Exploring Performance of Target and Context Word Embeddings on Extractive Text Summarization

Nate Carlson

3/21/22

Abstract

Extractive text summarization refers to the task of selecting a subset of K sentences that best convey the overall meaning of a given document. Both static word embeddings (word2vec, fastText, GloVe) and contextual word embeddings (BERT) have been used to accomplish this task. Static word embedding models learn 2 sets of embedding vectors, typically termed "target" and "context" embeddings. This work explores structural differences in target and context space, and analyzes performance differences between the embedding sets on two extractive text summarization approaches TextRank and k-means clustering. This project also includes a comparison between static embeddings and contextual embeddings for extractive summarization.

1 Introduction

Modern approaches to text summarization can be divided into two classes: abstractive and extractive. Abstractive summarization more closely mirrors the approach that a human would take to solve this task. It generates a unique paraphrase, possibly using words that are not in the document vocabulary, that conveys the main ideas found in the document. This approach is very desirable, since it mimics a a humans thought process and has been shown to produce impressive results. However, it suffers from several drawbacks including a time and energy intensive training/fine tuning process and potentially volatile outputs. Due to these contraints, this project explores extractive text summarization since it provides a more straight forward and tractable solution to the summarization problem.

Extractive text summarization can be defined as follows: given a document $D = s_1, \ldots, s_N$, where the s_i are the sentences that make up the document, extract the K most relevant sentences in the document. In this project use two popular approaches to extractive summarization, the TextRank algorithm and k-means clustering (described in 3). I use word2vec target and context embeddings in each of these frameworks and compare

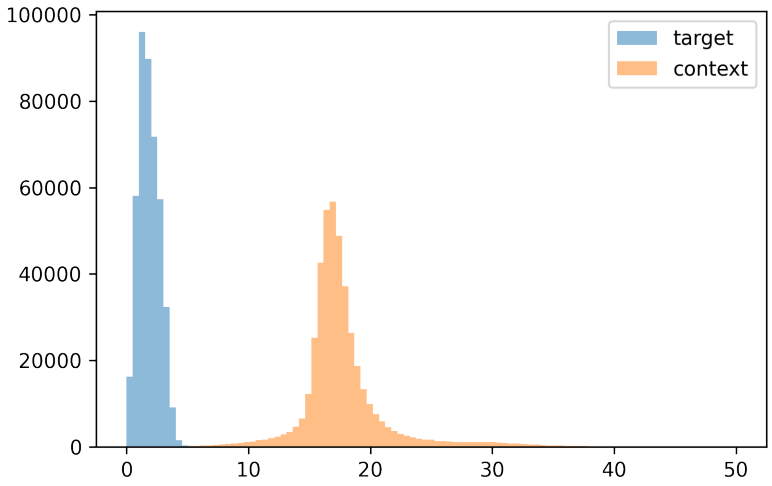

Figure 1: Distribution of word norms for word2vec target and context embeddings. Note the that the distribution of context vector norms has greater variance and much larger mean compared to that of target vectors.

their performance. For the k-means approach I also use BERT sentence embeddings to provide a comparison between static and contextual embeddings for this task. In section 2 I provide a hypothesis as to why target and context vectors might perform differently on extractive summarization.

2 Word2vec Analysis

The **word2vec skip-gram** [3] model learns word representations via solving a neural context prediction problem. The word2vec architecture is a simple MLP with a single hidden layer and no activation function. Given a word the model seeks to maximize the dot product between the in weights associated with the word in question and the out weights of words that occur frequently in the same context window. We refer the to input words as the *target* and its neighbors as *context*. Typically the target embeddings (or in weights) W_{in} are extracted and used as word representations for downstream tasks, while the context embeddings (out weights) W_{out} are typically discarded. However, we will show in this section that the structures of target and context space are inherently different, and that these structural differences make context embeddings particularly well suited for summarization tasks.

