NLP Word Similarity Assignment

CS15B058 B Arjun, EE15B129 Shubhangi Ghosh

20^{th} March 2018

1 Introduction

In this assignment we aim to implement and compare various language models to estimate word similarity. This is a crucial task in NLP as it plays an important role in Word Sense Disambiguation, IR, Text Summarization etc.

We plan to critically analyse 4 major models used in NLP for word similarity estimation:

1. WordNet
2. LSA
3. Word2Vec
4. ESA

By finding the weaknesses and drawbacks of these models, in this assignment we propose some improvements and hybrid models, which may perform better than these standard models.

2 Methodology

For the corpus-based methods, ie. Word2Vec and LSA, we used the Brown Corpus from the NLTK package, as it had about 1 million words, the highest among the suggested corpora.Stop words were retained for both LSA and Word2Vec, as stop words were found to reduce the performance in these models. We also used Porter stemming to ensure multiple forms of the same word are given the same representation.

2.1 WordNet

Path based Similarity measures:
Two Path-based similarity measures were tried:

1. **Leacock-Chodorow Similarity:**
   By Leacock-Chodorow Similarity, synsets with the shortest path between them have the highest similarity.
   $lch(syn_1, syn_2) = -log\frac{len(syn_1, syn2)}{2L}$, where L is the complete depth of the taxonomy.

2. **Wu-Palmer Similarity:**
   By Wu-Palmer Similarity, synsets lower in the taxonomy have higher similarity with the same path length, compared to synsets higher above in the taxonomy.
   $wup(syn_1, syn_2) = 1 - \frac{N_1+N_2)}{(N_1+N_3)+(N_2+N_3)}$

Information theoretic approaches:
The basic idea of informaton theoretic approaches is nodes high in the hierarchy have smaller information content. One information theoretic measure was tried:

1. **Jiang-Conrath Similarity:**
   The information shared by two synsets can be represented by their Least Common Subsumer(ancestor). Similarity is measured as proportional to the amount of information the two synsets share. Jiang-Conrath Similarity is one such information theoretic measure, which assumes all words are nouns and ignores their part-of-speech.

   $$jcn(syn_1, syn_2) = 2log(p(lcs(syn_1, syn_2))) - (log(p(syn_1)) + log(p(syn_2)))$$

2. Semcor-ic was used to evaluate information content of synsets.

The following procedure was followed:

1. Wordnet synsets for each word were found.
2. The max similarity between two synsets each corresponding to one of the words, was considered as the similarity score between the words.

2.2 Latent Semantic Analysis

Preprocessiong:
Words from the Brown Corpus provided by NLTK were:

1. Tokenised
2. Checked for removing non-words using the UK English Dictionary
3. Stemmed using Porter Stemmer and reconcatenated.

Term Weighting Schemes:
LSA was implemented using the Numpy library. Multiple term weighting schemes were tested, both conventional ones like Term Frequency and TF-IDF, and non-conventional schemes involving powers of TF. The following schemes were used:

1. Term Frequency
2. TF-IDF
3. Normalized TF-IDF
4. (Term Frequency)^2
5. (TF)$^2\times$ IDF
6. (TF)$^3\times$ IDF

Hyperparameter Tuning:
Since the Brown Corpus has 500 documents, for each term weighting scheme, we tuned the Number of Concepts hyper parameter by trying all multiples of 5 from 1 to 500.

2.3 Word2Vec

The methodology for Word2Vec implementation is as follows
Preprocessiong:
Words from the Brown Corpus provided by NLTK were:

1. Tokenised
2. Checked for removing non-words using the UK English Dictionary
3. Stemmed using Porter Stemmer and reconcatenated.

