ISSN 2395-8618

Word Embeddings and Length Normalization for Document Ranking

Sannikumar Patel, Markus Hofmann, and Kunjan Patel

***Abstract*—Distributed word representation techniques have been effectively integrated into the Information Retrieval retrieval task. The most basic approach to this is mapping a document and the query words into the vector space and calculating the semantic similarity between them. However, this has a bias problem towards documents with different lengths, which rank a small document higher compared to documents with larger vocabulary size. While averaging a document by mapping it into vector space, it allows each word to contribute equally, which results in increased distance between a query and the document vectors. In this paper, we propose that document length normalization should be applied to address the length bias problem while using embedding based ranking. Therefore, we have presented an experiment with traditional Length Normalization techniques over, *word2vec* (Skip-gram) model trained using the TREC Blog06 dataset for ad-hoc retrieval tasks. We have also attempted to include relevance signals introducing a simple Linear Ranking (LR) function, which considers the presence of query words in a document as evidence of relevancy while ranking. Our combined method of Length Normalization and LR significantly increases the Mean Average Precision up to 47% over a simple embeddings based baseline.**

***Index Terms*—Word embeddings, neural information retrieval, distributed word representations, Word2Vec.**

I. Introduction

MATCHING semantically similar documents to a query is a core challenge in ad-hoc retrieval. For large scale search engines, this problem does not seem too complicated, as it is possible to identify similar documents by considering user behavioral data such as clicks and hyperlinks as ranking measures [1]. However, for many other Information Retrieval (IR) tasks, it is quite challenging to identify a relevant set of documents and rank them correctly. Current methods for relevancy matching are mainly based on various term-frequency based approaches. However, these methods fail to correctly identify relevant matches whenever a document contains the query words without being relevant, or it is using different vocabulary. These existing methods for similarity matching measure a count of query words in the set of documents.

BM25 is such a traditional model that considers query count as evidence of similarity in the document [2]. The main idea behind this model is to consider a count of query words in a document as evidence of similarity, whereas non-query words are less useful for ranking. However, this model is less useful when looking for similarity in semantically different documents.

Addressing this problem requires a set of techniques that consider word semantics at the root and not solely depend on term-frequency. One such semantically diverse method is Distributed Word Representation, which captures a precise semantic word relationship by learning high quality distributed vector representations [3]. Distributed Word Representations are based on the idea of statistical language modeling and feature learning where words or phrases from the vocabulary are mapped to vectors of real numbers [4], [3]. A Skip-gram and Continues Bag of Word (CBOW) are examples of such a method. These methods use Neural Network models to learn vector representation of words from unstructured text data [4]. The main objective of these models is to find word representations that are useful for predicting the surrounding words in a sentence or context. Vectors generated by this model are useful when considering the semantic similarity between words or documents. For this reason, the Neural Network-based Distributed Word Representation approaches are widely used in Natural Language Processing (NLP) tasks such as automatic question answering and machine translation [5], [6]. However, after observing its success and usefulness in a wide range of NLP tasks, some studies have started utilizing it for query-document similarity matching in ad-hoc retrieval [7], [8], [9], [10], [11]. The core idea is to map all document terms against query terms in a shared semantic space and formulating a similarity score by applying measuring techniques such as *cosine similarity* or *euclidean distance*. Therein it gives a relevancy score between query and document, which later can be used to rank documents [8].

In document ranking, we wish to reasonably rank documents without biasing them based on different factors such as its length, vocabulary size, the occurrence of query word, *etc*. In a traditional term-frequency based ranking systems, there are plenty of ways for rewarding and penalizing documents based on such specific parameters. One such inevitable approach is the normalization of the document's length. The length of target documents is one of the most significant factors which considerably affects its ranking [12], because the same term may repeatedly occur in long documents. For example, in a document about a *dog* that has 1,000 terms, it is more likely to have a frequent occurrence of *dog*, than a document with just

Manuscript received on April 27, 2019, accepted for publication on August 12, 2019, published on December 30, 2019.

The authors are with the Technological University Dublin, Ireland (e-mail: Patel.Sannikumar@mytudublin.ie, Markus.Hofmann@tudublin.ie, kunjanpatel@hotmail.com).

ISSN 2395-8618

300 terms. This will help documents with increased length to being ranked first even if it is not necessarily relevant to the query.

In the contrary, higher ranking of long documents in term-frequency based approaches, are usually ranked lower in embeddings based methods, because with a higher number of terms in the document its centroid is also likely to be sparse *(as every word in document contributes equally)* which results in increased distance between query vector and centroid. The proof is shown in Fig. 1a where long documents are ranked lower despite being relevant *(red * in the figure)* to the query. However, after normalization, the size of the document gets decreased as additional noisy terms are removed, and as a result, relevant documents are ranked higher.