The training of the **word2vec skip-gram** can be summarized by two actions [4]

- **Action 1**: Maximize the dot product between target embeddings for input words and context embeddings for their neighbors
- **Action 2**: Negative sampling, or minimize the dot product between

target embeddings for input words and randomly sampled context embeddings

The context vectors in **Action 2** are sampled from the distribution of square rooted word frequencies, while target and context vectors in **Action 1** are both sampled from the distribution of true word frequencies in the corpus. This means that context vectors for less common words will appear more frequently in **Action 2** than in **Action 1**, conversly context vectors for common words will appear more frequently in Action 1 than in **Action 2**. Since context embeddings, unlike target embeddings, do not appear with the same frequency in the two training tasks, there is more variability in their lengths. Context vectors likely become large to product high positive dot products with neighbors and low negative dot products with non-neighbors. We can observe this phenomena by plotting the distributions of word norms for target and context embeddings seperately, see Figure 1.

Interestingly, the vector norms for stop words in context space are on average far shorter than the norms for non stop words. I computed the average stop word norm for context embeddings and compared it against the average context embedding word norm. The average word norm for context embeddings is 17.74 while the average stop word norm is only 6.22. This phenomena does not occur in targe space where the average stop word norm is 2.86 and the average word norm is 1.85. In context space stop word vectors are on average 3 times smaller than vectors for regular words. This suggests that context embeddings might be uniquely well suited for tasks that involve summing word embeddings together, i.e. extractive text summarization. Since stop word norms are much smaller than regular word norms the sentence embeddings created in these extractive summarization tasks will not be diluted by stop words that contain no meaninful information.

3 Approach

3.1 TextRank

The TextRank algorithm [2] constructs a matrix of similarity scores between sentences in a document and runs the PageRank algorithm to obtain a ranking of the sentences, the top K sentences are then concatenated and returned as the summary. There are several choices one must make when implementing TextRank including how to represent a sentence and how to compute similarity between two sentences. The original TextRank paper represents a sentence as a bag of words, that is a vector whose length is equal to the size of the vocabulary that has a n in the ith index if the ith word appears n times and a 0 otherwise. Given two sentences in a document $s_i = w_1^i, \dots w_{n_i}^i$ and $s_j = w_1^j, \dots w_{n_j}^j$ the TextRank paper algorithm uses the following similarity measure.

$$\mathit{Similarity}(s_i, s_j) = \frac{|\{w_k \mid w_k \in s_i \& w_k \in s_j\}|}{\log |s_i| + \log |s_j|} \tag{1}$$

The obvious draw back to the above mentioned approach is that a bag of words vector does not capture the semantic meaning of a sentence. Researchers have also used static word embeddings like word2vec to represent sentences in TextRank [6] and computed cosine distance between sentences to construct the similarity matrix.

3.2 K-Means

Another approach to extractive text summarization is to use k-means clustering [5]. This method is both simple and intuitive, the algorithm is as follows. Create an embedding for each sentence s_i in a document $D = s_1, \ldots, s_N$ using some embedding algorithm, in our case word2vec or BERT. Once embeddings $e_1, \ldots, e_N$ are obtained for each sentence in the document run k-means clustering on the sentences for a set number of clusters K. After clustering the sentences for each cluster centroid c_i find the closest sentence, that is

$$x_i = \arg\min_{e \in C_i} ||c_i - e|| \tag{2}$$

where C_i is the set of all embeddings assigned to cluster i. Then simply concatenated the sentences associated with $x_1, \ldots x_K$ to construct the summary.

It makes sense that this method could be effective if the sentence embeddings can accurately capture the semantic information in the articles. If sentences are able to be partitioned into distinct semantic clusters that each represent a different subtopic in the article, then taking the most representative sentence from each cluster would provide a reasonably good overall summary of the article.

4 Data

The BBC News Summary dataset [1] contains 2225 articles from 5 different news categories including business, entertainment, politics, sports, and tech. Each article is paired with a brief summary of the article contents.