Training and Evaluation:

1. This filtered corpus was fed to the Word2Vec model provided by Gensim and word representations were trained.
2. Hyperparameters were tuned to obtain the best word representations. The word representations were evaluated by maximising the correlation with the word similarity score from Wordsim-353 dataset. Cosine similarity is used to compute similarity between word vectors.
3. The model choice and hyperparameters tuned were:
   (a) Skipgram/CBOW(Continuous Bag of Words)
   (b) Hierarchical Softmax/ Negative Sampling
   (c) Context Window
   (d) Embedding Size

(e) Minimum count of words (if certain words occur with lesser frequency than this count, they are discarded)

4. The final model specification and hyperparameter choice:
   (a) Model : Skipgram with Hierarchical Softmax
   (b) Initial Learning Rate: 0.025
   (c) No. of Epochs: 5
   (d) Optimiser: Stochastic Gradient Descent
   (e) Context Window: 5
   (f) Embedding Size: 2000
   (g) Minimum count of words: 2
5. Hyperparameters were tuned individually while fixing all other hyperparameters and choice with best Wordsim-353 correlation was selected. Grid Search wasn't implemented due to time and computation efficiency constraints.
6. During evaluation, if one of the words in Wordsim-353 pair is not found in the vocabulary, we have given the similarity score for the words to be 0.5, as the cosine similarities are in the range of 0-1, and 0.5 is the expected similarity score.
7. Further, Google's pre-trained word vectors were used to evaluate similarity between words and correlation with Wordsim-353 similarity scores, which gave a better correlation than our word representations.
8. Further discussion on rationale behind choosing hyperparameters, why Google's pre-trained word vectors perform better and comparison with LSA can be found in Results and Analysis section.

2.4 Explicit Semantic Analysis

Implementation:
ESA was implemented using the Descartes Library for Java. The inverted index file was created using a Wikipedia dump file dated 3^{rd} March 2018.

Hyperparameter Tuning:
For each word we computed the k best concepts, and in order to compute the cosine similarity between any two words, we took the sum of products of the weights of the common concepts, and divided by the product of the two norms.

This had to be done as the Descartes library provides the k highest ranking concepts for each word, which may vary from word to word. We tried different values of $k \in \{100, 500, 1000, 2000, 3000, 5000\}$ while tuning the model.

3 Results and Analysis:

For each of the models, we computed the Spearman Correlation Coefficient with the WordSim353 Scores. Spearman correlation is used as opposed to Pearson Correlation as it is more effective is evaluating ordinal relationships. In some of the models, certain words which were present in WordSim353 did not have a representation, due to limited corpus size. In these cases, we ignored such pairs of words to ensure the Correlation Coefficient is still a valid representation of the model performance.

3.1 WordNet

Path-based measures:
Among path-based measures, we see that Wu-Palmer performs better than Leacock-Chodorow Similarity, as it also takes into account that when we go lower in the taxonomy, word similarity increases for same path length.

Path-based Measure	Spearman Correlation Coefficient
Leacock-Chodorow	0.30
Wu-Palmer	0.33

Information-based measures:
In our case, Jiang Conrath similarity could have performed worse than path-based measures as Part of Speech is ignored and all words are assumed to be nouns.

Path-based Measure	Spearman Correlation Coefficient
Jiang-Conrath	0.28

3.2 Latent Semantic Analysis

We observed that some non-conventional weighting schemes performed better than the conventional weighting schemes. A summary of the best correlation scores of each weighting scheme along with the corresponding number of concepts has been tabulated below.

Term Weighting Scheme	Number of Concepts	Spearman Correlation Coefficient
Term Frequency	90	0.289
TF-IDF	165	0.281
Normalized TF-IDF	140	0.283
$(TF)^2$	105	0.315
$(TF)^2 \times IDF$	75	0.331
$(TF)^3 \times IDF$	65	0.343

3.2.1 Term Weighting Scheme Selection

It was observed that directly using the Term Frequency performed better than using TF-IDF. This implied that the IDF scores were not very reliable, which

can be explained as the number of documents in Brown Corpus are very small. However, Normalized TF-IDF performed better than TF-IDF, which further showed that the direct document term counts weren't completely reliable so a scaling factor which varied between documents had to be introduced.

