, embedding performance and some implementation details will be considered into the score. We provide a more sophisticated version in the resource package.

3 Word2vec Improvement

In this part, you should utilise some technologies to further implement word2vec models. In general, you can consider these directions: incorporating word sense, utilizing word knowledge such as WordNet and HowNet or considering character embedding. Write how you improve your word embedding and evaluate word embedding quality on WordSim353 with Spearman's correlation coefficient. Then you should observe the change of the word embedding in 300 dimension (such as cosine similarity of two embeddings from different models of the same word). Note: The basic improvement can be found in slide 2 section 3 page 100-107 and other technology can be found in slide 2 section 3 page 80-112.