Fig. 1b shows how relevant documents are pushed forward after length-normalization. The main aim of this study is to evaluate whether it is possible to reduced length biased ranking in the embedding approach by applying traditional document normalization techniques. Therefore, we present an experiment with two normalization techniques over the Blog06 dataset [13]. Moreover, for evaluation purpose, we have used Mean Average Precision (MAP). In subsequent sections, we discuss the methodology and a set of experiments performed over the embeddings space model and Length Normalization methods. Then we discussed the results, followed by our conclusion.

II. Related Work

Word embeddings learned unsupervisedly have been surprisingly successful in many NLP tasks. In fact, in many NLP architectures, they have almost completely replaced traditional distributed feature techniques such as Latent Semantic Analysis (LSA) [14]. For many years vector space models have been used in word semantics related tasks and evolved over-time. Latent Dirichlet Allocation (LDA) and LSA are example of models used extensively before the invention of neural word embeddings [15], [14].

A foundation for neural word embeddings were established after the introduction of a unified architecture for NLP [16]. However, term *word embeddings* appeared first in 2003 when researchers presented a study training joint neural language models [17]. The Neural Networks based Skip-gram and CBOW model introduced in 2013 made word embeddings popular among the NLP community, and a toolkit of this model called *word2vec* was made available [4]. Despite being so popular, one major downside of *word2vec* is dealing with terms missing in embedding. To address this problem, an extension to this called *FastText* was introduced. In contrast to *word2vec*, *FastText* treats each word as a composition of character n-grams [18]. Further, researchers released the co-occurrence matrix-based model, signaling that word embedding has reached the mainstream in NLP [19]. After this, many advanced embeddings models have been invented using Neural Networks, overcoming different problems such as missing words and the context of words [20], [21].

However, the earliest study with word embedding for IR was anticipated for retrieval and clustering experiments in which the traditional topic model was used to calculate word embeddings [22]. Usually, strategy for using word embeddings in IR involves deriving a dense vector representation for the query and document terms from embedding space, and then an average of the derived vectors is used as document vector to measure semantic similarity between them, which is relatively easy and popular in the community [7], [8]. After an introduction of word embeddings for IR tasks, it has grown beyond what can be concisely described here. Especially, *word2vec* saw wider adoption in the community [23], [24], [25], [26]. Further, in another study, the impact of word embeddings on scoring in practical IR has evaluated extensively by re-weighting terms using Term-Frequency scores and re-ranking documents based on Word Mover Distance (WMD) [27], [28]. In an attempt to ad-hoc retrieval, studies has also presented experiments with multiple vector spaces *(know as IN and OUT)* available in *word2vec* [4], [25].

In which, they tried to map a query and document terms in different vector spaces *(such as query into IN and document into OUT)* and then ranked them based on cosine similarity scores [25].

However, word embeddings are not just used for similarity matching between query and document some studies have demonstrated that word embeddings such as *word2vec* and GloVe, trained globally, can be used for query expansion [29]. In another similar study, the author has introduced an Artificial Neural Network classifier to predict the usefulness of expanded query terms over word embeddings. They concluded that terms selected by the classifier for expansion significantly improve retrieval performance [30]. Further, instead of just averaging word vectors, a generalized language model has constructed to derive transformation probabilities between words by using word embeddings, and it showed significant improvement over language model baselines [24]. Beyond this, few studies have also explored learning embeddings by utilizing clickthrough and session data [31], [32]. We can say concisely that, with the new development in Neural Networks, word embeddings are further improving, opening new possibilities for its application to IR tasks.