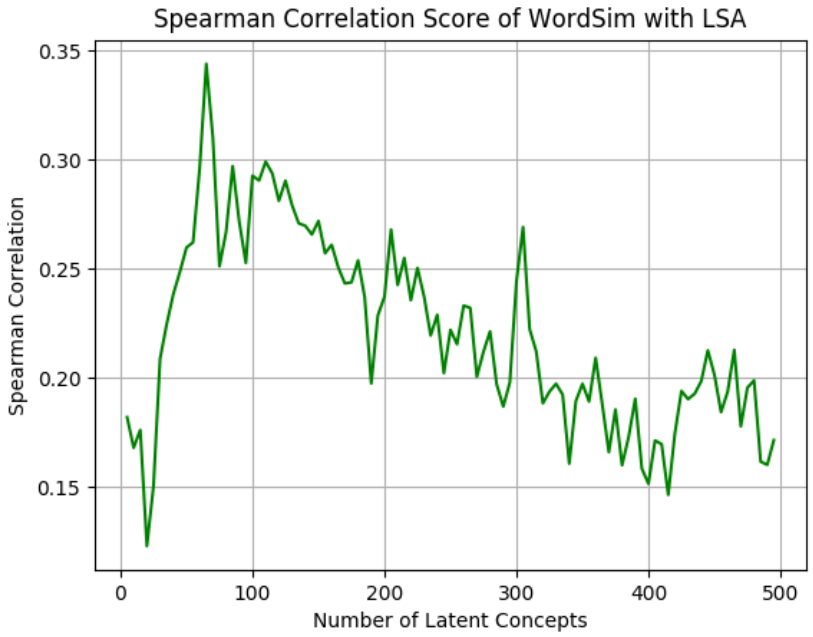
This motivated us to try different weighting schemes which would give a higher weight to the Term Frequencies than the IDF. This could not be accomplished by scaling with a multiple as the representations captured after SVD would not capture this, ie. the representations will be simply scaled. The best way to increase the Term Frequency weight was to take powers of TF.

We tried both square and cube of TF, and also tried to merge it with the idea of TF-IDF. It was observed that $(TF)^3 \times IDF$ performed the best, which implied that a better balance between $(TF)^3$ and IDF was found, as compared to TF and IDF. Also the necessity of IDF can be linked with the fact that stopwords were not filtered, so IDF would reduce the weight of such words, with occurrences in most of the documents.

Another observation we made was that the optimal Concept count was smaller for the schemes using higher weighted Term Frequencies, thus the models are better both in terms of performance and memory efficiency. This can be explained because taking a larger power of TF, it is similar to expanding the Corpus size, and thus a lesser number of concepts are needed to obtain good representations.

3.2.2 Number of Concepts

While tuning the number of concepts, it was observed that after reaching a maximum correlation at a certain number of concepts, increase in number of concepts resulted in a steep drop in correlation score, and for large concept count it varied noisily.



This can be attributed to the fact that with increase in number of documents initially we are adding more reliable concepts to the representations resulting in improving score. After a point, the concepts are noisy and this results in high variance between the scores.

3.2.3 Individual performance in WordSim pairs

Certain word pairs which had high WordSim scores(¿9) were found to have very low cosine similarities, such as (money, currency). However (money, cash) has a high cosine similarity. This is because of the small corpus size, and limited occurrences of currency as compared to cash.

Another case was the word pair (lad, brother), which had a low WordSim score(4.46), had a cosine similarity of 0.47. This is because the Brown Corpus was made in 1961 and a large part of the text was from Novels, where the usage of words like lad, was very common. Since these words were found to co-occur in most of these texts, the vectorial representations obtained from LSA were found to have a high similarity.

The major cause for bad performance of LSA is the corpus size. Due to the limited number of documents there are very few documents involving scientific terms, resulting in bad representations of these terms. Thus pairs like (physics, chemistry) have unreliable similarity estimates.

Polysemous words were also found to perform poorly, such as bank. Words which have high co-occurrence, but differ in meaning were also found to give a high similarity, as LSA does not capture any notion of "word meanings", it relies on co-occurrence. For example (baby, mother) have a high cosine similarity, even though their WordSim score is not very high.

3.3 Word2Vec

Some experiments performed, results and interpretations are illustrated below:

3.3.1 Skipgram vs. Continuous Bag of Words:

Model	Correlation with Wordsim-353
Skipgram	0.41
CBOW	0.16

*Spearman correlation was used.

Thus, we see that Skipgram performs much better CBOW in our case.