5 Experimental Setup

For each article I compute a 3 sentence summary. I then compute the Rouge-1, Rouge-2, and Rouge-L scores and average them on a per-topic basis. Due the the results of the analysis performed in section 2 I hypothesize that word2vec context will outperform word2vec target in both approaches. For each method I also used pretrained BERT embeddings to compute sentence similarity to perform a comparison between static and contextual embeddings for extractive summarization.

I used word2vec embeddings that were trained on Wikipedia text here at BYU. For BERT I use the all-MiniLM-L6-v2 model from the Sentence Transformer library.

5.1 TextRank

I evaluate the TextRank algorithm using word2vec target embeddings and context embeddings to compute sentence similarity.

This project does not compare the original implementation of TextRank given in equation (1) with the word3vec implementation. Instead, I focus on comparing performance of TextRank implemented using word2vec target and context vectors.

5.2 K-Means

For the k-means clustering experiments I set the number of clusters to be 3 to obtain a 3 sentence summary. I used the scikit-learn implementation of k-means clustering with the default parameters. For each article I cluster I also report the k-means inertia and analyze the correlation between inertia and Rouge scores.

6 Results/Analysis

Business	Rouge-1	Rouge-2	Rouge-L
`w2v target`	0.543	0.429	0.533
`w2v context`	**0.567**	**0.458**	**0.559**

Entertainment	Rouge-1	Rouge-2	Rouge-L
`w2v target`	0.542	0.435	0.533
`w2v context`	**0.566**	**0.458**	**0.557**

Politics	Rouge-1	Rouge-2	Rouge-L
`w2v target`	**0.537**	**0.427**	**0.526**
`w2v context`	0.528	0.411	0.518

Sports	Rouge-1	Rouge-2	Rouge-L
`w2v target`	0.520	0.412	0.510
`w2v context`	**0.529**	**0.418**	**0.519**

Tech	Rouge-1	Rouge-2	Rouge-L
`w2v target`	0.492	0.369	0.481
`w2v context`	**0.499**	**0.371**	**0.487**

Table 1: Results for TextRank extractive summarization on the BBC News dataset. The mean Rouge-1, Rouge-2, and Rouge-L scores are presented for each topical category. The best scores for each column are bolded

6.1 TextRank

The results of the TextRank experiment are presented in table 1. Overall we observe that context embeddings perform slightly better, beating target embeddings on 4 of the 5 topic categories. However, the difference in scores

between target and context embeddings seems minimal. To further investigate the differences in performance between target amd context vectors I performed a T-test on the means of their scores. I performed the test across all topics for Rouge-1, Rouge-2, and Rouge-L scores. The p-value for Rouge-1 was 0.134, Rouge-2 was 0.548, and Rouge-L was 0.159. None of the p-values provided enough evidence to reject the null hypothesis that the means of the scores are identical. The size of the test set was rather small at just 2255 articles. Perhaps further investigation on a larger test set could uncover any statistically significant differences in scores if they exist.

Business	Rouge-1	Rouge-2	Rouge-L	K-Means Inertia
`BERT`	0.471	0.342	0.460	7.62
`w2v target`	0.484	**0.363***	**0.474**	1831.59
`w2v context`	**0.485**	**0.363***	0.457	6635.40

Entertainment	Rouge-1	Rouge-2	Rouge-L	K-Means Inertia
`BERT`	**0.487**	0.370	**0.478**	8.65
`w2v target`	0.464	**0.384**	0.465	1972.15
`w2v context`	0.460	0.375	0.460	7539.83

Politics	Rouge-1	Rouge-2	Rouge-L	K-Means Inertia
`BERT`	0.435	0.305	0.424	11.47
`w2v target`	0.438	0.312	0.426	2949.78
`w2v context`	**0.439**	**0.313**	**0.429**	10967.35

Sports	Rouge-1	Rouge-2	Rouge-L	K-Means Inertia
`BERT`	0.450	0.359	0.450	8.89
`w2v target`	**0.455***	**0.369**	**0.455***	1928.95
`w2v context`	**0.455***	0.367	**0.455***	7698.84

Tech	Rouge-1	Rouge-2	Rouge-L	K-Means Inertia
`BERT`	**0.422**	**0.287**	**0.411**	13.03
`w2v target`	0.407	0.271	0.395	3113.46
`w2v context`	0.397	0.259	0.384	11204.29

Table 2: Results for k-means clustering extractive summarization on the BBC News dataset. The mean Rouge-1, Rouge-2, and Rouge-L scores are presented along with mean k-means Inertia for each of the 5 different topical categories. Best scores for each metric are bolded, note that * denotes a tie between two models.

