Introducing Inter-Relatedness between Wikipedia Articles in Explicit Semantic Analysis

Naveen E[1 [ME16B077]] and Pawan Prasad K[1 [ME16B179]]

Indian Institute of Technology, Madras
{me16b077,me16b179}@smail.iitm.ac.in

Abstract. Explicit Semantic Analysis (ESA) is a technique used to represent a piece of text as a vector in the space of concepts, such as Articles found in Wikipedia. We propose a methodology to incorporate knowledge of Inter-relatedness between Wikipedia Articles to the vectors obtained from ESA using a technique called Retrofitting to improve the performance of subsequent tasks that use ESA to form vector embeddings. Especially we use an undirected Graph to represent this knowledge with nodes as Articles and edges as inter relations between two Articles. Here, we also emphasize how the ESA step could be seen as a predominantly bottom-up approach using a corpus to come up with vector representations and the incorporation of top-down knowledge which is the relations between Articles to further improve it. We test our hypothesis on several smaller subsets of the Wikipedia corpus and show that our proposed methodology leads to decent improvements in performance measures including Spearman's Rank correlation coefficient in most cases.

Keywords: Wikipedia Graph · ESA · Retrofitting.

1 Introduction

Methods of capturing semantic relatedness in words/documents have been explored to a significant extent by the NLP community. Major approaches include data-driven learning of vector representations [7–9] from a very large corpora. While such *bottom-up* approaches have proven to be effective in recent times, the outcomes of such models can be further amplified by using some form of *top-down* knowledge gained over time, by humans. Such information has been crystallized into structured databases since as early as 1995 [5]. These top-down resources can be utilized to guide bottom-up expeditions to identify relations among words/documents.

Several works in this domain have surfaced recently. One such elegant idea, arrived at by Faruqui et al., is *retrofitting*. We build our proposal on their paper [3], utilizing the technique rather unconventionally, in Explicit Semantic Analysis (ESA) [1], for improving word representations using top-down knowledge from Wikipedia.

Explicit Semantic Analysis is a popular technique that represents the meaning of texts in a high-dimensional space of concepts derived from Wikipedia and using machine learning techniques explicitly represent the meaning of any text as a weighted vector of Wikipedia-based concepts. The information used here from the Wikipedia corpus are essentially the Article titles and the text or body of the Article. We suggest that by using an additional information which is often disregarded, such as the way in which one Wikipedia Article references to another, we could improve upon the earlier work as performed in ESA. One could imagine such references made back and forth between several Wikipedia Articles as a network of useful information. We incorporate this on top of ESA and see some minor improvements on several Benchmark tasks such as Word Similarity tasks like WS-353, RG-65 and MEN3k [10–12]. Especially we see improvements of around 2% on average in performance measured using Spearman's Rank Correlation Coefficient. To represent the knowledge in a useful manner we take advantage of the Graph data structure, specifically an undirected Graph, wherein the nodes are the Articles and edges between nodes correspond to Inter-Relatedness between corresponding Articles.

In addition our work also comprises of implementing the paper - Retrofitting word vectors to semantic lexicons on a number of evaluation benchmarks across several data sets and cross-validating the results cited in the paper [3]. In section 2 we discuss a few papers as background and related work that we consider and take motivation from, while forming our hypothesis. In section 3 we discuss our implementation of the paper that introduces Retrofitting [3] which is a concept that we use to apply the Inter-Relatedness knowledge on top of ESA. In section 4 we introduce our main contributions and hypothesis to improve upon ESA using the discussed knowledge of Inter-Relatedness between Articles and present the results in doing so by drawing comparison before and after incorporating that knowledge.

2 Background and Related Work

2.1 Retrofitting

Faruqui et al. introduced the novel method of retrofitting for capturing relational information among vector representations of words [3]. Retrofitting is applied on word vectors as a *post-processing* step such that, semantically related words will be nearer in their vector embedding space. This includes synonyms, hypernyms, hyponyms and paraphrases for the given word. Such relations among semantic lexicons are captured in resources including the Paraphrase Database [4], WordNet [5] and FrameNet [6].

Although retrofitting is agnostic to the manner in which the vectors were derived, a key point to note is the type of top-down knowledge database to be used for retrofitting. This in fact, depends on the nature of vectors at hand. That is, Faruqui et al. consider *word vector* representations and hence any database

which captures the many relations among *words* would suffice (for example, WordNet, FrameNet). However if one were to refine the vector representations of documents for example, we need to consider a database of similar text documents to construct a graph wherein edges connecting two nodes correspond to two documents having similar content. Our proposal exploits this observation to build a customized graph of Wikipedia articles for retrofitting, as elaborated further in Section 4.