This is because Skipgram is better for infrequent words than CBOW, although it is slower than CBOW. Since our corpus size is relatively small, most words occur with smaller frequency. Skipgram learns better representations for rare words than CBOW as:

1. CBOW learns vectorial representation of words by learning to predict the most probable words given its context words. For example, given the context It was a —- evening, CBOW is most likely to predict beautiful or nice, and it is not likely to predict a rare word like delightful. In CBOW, rarer words like 'delightful' are made to compete with more common words. Thus, infrequent words get less attention of the model and are smoothed over by commonly occuring words with similar context.

2. In skipgram, given a word its context words are predicted. Thus, infrequent words are given much more meaningful representations in Skipgram than CBOW.

3. Since our corpus is small, words in general occur with lower frequencies, and hence skipgram helps us get more meaningful representations.

3.3.2 Hierarchical Softmax vs.Negative Sampling

–	Correlation with Wordsim-353
Hierarchical Softmax	0.41
Negative Sampling	0.31

*Spearman correlation was used.

Thus, we see that Hierarchical Softmax performs better than Negative Sampling.

1. Negative sampling requires a lot of incorrect word pairs for every correct word pair. This method maximises the probability of a correct word pair occurring and minimises the probability of an incorrect word pair occurring. But since our corpus size is relatively small, required no. of meaningful incorrect word pairs are difficult to obtain (as rare words may not occur with a lot of words in the given corpus with which they could have potentially occurred otherwise.

2. In hierarchical softmax, leaf nodes representing infrequent words will inherit their ancestors'(more common words) vectorial representations, and thus will be more accurately represented.

3. Since our corpus is small, Hierarchical softmax is preferred.

3.3.3 Context Window:

Context Window	Correlation with Wordsim-353
3	0.34
5	0.41
7	0.36
10	0.36

*Spearman correlation was used.

Window size of 5 gives best correlation.

3.3.4 Embedding Size:

Embedding size	Correlation with Wordsim-353
500	0.34
1000	0.37
2000	0.41
3000	0.38

*Spearman correlation was used.

Embedding size of 2000 gives best correlation.
Lower embedding sizes are not sufficient to represent all the words in the vocabulary effectively, while higher vocabulary sizes may make the word vector too sparse and we may lose out on latent relationships.

3.3.5 Minimum count of words:

min_count	Correlation with Wordsim-353
2	0.41
5	0.38
10	0.35

*Spearman correlation was used.

min_count of 2 gives best correlation.
This is because our corpus is small and words occur with less frequency. Thus, discarding words with a lower frequency than something as high as 10 may not help.

3.4 Anomalies:

Notable Similarity differences:

Word 1	Word 2	Word2vec score(0-1)	Wordsim353 score(0-10)
telephone	communication	0.13	7.5
soap	opera	0.01	7.94
weather	forecast	0.19	8.34
rock	jazz	0.16	7.59

The reasons for the anomalies could be as follows:

1. **telephone & communication:**
   Communication could be of many forms and telephone is a means of communication. Word2vec is not very good at mapping hypernymy similarities. In this corpus, communication doesn't occur in the same context as telephone very much. Maybe, increasing the size of the corpus could also help increase the similarity score as both the words are very infrequent in this corpus, each occurring only about 100 time in a corpus of a million words.

2. **soap & opera:**
   Soap is a polysemous word with its primary context being one where it means washing soap. Word2vec is not good at handling polysemous words. Also, opera is a rare word.

3. **weather & forecast:**
   The word 'forecast' can be used to forecast other activities as well, such as 'forecast economy', 'forecast expreiment results' etc., and in Brown corpus, forecast has occurred in many of the alternative contexts. Also, the word 'forecast' has occurred only 25 times in a million word corpus, and is thus very rare. Thus, similarity score is low.

4. **rock & jazz:**
   'rock' is a polysemous word with its alternative interpretation being more likely. 'jazz' is a rare word.

5. Also, we have noticed that Word2vec gives nearly same similarity scores for **king & queen**(0.49) and **king & cabbage**(0.3). This is because the words 'king' and 'queen' usually occur in the context of proper nouns such as 'England' or 'Victoria'. Thus, they may not essentially occur in the same contexts as proper nouns make the context much more varied. Also, 'queen' occurs very rarely in Brown corpus(54 times).

