Vector vs. Lexical Semantics*

Mayank Kumar
University of Windsor
kumar48@uwindsor.ca

1 INTRODUCTION

Semantics is the study of the meaning of words. In Natural Language Modelling (NLM), the semantics of words can be explained using two approaches, namely i) lexical semantics determined manually by linguistics and ii) vector semantics that is based on the idea of distributional semantics. In our work, for simplicity, we argue that lexical semantics are the golden standard about the semantics of the words based on which we evaluate the results of vector semantics methods - TFiDF[1], Word2vec[2] using nDCG[3] as our metric.

2 MOTIVATION

NLM literature has some good data resources curated by linguists for traditional techniques, including semantics data. Recent go-to techniques for word representation are vector methods ranging from simple techniques such as Count Vectorizer and TF-IDF Vectorizer to complex ones such as word2vec. We intend to compare these vectorizer techniques with our Golden Standard - lexical semantics, and quantitatively report the same using nDCG.

3 PROBLEM DEFINITION

Given a golden standard G and a large corpus of text C for the English language, calculate the average Information Retrieval (IR) metric m of top-k similar words retrieved by the vector semantics based on method v. For our work, we have G : SimLex-999[4], C: Browns(Romance, News)[5], v: TFiDF and Word2Vec using cosine similarity, top-k: top-10, m: Average nDCG.

3.1 Example

We develop a golden standard by processing SimLex-999 such that each word **'w'** will have at max ten linguistically defined similar words. Let's look at w - 'accept' from G, we get top-k-G : [resect, acknowledge, believe, deny, forgive ..]. Lets say we trained TFiDF vectorizer on 2 Brown Coprus subcategory - romance and news separately, v1_rom and v1_news. Using cosine similarity, we can then compute top-10-v1. Ex: top-10-news_1_10 :[letter, assistance, Another, Union, band, library,..].

4 EXPERIMENT

4.1 Golden Standard

We select SimLex-999 as our golden truth. For each word w, we order the top-10 similar words to w as the golden list for w. To achieve the same, we simply groupby the data on word1 and aggregated the word2 values. We kept maximum ten words for each word either by truncating the list to top 10 or by expanding using the transitivity rule, i.e., w similar-to a, a similar-to b, then w similar-to b. We expanded using grouped data by indexing words to reach their transitively similar word.

4.2 Vectorizer - TFiDF

We use two different corpora: brown's news(C1) and romance(C2), and trained TFiDF vectorizer. For TFiDf, we used a minimum word occurrence of 5, strip accent to true, and a preprocessor function to remove characters other than words and spaces. We then use cosine similarity to generate top-10 words for all words in our G and report the average nDCG(Fig 1).

4.3 Vectorizer - Word2Vec

We use gensim to train Word2Vec over our corpus C1 and C2 separately. We train multiple models, for 5000 epochs, using all combinations of parameters - context window size 1, 2, 5, 10 and vector size 10, 50, 100, 300. We compare the top-10 similar words generated from word2vec models using cosine similarity and record the average nDCG. We find the best word2vec model as news_1_50 and romance_1_300.

4.4 Results

Figure 1 represents the average nDCG of various vectorizer methods - TFiDF, word2vec(best) over C1- news and C2- romance.

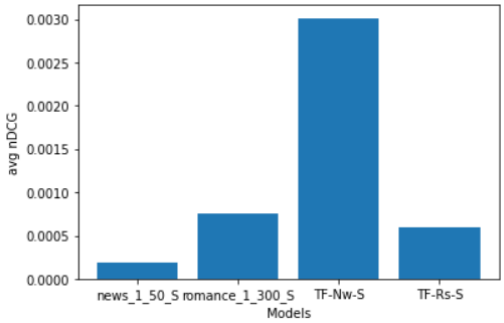

Figure 1: Avg nDCG for various vector models

5 CONCLUSION

For our specified Golden Standard, all the models performed poorly. The best came to be a TFiDF model trained on Brown's news corpus with a marginal average nDCG score of 0.003. The SimLex-999 corpus is well known to measure how well models capture similarity rather than relatedness or association. For ex, the word 'old' and 'new' are similar, with a simlex score of 1.58.

Also, in our setup, word2vec suffers more as most of the sentences have very short context, less than ten words, and the total volume of the training sample was close to 4500 for a context window of 1 and 3759 for a context window of 10, which are very low. The performance of TFiDF helps further understand how even in the worst case scenario, these models can learn co-occurrence from a small volume of data. Also, in terms of train time efficiency, TFiDF is much faster as it is simple matrix multiplication.

*https://github.com/mayankkom-dev/NLU_Assignment3

Conference'17, July 2017, Washington, DC, USA
2022. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

REFERENCES

[1] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html.

[2] https://radimrehurek.com/gensim/models/word2vec.html.

[3] https://en.wikipedia.org/wiki/Discounted_cumulative_gain.

[4] Felix Hill, Roi Reichart, and Anna Korhonen. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4): 665–695, December 2015. doi: 10.1162/COLI_a_00237. URL https://aclanthology.org/J15-4004.

[5] W. N. Francis and H. Kucera. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979. URL http://icame.uib.no/brown/bcm.html.