CAPSTONE PROJECT PROPOSAL:

SEMANTIC SIMILARITY USING N-GRAMS AND NEURAL N-GRAMS

NGOC (MIKE) TRAN

UDACITY MACHINE LEARNING ENGINEER NANODEGREE PROGRAM

FEB 20TH, 2021

DOMAIN BACKGROUND

Semantic similarity between two words is a metric that measure the similarity between the meanings of the two. Historically, there are many ways to compute semantic similarity. According to Slimani (2013), it can be measured by ontologies, topologies, graph based and statistical similarity. In our context, we are only interested in calculating semantic similarity based on statistical inference. Specifically, we will build a statistical language model and transform all the words to vectors. To compute the semantic similarity, we will compute the cosine distance between two vectors represented the two words. A cosine similarity is range from [0,1] where a 1 is interpreted as identical and 0 is totally different (Sidorov 2014).

Semantic similarity is one of the most important applications of statistical language model. It does not only allow for finding interchangeable words in a same context, but also extend the research of natural language processing to find the similarity between larger mediums like paragraphs or texts. Another application of semantic similarity is plagiarism detector which we already did in the previous Udacity project. A good semantic similarity measurement depends on how good is the statistical language model. To date, there are many types of statistical language model: n-grams, decision tree models, linguistically motivated models, exponential models, and neural network (Rosenfeld 2000).

PROBLEM STATEMENT

A statistical language model is used to model the joint probability distribution of words in a language. The n-grams model has been used widely and achieved many successes. In 2003, Bengio et al. proposed a new model that used neural network as base which is called a neural probabilistic language model. The authors claimed that the model is 24% better than a traditional n-gram. In this project, we will try to replicate the work of Bengio et al. (2003) by implementing both models: the classic n-grams model and the neural n-grams model to validate if later is actually better. We will also compare and contrast the two models on the task of calculating semantic similarity between words in English.

DATASETS AND INPUTS

The datasets we will use is the Brown Corpus (Francis 1979). The Brown corpus consists of 1,200,000 English words. We divided the corpus into a training set of 800,000 words, a validation set of 200,000 and the rest is for testing. According to Bengio et al. (2003), there are 47,578 distinct words. There are also uncommon words (appeared less than 4 times) were merged together to reduce the vocabulary size.

SOLUTION STATEMENT

We will implement the two models based on the set up described in the paper. First, we will download and clean up the Brown corpus. Implementing the traditional n-grams model will be straight forward. In fact, we already did an n-grams implementation for the previous project: Plagiarism Detector. However, we would need to modify that version to fit with the n-grams model used in the paper. The neural n-grams model has 3 layers: an input layer which is a word-embedding layer, a hidden layer with a tanh function, and an output layer with softmax function. First, we take the words, represent them as embedded vectors then concatenate them. Then, we will feed forward through the tanh function and change the dimension to match the vocabulary' size. Lastly, we will use the softmax function to have the probability distribution of words. With that distribution, we will calculate the log perplexity which we try to minimize as the loss function. Finally, the results of the two models are compared to each other.

BENCHMARK MODEL

We have the benchmark model from Belgio et al. (2003). The authors have reported all the perplexity of each model with various versions of different benchmarks and parameters.

EVALUATION METRICS

In natural language processing, the log perplexity is a metric to evaluate statistical language model. We will use it as the evaluate metrics to compare models. Furthermore, we will calculate the cosine distance for 20 pairs of similar words pick randomly from the dictionary.

PROJECT DESIGN

Estimated work flow for the project:

1. Download and examine the corpus
2. Review n-grams implementation from the project Plagiarism Detector
3. Modify the n-grams model to match with the implementation from Bengio et al.
4. Study the paper to understand how the neural n-grams work
5. Implement the neural n-grams model:
    a. Training code
    b. Evaluation code
    c. Hyper parameters tuning
6. Compare the two models based on log-perplexity
7. Test the two models for semantic similarity

REFERENCES

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A Neural Probabilistic Language Model. *Journal of Machine Learning Research 3,* 1137-1155.

Francis, N., & Kucera, H. (1979). *Brown Corpus Manual*. Brown Corpus Manual. http://korpus.uib.no/icame/manuals/BROWN/INDEX.HTM#bc2

Sidorov, G., Gelbukh, A., Gómez-Adorno, H., Pinto, D. (2014). Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model. *Computación y Sistemas*. 18 (3): 491–504. doi:10.13053/CyS-18-3-2043.

Slimani, T. (2013). Description and Evaluation of Semantic Similarity Measures Approaches. *International Journal of Computer Applications*. Vol 80. 25-33. 10.5120/13897-1851.

Rosenfeld, R. (2000). Two decades of statistical language modeling: where do we go from here? *in Proceedings of the IEEE*, vol. 88, no. 8, pp. 1270-1278, Aug. 2000, doi: 10.1109/5.880083.