2.2 Explicit Semantic Analysis (ESA)

ESA is a novel method in which semantic relatedness between words or documents are captured by representation of meaning in a high-dimensional space of natural concepts derived from Wikipedia [1]. Each concept is represented as an attribute vector of words that occur in the corresponding wikipedia article. Entries of these concept vectors are assigned weights using TF-IDF scheme. We should note that concepts are merely the titles of articles across which TF-IDF scores are calculated, for example, Computer Science, Automobile or India.

An inverted index is used to retrieve the relevant concepts for a given word. In the words of Markovitch et al., *Each word appearing in the Wikipedia corpus can be seen as triggering each of the concepts it points to in the inverted index* [2]. Moreover, a piece of text (string of words) is simply represented as the centroid of the vectors representing its words in the concept space.

2.3 Non-Orthogonal Explicit Semantic Analysis (NESA)

One of the drawbacks in using ESA to obtain vector embeddings is that ESA inherently assumes that Wikipedia concepts are orthogonal to each other. Therefore, it considers that two words are related only if they co-occur in the same articles. However, two words can be related to each other even if they appear separately in related articles rather than co-occurring in the same articles. In this paper [18], the authors come up with a technique to successfully overcome this issue leading to improvements in several of the Benchmarks and achieving state-of-the-art results.

To compute text relatedness, NESA uses relatedness between the dimensions of the distributional vectors to overcome the orthogonality in ESA model. This is accomplished by utilizing a square matrix $C_{n,n}$ where n is the total number of dimensions or concepts containing the correlation weights between the dimensions. While obtaining relatedness score between two words, instead of using something like a cosine similarity measure which is multiplying the two word vectors and normalizing them, they first multiply the first word vector with the square matrix $C_{n,n}$ and then later multiply this with the second word vector. What this essentially boils down to is that each concept dimension is made to further constitute a semantic vector built in another vector space. To construct $C_{n,n}$ consisting of correlation scores between Articles they provide four different

methods and also discuss the performance improvements achieved by each one of them.

3 Paper Implementation

The objective of retrofitting is to reduce the Euclidean distance between a pair of *similar* vectors and in doing so we end up with a richer representation of words using vectors. Therefore, we want our word vector in question q_i to be close to the observed value $\hat{q_i}$ and close to its neighbours $q_j, \forall j$ such that $(i, j) \in E$, that is they constitute an edge in the graph constructed from lexicon database. Thus, the optimization criteria is [3]:

$$\Psi(Q) = \sum_{i=1}^{n} \left(\alpha_i ||q_i - \hat{q_i}||^2 + \sum_{(i,j)\in E} \beta_{ij} ||q_i - q_j||^2 \right) \tag{1}$$

α *and* β are hyper-parameters controlling the relative strengths of associations between two vectors
n = Number of word vectors to be retrofitted
Differentiating ψ with respect to q_i gives the following online update equation (refer paper by Faruqui et al. for details [3]):

$$q_i = \frac{\sum_{j:(i,j)\in E} \beta_{ij} q_j + \alpha_i \hat{q_i}}{\sum_{j:(i,j)\in E} \beta_{ij} + \alpha_i} \tag{2}$$

Following the implementation details stated in [3], we apply retrofitting on 3 different sets of word vectors and by using 4 different semantic lexicon resources. We assess the performance of each combination with and without retrofitting on standard lexical semantic evaluation tasks. An analysis of the results obtained (see Table 4) as well as comparison with the results reported in [3] are detailed in Section 3.4. Refer Section 3.3 for details regarding evaluation tasks.

3.1 Semantic Lexicons

We use Paraphrase Database (PPDB), WordNet$_{syn}$ (WN_{syn}, captures only synonymous relations between words), WordNet$_{all}$ (WN_{all}, captures relations including synonyms, hypernyms and hyponyms) and FrameNet [4–6]. The size of the graphs obtained from the different top-down knowledge databases are shown in Table 1.

3.2 Word Vectors

We use publicly available pre-trained English word vectors for implementation. These include the Glove, Skip-Gram (SG) and Global Context (GC) vectors[7–9]. The dimensions and training corpus characteristics for each set of word vectors are enlisted in Table 2. We once again emphasise here that retrofitting is independent of the training models used for extracting vector representations [3].