6.2 K-Means

The results of the k-means experiments are displayed in table 2. Perhaps the most significant observation from the experiments is the competiveness of word2vec with BERT. The BERT based approach only beat out word2vec on the entertainment and tech sections, with either word2vec target or context beating out BERT on the remaining categories. This is significant since constructing sentence embeddings using pretrained word2vec is computation-

ally cheaper than with BERT. If word2vec provides competitive performance on this task with less computational cost, it may be a better choice despite the fact that contextual language models like BERT generally exhibit better performance on natural language tasks.

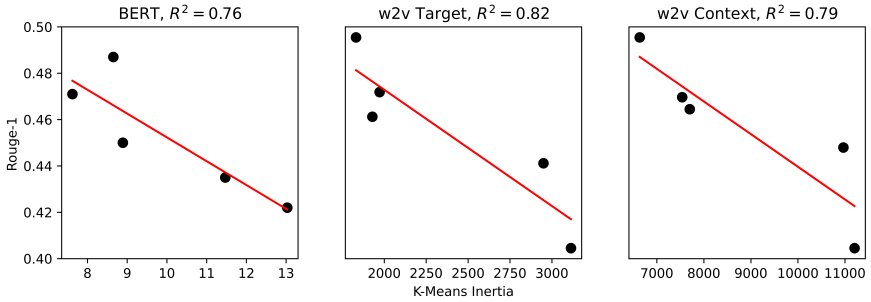

Figure 2: K-Means inertia plotted against Rouge-1 scores across all 5 topic categories for BERT, w2v target, and w2v context. R^2 values are included in the title.

Another observation from the tables is the strong negative correlation between k-means inertia and Rouge scores [1]. Inertia is the sum of squares distance from points in a cluster to the cluster centroid. Inertia can be thought of as a measure of how well the data form well defined clusters. It is intuitive then that lower inertia values would result in better summaries since more well defined sematic clusters . This can be seen in figure 2 where I have plotted the inertia values against Rouge-1 scores for BERT, word2vec target, and word2vec context along with a line of best fit interpolating the data points, a clear strong negative correlation appears in all three plots.

Unfortunately contrary to our expectations there was no clear difference in performance between word2vec target and context embeddings. Perhaps summing vectors to create sentence embeddings diluted the individual meanings for the word embeddings too much making the structural differences between target and context space relatively unimportant.

7 Conclusion

In this project I compared the performance of word2vec target and context embeddings on two extractive text summarization approaches, TextRank and k-means clustering. Contrary to my hypothesis there was not a statistically significant difference between target and context embeddings for either of the approaches. Most notably, static embeddings outperformed contextual BERT embeddings on the k-means method in 3 of the 5 topical categories suggesting that static word embeddings might be the better choice if one is looking for fast off the shelf performance with minimal fine tuning.

[1]Note that because the structure of the embedding space for all three algorithms is inherently different it is not meaningful to compare inertia values across algorithms. However, comparing inertia for the same algorithm on different test sets can yield meaningful insights.

References

[1] Derek Greene and Pádraig Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proc. 23rd International Conference on Machine learning (ICML'06)*, pages 377–384. ACM Press, 2006.

[2] Rada Mihalcea and Paul Tarau. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[4] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013.

[5] Derek Miller. Leveraging BERT for extractive text summarization on lectures. *CoRR*, abs/1906.04165, 2019.

[6] Xiaolei Zuo, Silan Zhang, and Jingbo Xia. The enhancement of TextRank algorithm by using word2vec and its application on topic extraction. *Journal of Physics: Conference Series*, 887:012028, aug 2017.