**MSiA 414**

**Text Analytics**

**Homework 2**

**Creating Word Embeddings with Gensim**

A few words that seemed to capture the topic of the text were chosen from the news group corpora. Below is the list of 11 words that was used to test the word embeddings from gensim:

secular, religion, society, atheism, agnosticism, email, password, software, computer, good and bad.

The words good and bad were added as a test, since they are quite abstract and opposites of each other.

With the Skip-gram model ‘secular’ had agnosticism as the highest cosine similarity of 0.97. This indicates a good word embedding since secular and agnosticism are similar in the sense that they are both opposite to religious beliefs. ‘password’ was the second with 0.95, which does not make very much sense. ‘atheism’ and ‘religion’ were ranked 5th and 6th place with 0.867 and 0.85 cosine similarity, respectively. I would expect ‘atheism’ to be in the top rank, since the it is similar to secular. I would expect religion to have a negative cosine similarity since it is opposite to secular. For ‘good’, ‘bad’ had the highest cosine similarity of 0.97. ‘bad’ is the direct opposite of bad and should have a negative cosine similarity or at least a very low embedding. The word embeddings never seem to have a negative cosine similarity, which does not reflect a very good word embedding. The most similar word to ‘religion’ in terms of Euclidean similarity was ‘difference’. Perhaps, this is due to the connotation that ‘religion’ makes a ‘difference’ in peoples’ life. ‘morality’ came in as the third most similar, which makes sense. ‘evidence’ came in as 9th most similar. It is hard to find reasoning to support why ‘evidence’ should be similar to religion. It is important to note that the most\_similar function in the genism library uses Euclidean similarity, and this similarity is not reliable with a high number of features (as it increases with higher features). The modelling was set to 50 features (*size* option in Word2Vec).

With the features decreased to 5, the performance decreased significantly. ‘secular’ had ‘password’ as the most similar, when viewing cosine similarity. ‘atheism’ and ‘religion’ had about the same cosine similarity of 0.86 and 0.85. The most similar in terms of Euclidean distance was ‘gifs’, second most similar was ‘crank’. It is hard to find reasoning as to why ‘gifs’ and ‘crank’ should be related to ‘secular’. This indicates a decrease in the word vector accuracy as only 5 features were modelled.

The model was next changed to use Continuous Bag of Words (CBOW) with 50 features. from the list of 11 words used , ‘computer’, ‘society’, ‘software’, ‘password’, and ‘email’ had the highest cosine similarity to ‘secular’ (with 0.99, 0.99, 0.98, 0.98, 0.98 similarity respectively). Skip-gram seems to output better word embeddings since the cosine similarity rankings made more semantic sense. However, ‘atheism’ did have a higher similarity to ‘secular’ (0.98). For Euclidean similarity, ‘secular’ had ‘armenian’ and ‘war’ in first and second place. This makes sense as an internet search shows there were Armenain-Turk wars that occurred during the 1940s due to secularist ideas.

Overall, from my usage, Skip-gram tends to perform better and more features helps improve accuracy.

**Literature Reading – Method Comparison**

|  |  |  |
| --- | --- | --- |
|  | Word2Vec | BERT |
| Citations | 15320 | 1851 |
|  |  |  |

Word2vec can train on 100billion words a day. This work trained on 30 billion words. Previous works used 2 to 3 orders magnitudes less (so divide by 10^2 or 10^3) – Wrote in 2013

BERT: first submission 11 Oct. 2018

Bidirectional Encoder Representations from Transformers:

reads from left to right and right to left.

Built from transformer.

ELMo extracts context-sensitive features from left to right and right to left language models and concats them.

Figure two shows how tokens are split

In order to train a deep bidirectional representation,

we simply mask some percentage of the input

tokens at random, and then predict those masked

tokens.

With bert there is pretraining and then finetuning for a specific task such as word prediction or sentence prediction.

learning model details, summary of word context approaches, corpus size requirements, computational requirements, ease of installation/use of source code, date of publication and number of google scholar citations :),