Table 1. Size of graphs obtained from lexicon databases

Lexicon	Words	Edges
PPDB	$102,90$	$374,555$
WordNet$_{syn}$	$148,730$	$304,856$
WordNet$_{all}$	$148,730$	$934,705$
FrameNet	$10,822$	$417,456$

Table 2. Word Vector Characteristics

Word Vector	Dimension	Training Corpus
Glove Vectors	300	6 Billion tokens
Skip-Gram Vectors (SG)	300	100 Billion tokens
Global Context Vectors (GC)	50	1 Billion tokens
Multilingual Vectors (Multi)	300	100 Billion tokens

3.3 Retrofitting: Evaluation Benchmarks

The resultant vector representations are evaluated on tasks that test how well they capture both semantic and syntactic aspects of the representations along with an extrinsic sentiment analysis task. A summary of the different tasks are provided in Table 3. For the sake of comparison of results, we perform the same evaluation tasks as given in the base paper [3].

We should note that the Syntactic Relations task is computationally expensive since for each list of 4 words (refer Section 3.4), we find cosine similarity with each of the word types in the vector representations. Moreover, since the Skip-Gram vector set is enormous ($\approx$ 3GB compressed and $\approx$ 8GB in .txt file) the estimated time was large($\approx$ 22 hours) and we were facing issues with physical memory.

Table 3. Evaluation metrics and Dataset characteristics

Evaluation Task	Metric	Dataset
Word Similarity: MEN-3k	Spearman's rank correlation coefficient	3000 word pairs
Word Similarity: RG-65		65 word pairs
Word Similarity: WS-353		353 word pairs
Synonym Selection (TOEFL)	Accuracy	80 sets of 5 words
Sentiment Analysis (SA)		6920 + 872 + 1821 sentences

Word Similarity We evaluate on 3 different benchmarks commonly used for measuring word similarity. These are the WS-353 dataset [10] containing 353 pairs of English words that have been assigned similarity ratings by humans, the RG-65 dataset [11] that contain 65 pairs of nouns and the MEN dataset [12]

consisting of 3,000 word pairs. Assessing the similarity of words amounts to comparing the corresponding vectors using the cosine metric. We report Spearman's rank correlation coefficient between the rankings produced by our model against the human rankings.

Synonym Selection (TOEFL) The TOEFL synonym selection task is to select the semantically closest word to a target word from a list of four candidates [13]. The dataset contains 80 such questions. An example is "rug → sofa, ottoman, carpet, hallway", with carpet being the most synonym-like candidate to the target word.

Sentiment Analysis (SA) Socher et al. created a treebank containing sentences annotated with labels on phrases and sentences from movie review excerpts [14]. The coarse grained treebank of positive and negative classes has been split into training, development, and test datasets containing 6,920, 872, and 1,821 sentences, respectively. We train an L2-regularized logistic regression classifier on the average of the word vectors of a given sentence to predict the coarse-grained sentiment tag at the sentence level, and report the test-set accuracy of the classifier.

Syntactic Relations (SYN-REL) The dataset is composed of analogous word pair that follow a common syntactic relation [15]. For example, walking and walked; The task is to find a word (word vector) q_d that best fits the relation $q_a : q_b :: q_c : q_d$, given q_a, q_b, q_c. An estimate of q_d is obtained by using the vector offset method [8, 16] as follows:

$$\begin{aligned} & q_a : q_b :: q_c : q_d \\ & \quad \Rightarrow q_a - q_b = q_c - q_d \\ & \quad \Rightarrow q_d = q_b - q_a + q_c \end{aligned}$$

Therefore, we find the vector q which has the highest cosine similarity to the estimate.

3.4 Results of Retrofitting

In table 4, the results of our implementation of retrofitting are presented. We present the absolute scores in each case of retrofitting. Note that the first three columns represent Spearman's rank correlation coefficient and the last two represent Accuracy. Note that all absolute values of performance measures are scaled to 100 and reported as percentages. Key observations based on the model performance are enlisted below:

– Firstly we note that with similar hyperparameter settings, we obtain an equivalent performance corresponding to the results given in base paper [3].

Table 4. Results of absolute performance with retrofitting obtained by us; Spearman's correlation (columns 2,3 and 4 from left) and accuracy (columns 5,6 and 7 from left) on different tasks. Higher scores are always better.