3.4.1 Comparison with LSA:

1. LSA is a count-based method, i.e. word counts in documents are maintained. SVD is applied on the term document matrix to get lower dimensional word embeddings in terms of the concepts.

2. Word2vec is a predictive model, i.e. it directly predicts vectorial representations of words using a Neural Network.

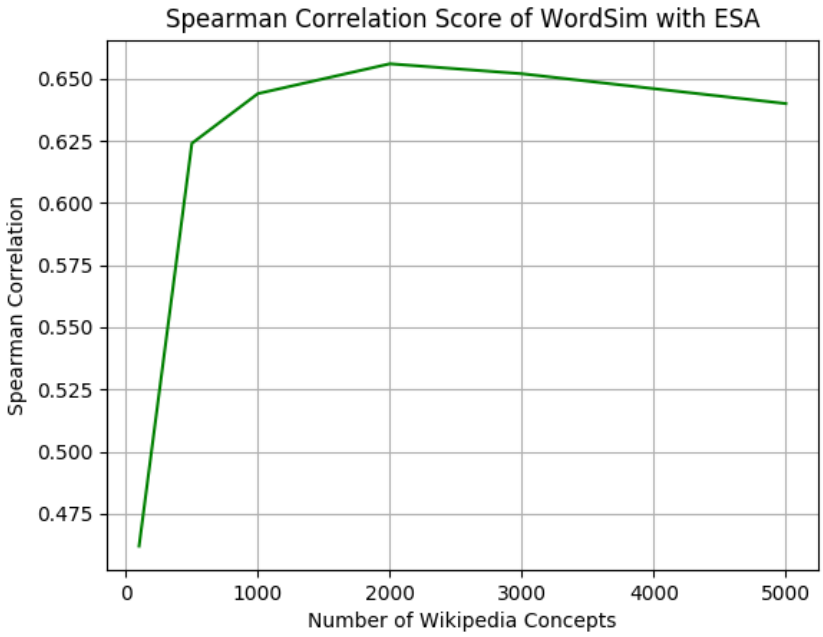
3. We have seen that Word2vec models have performed better than LSA on the same corpus(Brown). Prediction based methods typically outperform count based methods as parameter tuning is much more efficient in these methods. Thus, word2vec is usually the method of choice.

4. However, when corpus size is very small, there less data to train for optimal parameters, and LSA might capture word relationships more effectively.

3.5 Explicit Semantic Analysis

The correlation scores with varying number of ESA Concepts are summarized in the following table:

Number of Concepts	Spearman Correlation Coefficient
100	0.462
500	0.624
1000	0.644
2000	0.656
3000	0.652
5000	0.640



3.5.1 Variation of Number of Concepts

Our observations showed us that with an increase in the number of concepts, ESA need not always improve. It increases till a maxima, and then drops slightly. The following arguments can be put forth to explain these observations:

- As the number of concepts increase, more common concepts can be found between similar, leading to better representations for the word pairs, and better estimation of cosine similarity.
- For very large number of concepts, there may be many "noisy" Wikipedia articles, and the similarity between truly similar words gets reduced, as these noisy concepts may not overlap between the two words.
- ESA uses the assumption that the Wikipedia concepts are orthogonal. But this is definitely not the case, as we can observe most of the top concepts are highly correlated. Thus there may be different concepts between pairs of words, which are actually similar, and thus our similarity computation would be an underestimate in such a scenario.

It was observed that the highest cosine similarity obtained by the optimal ESA(2000 concepts), was 0.31. This low value can be explained because of the large number of concepts, and the limited number of overlaps.

3.5.2 Individual performance in WordSim pairs

Some word pairs, like gender and equality, have a low WordSim score(6.41) but a fairly high cosine similarity(0.213), whereas a cosine similarity of 0.2-0.3 mostly occurred for word pairs with WordSim score larger than 8. This abnormality can be explained because in most of the Wikipedia concepts, gender and equality occur together, so even though they are not similar in meaning, they have a high co-occurrence. Thus the cosine similarity is large. For example, the third most related Wikipedia concept to equality is Gender Equality, which is also the seventh most related Wikipedia concept to gender.

