Natural Language Processing Assignment #1
word2vec

February 29, 2020

Due date: 3/29 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 and finish homework independently. Plagiarism is prohibited for all homework. You should final submit a package that consists: a report (Latex) and source code with some necessary annotations. You should put your implementation ideas and observations into your report. And the report needs your name and student number. Your basic word2vec and its improvement version should be put into two folds. 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 in the Wechat group NLP@THU2020 during office hours.

1 Gradients Calculation

Assume you are given a predicted word vector v_c corresponding to the center 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 a word2vec model and train your word embeddings with deep learning frameworks, Tensorflow or Pytorch. First, you should process a large English corpus, such as Wikipedia. Then you can choose CBOW or Skip-gram architecture to train word2vec with Negative Sampling. Finally, you should evaluate word embedding quality on WordSim353 with Spearman's correlation coefficient, the embedding performance with different dimensions (100, 200, 300) should be evaluated. The observation, embedding performance and implementation details will be considered into the score. Therefore, to enrich your experiments and observations, you can visualize the word embedding or involve more evaluations, such as word analogy[1].

Note: Some python packages can help you to process corpus in your homework, such as gensim and nltk. You should utilize `nltk.metrics.spearman` to calculate Spearman's correlation coefficient for evaluation. For more word2vec evaluation datasets, you can refer to the Github project[2].

3 Word2vec Improvement

In this part, you should utilize some technologies to further improve the performance of word2vec model in Problem 2. In general, you can consider these directions: incorporating word sense, utilizing word knowledge 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 differences of the word embedding of 300 dimension (such as cosine similarity of two embeddings from different models of the same word). The idea, observation, and embedding performance will be considered into the score. Therefore, to enrich your experiments and observations, you can visualize and compare the word embedding or involve more evaluations, such as SCWS [1].

Note: The basic improvement inherits the previous work [2] and you can come up with a more effective model than this work. The model is introduced in class and can be found on page 142-149 of the slide. Some other technologies can be found in the section "Applications of Word Embedding" (page 124-153) of the slide. For the word knowledge base, you can choose WordNet or HowNet[3] in your experiments. You can use WorNet through NLTK package following the API indication[4].

References

[1] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.

[2] Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035, 2014.

[1] `https://aclweb.org/aclwiki/Analogy_(State_of_the_art)`

[2] `https://github.com/kudkudak/word-embeddings-benchmarks`

[3] `https://openhownet.thunlp.org`

[4] `https://www.nltk.org/howto/wordnet.html`