Word Vector+Lexicon	MEN-3k	RG-65	WS-353	TOEFL	SA	SYN REL
GLOVE	73.75	76.95	60.85	88.75	78.31	78.1
GLOVE+PPDB	76.36	82.49	63.08	93.75	79.24	77.1
GLOVE+WN_{syn}	73.71	79	61.34	93.75	78.97	66.8
GLOVE+WN_{all}	75.94	85	61.52	91.25	79.24	66.1
GLOVE+FN	70.09	75.62	56.14	91.25	77.76	69.0
SG	73.22	76.08	65.9	87.5	81.82	74.2
SG+PPDB	75.25	81.5	71.28	95	82.59	77.7
SG+WN_{syn}	72.81	76.35	66.47	93.75	82.31	59.9
SG+WN_{all}	73.65	84.39	67.4	93.75	81.82	59.1
SG+FN	69.88	79.02	61.84	87.5	81.38	64.64
GC	31.4	62.99	62.58	57.5	67.38	18.8
GC+PPDB	40.31	68.99	64.58	71.25	68.92	27.3
GC+WN_{syn}	34.87	69.3	63.18	63.75	68.53	13.8
GC+WN_{all}	38.01	73.23	64.8	62.5	68.7	13.9
GC+FN	33.15	67.01	62.57	61.25	67.49	14.7

- We obtain major improvements in word similarity tasks (columns 1-3 in table 4) when implementing retrofitting with all the top-down knowledge databases with the exception of FrameNet. In fact, with the use of FrameNet on any evaluation task, we seldom observe any improvements (in some cases, we observe decrease in performance) with retrofitting. Faruqui et al. cite the reason being that the nature of top-down knowledge captured by FrameNet between words is very abstract and distantly related, (for example, *prominent* and *eye-catching*).
- In the case of TOEFL task, we observe very high improvements in accuracy, the highest increment (+13.5%) occurring in the case of GC vectors with PPDB retrofit.
- We observe modest improvements in the case of Sentiment Analysis task (SA), the highest increment (+1.5%) occurring in the case of GC vectors with PPDB retrofit.
- For the SYN-REL task, we observe improvement in case of GC (+8.5%) and SG (+3.5%) with PPDB retrofit. For all other cases, we always observe drop in accuracy, the highest reduction occurring in the case of SG vectors with WN_{all} retrofit (−15.1%). As stated by Faruqui et al. in the base paper [3], the poor performance can be attributed to the fact that SYN-REL is inherently a syntactic task and by retrofitting, we only incorporate semantic information in the vector representations.
- Finally, we note that retrofitting with PPDB database gives the best improvement across a majority of the combinations (Word vector set + Evaluation task), followed by WN_{all}, same as in the case of results provided in the base paper [3].

4 Retrofitting in ESA

We aim to incorporate the knowledge of Inter-Relatedness between Wikipedia Articles into ESA to improve the quality of vector representation of words/texts and we use the method of retrofitting to accomplish this. Specifically, we retrofit the concept vectors which are vector embedding in the space of words using a graph like network consisting of the inter-relatedness knowledge. So, once we arrive at the representation of Wikipedia Articles as an attribute vector of words by using the TFIDF scheme, we incorporate semantic relatedness in vector embedding of Articles by means of an information which is the inter-relatedness of documents/articles in Wikipedia.

There might be cases where two words can be related to each other even if they appear separately in related articles rather than co-occurring in the same articles which would then indicate that the word vectors have different representations and thus not similar. This is one of the drawbacks of ESA which we hope to overcome by incorporating our proposed methodology to enhance the word vectors obtained through ESA. Note that in the paper Non-Orthogonal Explicit Semantic Analysis (NESA) [18] the authors also try to overcome a similar problem but one of the key difference is that they accomplish this by making each concept dimension further constitute a semantic vector built in another vector space. Whereas we aim to solve this by retaining the concept space by using retrofitting as a means to enhance the vector representations using knowledge from Inter-Relatedness Graph obtained from Wikipedia corpus.

If we view an article in Wikipedia, in its description or content, there are links to other related Wikipedia articles in sections such as “References”, “See also” etc. and also as hyperlinks within the body of the Article. This type of information is particularly disregarded when working with plain text in ESA. We incorporate this form of knowledge so that closely interrelated concepts are nearer in their corresponding weighted representation of words. For the purpose of retrofitting, we build a graph-like data structure representing the inter related concepts explained in detail in Section 4.1. Then we modify the weighted representation of words for each concept to be nearer to itself as well as its related concepts using the same optimization criteria arrived at by Faruqui et al. (refer Equations 1 and 2).