There are however, many word pairs which have a high WordSim score(¿9) but a very low cosine similarity(less than 0.05), such as (dollar, buck), (football, soccer), (journey, voyage), (coast, shore), (magician, wizard). If we inspect the most relevant concepts for the word "buck", we find most of them to be pertaining to famous personalities named/nicknamed "Buck". This is because the word "buck" is an informal term for a dollar, so its occurrences in Wikipedia articles with that meaning would be very low.

Similarly, soccer is a word used only in USA, for football, and thus very few articles overlap between the two words. In the case of (coast,shore) and (magician, wizard), we observe that the word pairs rarely co-occur, and in case of Coast and Shore, the top concepts for coast are mostly pertaining to the Coast Guard, which shares no similarity to the word shore. Also the top concepts for shore pertain to the names of Schools and locations situated in USA. Magician is a more generic word referring to magic tricks as well, so a large number of Wikipedia concepts are related to that, whereas for Wizard most of the concepts are related to fantasy characters.

These observations make it obvious that there are many Wikipedia articles which are "noisy", and the orthogonality assumption made in ESA is false, as many articles have similar content. Polysemous words also get affected adversely, and the most frequent meaning of a word will dominate as majority of the Wikipedia articles pertain to that meaning.

It was also observed that LSA performed well on many word pairs where ESA failed, which motivates our hybrid model mentioned in the Proposals section.

4 Conclusion

The Google Word Vectors were found to perform the best among all the models, giving a Spearman Correlation with WordSim of 0.7, closely followed by ESA with a score of 0.656.

Model	Spearman Correlation Coefficient
Google Word Vectors	0.7
ESA	0.656
Word2Vec using Brown Corpus	0.41
LSA using Brown Corpus	0.34
WordNet	0.33

This dominance of ESA and Google Word Vectors can be explained by the immense data used by these models. ESA uses a 30GB inverted index from Wikipedia and Google Vectors are trained on data containing around 100 billion words, which is much larger than Brown corpus which consists of roughly a million words. If LSA is applied on a large enough dataset, it can obtain scores very close to the Google Vectors.

The main drawback of LSA as compared to Word2Vec, as observed by the better correlation obtained by Word2Vec than LSA, is that LSA does not consider the sequence order of words. It only takes into account the co-occurrence counts. This is why the Skipgram model, which gives different representations based on the sequence of context words, performs better. Word2Vec also distinguishes between a word and its context, resulting in better representations which do not depend purely on the occurrence counts. For example, if our data contains "A dog eats food", and "A cat eats food", LSA would consider cat, dog and eat to be similar, but Word2Vec would only consider cat and dog to be similar.

WordNet had the worst performance as compared to the other models. However WordNet is the best model at capturing polysemous words, as a word may lie in multiple synsets. Given the context of any 2 words, WordNet can give a better word similarity estimate, as it can find the correct sense for the given word.

5 Proposals

5.1 Non-Orthogonal ESA using WordNet

It was observed that many of the Wikipedia concepts were similar but due to the orthogonality assumption of ESA, these similarities were not captured while evaluating the word representations. One solution to this is provided in Non-Orthogonal Explicit Semantic Analysis by N Aggarwal Et al., using various methods for Concept relatedness such as VSM-Text, Hyperlinks, etc. We would like to propose an alternate similarity measure by using the synset information available in WordNet.

Instead of taking a direct overlap of the text between concepts as in VSM-Text, we can use WordNet to find the synsets of all the words in the Wikipedia Concept. This is equivalent to converting the Wikipedia Concept to a "Synset

Vector" instead of a Term Vector. The relatedness between concepts can then be computed by the cosine similarities between these Synset Vectors.

While computing the relatedness between two words, during the computation of cosine similarities we no longer consider the concepts as orthogonal, instead the "between concepts" dot products will also have to be considered. However here we consider the synsets to be orthogonal to each other.

This method maintains the interpretability of ESA, but also captures better similarity in case of synonymous word occurrences in different documents.

To avoid the orthogonal synset assumption, another option to capture the relatedness is to compute the Wordnet similarity measures(such as Wu-Palmer, Leacock-Chodorow, etc) between the synsets, and sum up the similarities multiplied by the concept weights. This could lead to much better representations than ESA, as it captures relatedness between words which were not captured earlier, such as hierarchical similarity, which is not captured by co-occurrences.