4.1 Wikipedia Inter-Relatedness Graph (Wiki Graph)

The information that we are willing to retrofit the tf-idf matrix in ESA with is the inter relatedness between Wikipedia Articles. To do so, we construct an undirected graph with nodes as Article Titles and edges are constructed between two inter-related Articles. Inter-related articles could be viewed as the Top-Down Knowledge that we are incorporating into ESA while ESA is primarily a Bottom-Up approach. The main source of inter-relatedness is the hyperlinks present in each article which in Wikipedia parlance is called as internal links or wikilinks.

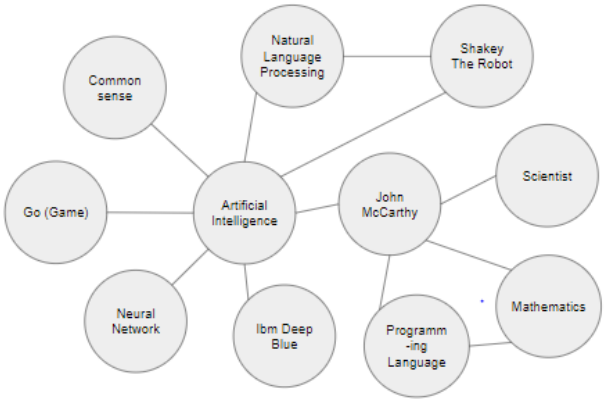

Fig. 1. A snapshot of a tiny portion of Wikipedia inter-relatedness graph (undirected) showing nodes as Wikipedia Articles and edges between two related Articles.

There are two ways that we have come up with to generate the Graph. One is based on prior work by [17] where they extract the hyperlinks from main text in each article which leads to another article within Wikipedia. The graph constructed this way is done so for the entire Wikipedia corpus at specific timestamps across history and we'll be using the latest one for our task in hand. Based on the corpus or the subset of the Wikipedia Articles that we are working with in ESA we extract the corresponding subgraph containing relations between only these Articles. Another method that we have come up with and is more efficient is directly parsing them from the raw data that we extract as dump from *Wikipedia*[1]. Each Article is represented as a json object and the key links_out contains as value a list of all Wikipedia Articles that have been referred to from the original Article. Using this we reconstruct the graph from scratch specific to the dataset or corpus in hand. This is faster in computation time since we are constructing from base-up only what we need instead of having to reduce a much bigger graph. Note that for purposes of reusing the graph, we save them in the form of an adjacency list in text file in local storage.

Owing to space complexity, as discussed in later sections for storing the tf-idf matrix, we are working with a smaller corpus derived for each specific Evaluation task and the structure for the corresponding graphs is summarized in Table 5. A tiny portion of the graph is illustrated in Figure 1 to show how Articles may be inter-related with each other.

[1] https://en.wikipedia.org/wiki/Special:Export

Table 5. Summary of the extracted Graphs corresponding to each corpus that we use in ESA. Articles per word denotes the number of articles extracted to form the corpus for each word in the dataset corresponding to the Evaluation Task

Evaluation Task	Articles per word	Articles	Nodes	Edges	% Wikipedia x(10^{-2})
WS-353	5	2133	1459	4637	5.3325
	6	2555	1768	5626	6.3875
RG-65	10	498	150	179	1.245
	20	993	338	445	2.4825
MEN-3k	3	2337	1775	8225	5.8425
	5	3972	3026	13244	9.93
WikiLink Graphs	-	13,685,337	13,685,337	163,380,007	-

4.2 Implementation

In Appendix 1, we present a detailed example for implementation of our proposal. An overview is provided in the subsequent paragraphs.

Firstly, the Wikipedia concepts are represented as vector embedding using words as dimensions with the TF-IDF scheme following the procedure for representing concepts in word space given in [1]. Then, the graph required for representing the inter-relatedness of Wikipedia concepts are extracted. (Section 5.1).

The TF-IDF vectors for each concept are now retrofitted in the same fashion as in [3] wherein the objective function (Eq. 1) is minimized thus resulting in the update step (Eq. 2).

Note that owing to the limited availability of compute resources at the time, the TF-IDF matrix for the entire Wikipedia corpus as well as an alternate (simple-Wikipedia) were unable to fit in the physical memory. Hence, we work on a much smaller corpora and extract a subset of Wikipedia pertaining to each Evaluation task (see Section 4.3) we consider at hand. To do so we take the top n articles obtained by searching for each word used in the Evaluation dataset and then compiled them together.

4.3 Evaluation

We evaluate our model on word similarity tasks using the standard benchmarks WS-353, RG-65, and MEN-3k dataset [10–12]. A summary of vector characteristics along with performance of our model on the benchmarks are given in Table 6.

Word Vectors Owing to the computational constraints for testing our hypothesis on the entire Wikipedia corpus we derive much smaller subsets of Articles from Wikipedia separately for each corresponding Evaluation Benchmarks that we perform. For a given evaluation dataset, we extract the relevant concepts for each of the words present in the dataset from Wikipedia corpus using the

Wikipedia library by uniformly extracting a fixed number of concepts for each word. We then construct our corpus wherein words are expressed in the space of the extracted concepts alone. For example, RG-65 (10) implies that for each word in RG-65 dataset top 10 concepts are extracted (See Column 1, Table 6). Thus,in the resulting corpus, words are expressed in the 498 dimensional space of concepts (Total extracted concepts $= 2 * 65 * 10 = 1300$ of which 498 are unique, see Column 2, Table 6). As in ESA [1], we express a concept in terms of words occurring in corresponding Wikipedia article using TF-IDF scheme. Refer Columns 1-3, Table 6 regarding the vector characteristics used for evaluation.

Once we obtain the Word/Concept matrix, we apply retrofitting on concept vectors using the top-down knowledge graph specific to the evaluation dataset (refer Table 5 and Section 4.1). Results of our model implementation are summarized in the next section.

4.4 Results

We ran experiments for each of the evaluation task coupled with it's characteristic corpus as shown in Table 5 and report the Spearman's Rank correlation coefficient. To validate our hypothesis we compare the before and after results by first performing Explicit Semantic Analysis for each task followed by introducing inter-relatedness between Wikipedia Articles using Retrofitting. These are summarized in Table 6.

Note that all absolute values of performance measures are scaled to 100 and reported as percentages. Overall from the several Benchmarks that we ran we are able to show that on average using our method results in improvements in the performance measure.

- For the WS-353 Benchmark, we observe an increase in performance by 2% and 1% for WS-353 (5) and WS-353 (6) variants where the number inside brackets denote the number of Articles used to form the corpus per word in the Benchmark dataset.
- We observe the highest improvement (+3.01%) in case of RG-65 (20) (RG-65 Benchmark with top 20 concepts extracted for each word).
- Whereas, in the case of MEN-3k Evaluation task we observe a slight improvement (+0.15%) as well as a slight decrement (−0.39%) in performance compared to vanilla ESA for the MEN-3k (5) and MEN-3k (3) variants respectively. Hence, we can say that there is not enough evidence that our methodology provides performance improvements when measured using the MEN-3k Benchmark task. Furthermore for most of the tasks the performance increases by $1 - 2\%$ unless when there is no clear improvements.

Next, we compare our model performance against results obtained with Glove, Skip Gram (SG) and Global Context (GC) word vectors on word similarity tasks before and after retrofitting with different semantic lexicon databases

(See Table 4 and refer Section 3.4). Note that we use a much smaller percentage of Wikipedia compared to the case of Glove and GC vectors owing to computational constraints (See Table 5) and has to be taken into account while comparing the results.

- On the MEN-3k benchmark, the highest correlation obtained by our methodology is 66.06% (MEN-3k (3) + Wiki Graph). Our results (ESA with Wiki Graph retrofit) are better than the performance of GC with/without retrofit by a margin of atleast +22.55%. However, both SG and Glove vectors with/without retrofit give better results compared to our model (minimum and maximum margins of +3.82% and +10.3% respectively).
- Evaluating our model on RG-65 benchmark, the highest correlation obtained is 81.30% (RG-65 (10) + Wiki Graph). Once again, we produce better results with our model (ESA with Wiki Graph retrofit) than GC with/without retrofit by a margin of atleast +8.07%. Performances of SG and Glove are similar to our model except in the case of retrofit with either PPDB or WN_{all}, wherein large improvements and higher correlations (85%) are observed (refer Section 3.4).
- On the WS-353 benchmark, we obtain 56.84% correlation (WS-353 (5) + Wiki Graph). With the exception of Glove + FrameNet retrofit, all combinations of word vectors + top-down database retrofit result in superior performance than our model (minimum and maximum decrements of −4.01% and −14.44% respectively).