In this model, the number of concepts would be a tunable parameter, and we can also tune the weight provided to the Wordnet similarity measure, to ensure ESA plays the major role in deciding similarity and not WordNet.

5.2 Unsupervised Non-Orthogonal ESA using SVD

With a similar aim to the previous proposal, instead of using similarity measures between Wikipedia articles, we can use SVD to learn latent similarities between concepts. SVD will have to be applied on the Term-Concept Matrix, and we can vary the number of latent concepts within these Wikipedia Articles.

After this we can reconstruct the Wikipedia matrix, using these limited concepts thus obtaining modified term concept weights. Certain terms which do not occur in a Wikipedia Article will also have an associated weight. Thus this combination of ESA and LSA, implicitly captures a similarity measure between the concepts due to the nature of SVD. If tuned properly, this model can capture better Concept representations, and similar words which have similar but not identical Wikipedia concepts can be identified, due to the modified term-concept matrix.

This model would perform better than ESA because there are a large number of noisy Wikipedia articles. Using SVD could reduce the weight provided to these noisy articles, resulting in a richer set of top concepts for any word.

5.3 ESA Concept Combination using K-Means Clustering and Gaussian Mixture Models

The major drawback of ESA is the orthogonality assumption that is used. In this method instead of capturing a similarity between concepts, we thought of combining the concepts which are very similar so that no information is lost while considering the top set of concepts, and to ensure more common concepts between words.

After obtaining the Wikipedia concept representations, we can use K-Means Clustering, where K will be a hyper parameter pertaining to the number of Wikipedia Concepts we want to use for our word representations. Each Wikipedia concept can be thought of as a data point in a term-dimensional space. The similarity between any two concepts can be captured by the term overlap, or other similarity measures as described in the Non-Orthogonal ESA Paper(Aggarwal Et al.), such as DiSER.

Starting with K random concepts as clusters we keep combining concepts into clusters till convergence. The representation of each Cluster of Concepts can be represented by the sum total of the individual concept representations belonging to that cluster.

While evaluating the model, each word will be represented as a combination of K Aggregated Concepts, and thus we have the same dimensions for the vector representing each word, unlike ESA. This is also an approximation to creating orthogonal concepts, because after tuning K, we can assume that each cluster captures all the similar articles, and the clusters themselves are dissimilar.

Like ESA, these word representations are still interpretable as each cluster refers to a combination of Wikipedia Concepts, which essentially capture similar/related information.

Such a method should be able to capture much better representations of words, as we are able to grow the article sizes, leading to better co-occurrence approximations. We would expect the performance to improve upto a certain K, because for small K, dissimilar Wikipedia concepts may get aggregated together. After the optimum K, the performance would decrease as it becomes more similar to ordinary ESA, ie. the orthogonality assumption becomes invalid.

One drawback of this method is that it assumes that concepts strictly belong to a single cluster in the final clustering that is obtained. However this may not be the case. This can be improved by using Gaussian Mixture Models instead of K-Means Clustering, thus allowing some concepts to come from a hybrid of multiple Gaussians. These methods involving Expectation Maximization are useful as we do not know which concepts or combination of concepts are truly important, and thus the important concepts are latent variables. These can be

found by EM. It is also possible to use other methods apart from GMMs and K-Means as well, which use EM to find these important concepts which control the performance of ESA.

5.4 Vector Based Hybrid Model of ESA and Word2Vec

Since it was observed that ESA and the Google Vectors were the best performing models, a hybrid vector can be formed, which simply increases the dimensionality of the vector by including both the Word2Vec dimensions, as well as the Wikipedia Concept dimensions.

By varying a hyperparameter which scales the Google Vector dimensions, we can change the contribution provided by each individual model to the final vector.

This hybrid could allow us to capture words pertaining to similar concepts, which was not captured by Word2Vec but captured by ESA, and vice versa.

It is also possibile that the combined representation could lead to better representations for a polysemous word, if the word2vec model captures one sense and ESA captures another, since these dimensions are independent, which would be better than any of the individual models.

6 References

- ESA Using Descartes
- N Aggarwal Et al. Non-Orthogonal ESA
- NLTK Documentation