Table 6. Results of absolute performance obtained using ESA and with our proposed methodology. The metric for performance is Spearman's Rank correlation coefficient (Higher scores are always better). Note that for a given evaluation task, we use different corpora. For example, in case of WS-353 (5), words are expressed in the 2133 dimensional space of concepts (Column 2, Row 1), which is a consequence of constructing corpora by extracting only the top 5 concepts for each word.

Evaluation Task	**Dimension (Word Vector)**	**Dimension (Concept Vector)**	**Vector Representation**	**Performance**
WS-353 (5)	2133	162080	ESA	54.61
			(+) Wiki Graph	56.84
WS-353 (6)	2555	183592	ESA	55.12
			(+) Wiki Graph	55.97
RG-65 (10)	498	44401	ESA	79.84
			(+) Wiki Graph	81.30
RG-65 (20)	993	67881	ESA	78.09
			(+) Wiki Graph	81.10
MEN-3k (3)	2337	189661	ESA	66.45
			(+) Wiki Graph	66.06
MEN-3k (5)	3972	253207	ESA	62.86
			(+) Wiki Graph	63.01

4.5 Conclusion & Future Work

Through our work in this project we first try to implement a paper conveying an elegant solution for enhancing Word Vectors by means of Retrofitting Semantic Lexicons to gain a deeper understanding on the subject and how research ideas are formed, hypothesized and evaluated. Then we propose our idea of improving the vector embeddings obtained through Explicit Semantic Analysis by introducing top-down knowledge of Inter-Relatedness between Wikipedia Articles. We do so by constructing this knowledge using the network of hyperlinks in each Article linking other similar Articles to itself and formally storing it as an undirected graph with nodes as Articles and Edges showing Inter-relatedness between two such nodes.

Based on our experimentation with different datasets, each evaluated using different Benchmarks, we show that this methodology in fact leads to decent improvements in performance of the Word vectors obtained after introducing this top-down knowledge by means of Retrofitting to ESA. In the future this could be extended to much larger dataset like the entire Wikipedia corpus to possibly realize state of the art results since we had to limit ourselves to smaller subsets of Wikipedia corpus owing to the computational constraints present at the moment.

References

1. E. Gabrilovich and S. Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In Proceedings of the 20th International Joint Conference onArtificial Intelligence (IJCAI 07), pages 1606–1611. MorganKaufmann Publishers Inc., 2007.
2. Egozi, O., Markovitch, S., Gabrilovich, E.: Concept-Based Information Retrieval using Explicit Semantic Analysis. ACM Transactions on Information Systems 29(2), 8:1–8:34 (2011)
3. Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015a. Retrofitting word vectors to semantic lexicons. In Proceedings of NAACL.
4. Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In Proceedings of NAACL.
5. George A Miller. 1995. Wordnet: a lexical database for english. Communications of the ACM.
6. Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. ACL '98.
7. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In Proceedings of EMNLP.
8. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
9. Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In Proceedings of ACL.

10. Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: the concept revisited. In WWW, New York, NY, USA.
11. Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. Commun. ACM, 8(10):627–633, October.
12. Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In Proceedings of ACL.
13. Thomas K Landauer and Susan T. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological review.
14. Richard Socher, Alex Perelygin, JeanWu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of EMNLP
15. Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In Proceedings of NAACL.
16. Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In Proceedings of CoNLL.
17. Consonni, C., Laniado, D. and Montresor, A., 2019, July. WikiLinkGraphs: A complete, longitudinal and multi-language dataset of the Wikipedia link networks. In Proceedings of the International AAAI Conference on Web and Social Media (Vol. 13, No. 01, pp. 598-607).
18. N. Aggarwal, K. Asooja, G. Bordea, and P. Buitelaar. Non-orthogonal explicit semantic analysis. Lexical and Computational Semantics (* SEM 2015), pages 92–100, 2015

5 Appendix 1

In this section, we provide a step by step walkthrough for implementing our proposal (refer Section 4). Consider the piece of text, "Information Retrieval in Search Engines". After pre-processing the query we are left with the following list of tokens:

['information', 'retrieval', 'search', 'engines']

In short, we are left with 4 word tokens:

$$[w_1, w_2, w_3, w_4]$$

In ESA [1], we represent each of these words as vectors in the concept space. We first retrieve the concepts corresponding to the words using inverted indexing. We then represent the concepts themselves as a vector embedding of the words which occur in the article corresponding to the concept (concept is just the title of article).

$$w_1 \rightarrow doc_1,\ doc_2,\ doc_3$$
$$w_2 \rightarrow doc_1,\ doc_2$$
$$w_3 \rightarrow doc_1,\ doc_3$$
$$w_4 \rightarrow doc_2,\ doc_3$$

Table 7. Word/Concept Matrix. Concepts can be expressed in 4 dimensional space of words occuring in corresponding articles. Consequently, words can be expressed in 3 dimensional space of concepts.

Word/Concept	c_1	c_2	c_3
w_1	tf_1^1	tf_1^2	tf_1^3
w_2	tf_2^1	tf_2^2	0
w_3	tf_3^1	0	tf_3^3
w_4	0	tf_4^2	tf_4^3

Thus, w_2 occurs in doc_1, doc_2, words w_1,w_2,w_3 occur in doc_1 and so on. This relationship is captured in the word/concept matrix (refer 7).

The concept vectors $\vec{c_1}$, $\vec{c_2}$, $\vec{c_3}$ corresponding to doc_1, doc_2, doc_3 respectively are vectors of words:

$$\vec{c_1} = tf_1^1\ \hat{w_1} + tf_2^1\ \hat{w_2} + tf_3^1\ \hat{w_3}$$
$$\vec{c_2} = tf_1^2\ \hat{w_1} + tf_2^2\ \hat{w_2} + tf_4^2\ \hat{w_4}$$
$$\vec{c_3} = tf_1^3\ \hat{w_1} + tf_3^3\ \hat{w_3} + tf_4^3\ \hat{w_4}$$

$c_i \implies i^{th}\ concept$
$w_i \implies i^{th}\ word$
$tf_i^j \implies TF - IDF\ of\ i^{th}\ word\ occurring\ in\ j^{th}\ document$

Applying retrofitting (using Eqn. 1 and 2 from Section 3) to these concept vectors based on the Wikipedia graph, we get,

$$\vec{c_1} = {tf_1^1}^R\ \hat{w_1} + {tf_2^1}^R\ \hat{w_2} + {tf_3^1}^R\ \hat{w_3}$$
$$\vec{c_2} = {tf_1^2}^R\ \hat{w_1} + {tf_2^2}^R\ \hat{w_2} + {tf_4^2}^R\ \hat{w_4}$$
$$\vec{c_3} = {tf_1^3}^R\ \hat{w_1} + {tf_3^3}^R\ \hat{w_3} + {tf_4^3}^R\ \hat{w_4}$$

${tf_i^j}^R \implies TF - IDF\ of\ i^{th}\ word\ occurring\ in\ j^{th}\ document\ after\ retrofit$
For example, ${tf_1^3}^R$ is the vector weight of w_3 occurring in doc_1 after retrofit.

Ultimately, we arrive at the word vectors $\vec{w_1}, \vec{w_2}, \vec{w_3}, \vec{w_4}$ which are vector embedding in the concept space, where the weight (magnitude) corresponding to concept $\vec{c_j}$ for i^{th} word vector $\vec{w_i}$ is given by ${tf_i^j}^R$. Thus,

$$\vec{w_1} = {tf_1^1}^R\ \hat{c_1} + {tf_1^2}^R\ \hat{c_2} + {tf_1^3}^R\ \hat{c_3}$$
$$\vec{w_2} = {tf_2^1}^R\ \hat{c_1} + {tf_2^2}^R\ \hat{c_2}$$
$$\vec{w_3} = {tf_3^1}^R\ \hat{c_1} + {tf_3^3}^R\ \hat{c_3}$$
$$\vec{w_4} = {tf_4^2}^R\ \hat{c_2} + {tf_4^3}^R\ \hat{c_3}$$

Therefore, the resultant vector embedding in the concept space for "Information Retrieval in Search Engines" is given by,

$$\begin{aligned}\vec{W} &= \vec{w_1} + \vec{w_2} + \vec{w_3} + \vec{w_4} \\ \vec{W} &= (tf_1^{1R} + tf_2^{1R} + tf_3^{1R})\hat{c_1} \\ &\quad + (tf_1^{2R} + tf_2^{2R} + tf_4^{2R})\hat{c_2} \\ &\quad + (tf_1^{3R} + tf_3^{3R} + tf_4^{3R})\hat{c_3}\end{aligned}$$