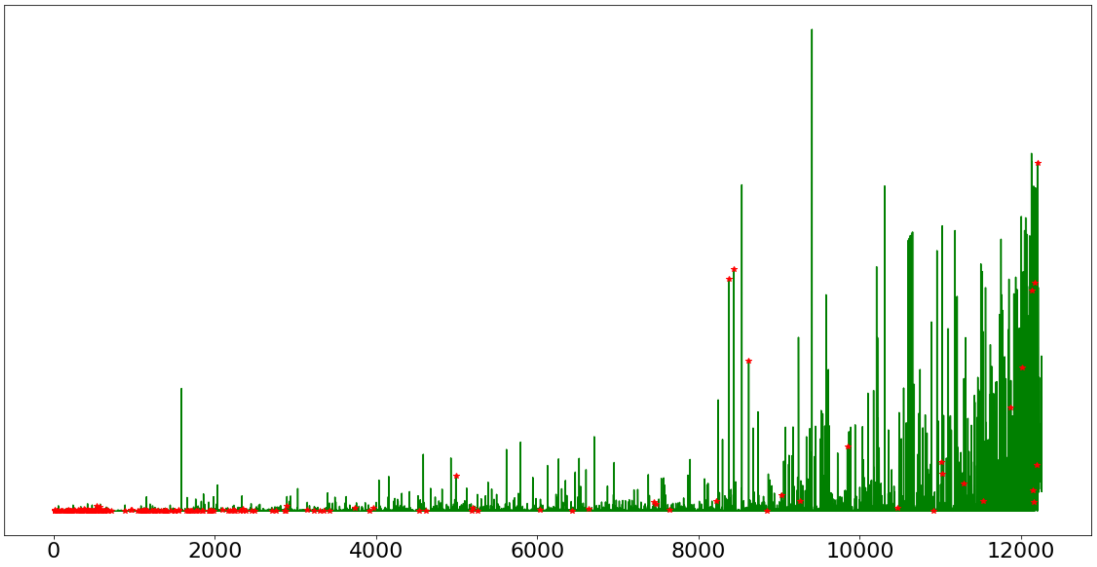
III. Methodology

In this section, we first give a formal introduction to the distributed word representation techniques and Document Length Normalization. Then, we present our embedding space model using Length Normalization and Linear Ranking for document ranking.

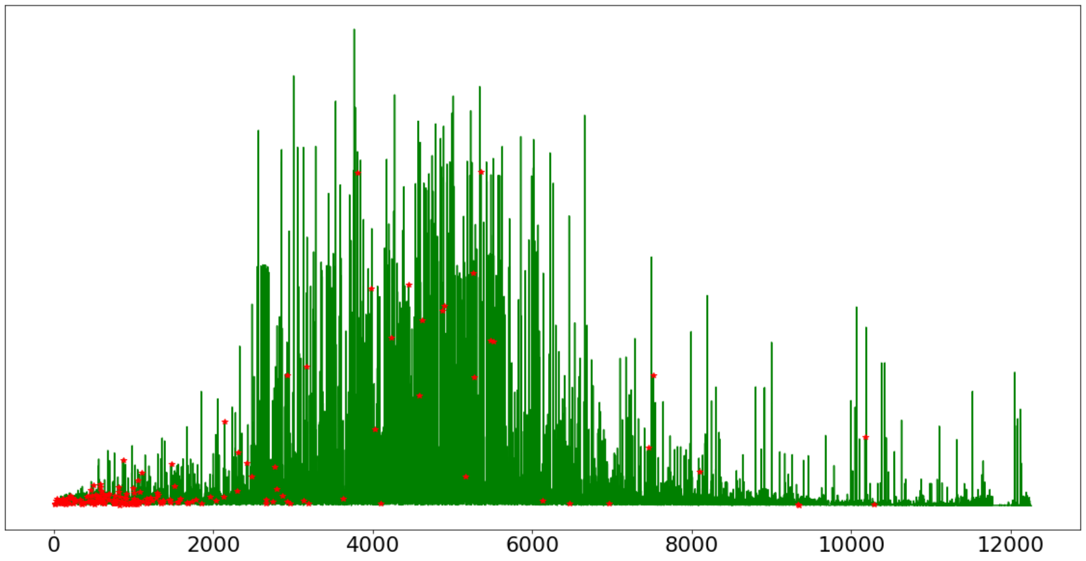
A. Distributed Word Representation

Many traditional frequency-based word representation techniques such as Term Frequency - Inverse Document Frequency, and co-occurrence matrix, are less efficient to capture word semantics. Therefore, its applicability is less

ISSN 2395-8618



(a) Large documents are ranked lower in simple embedding based approach.

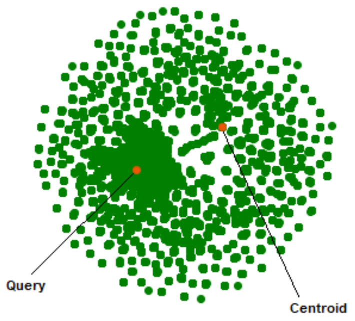


(b) Ranking improves with length-normalization as large documents are pushed forward.

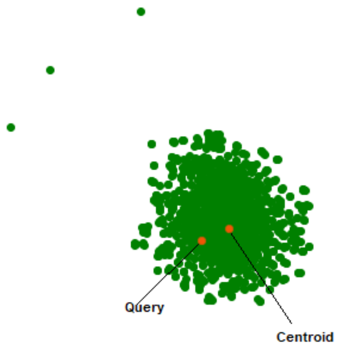
Fig. 1. Projection of initial12K ranked documents along with their length before normalization and after normalization while using the embedding space model for ranking.*(red [*] denotes a relevant document to the selected query).*

effective for document ranking as it does not preserve semantic relations, resulting in a loss of useful information. However, to deal with this problem, the researchers have started using distributed representation based CBOW and Skip-gram techniques, which preserves word semantics. So far, these techniques have outperformed in various NLP tasks.

The widely used Skip-gram and CBOW are the models in this category that learns word embeddings based on the probability concept. Such as predicting the occurrence of a word around other words in a fixed-length window. In CBOW, the word in a window serves as an input and then the model attempts to predict context words. Opposite to this, a Skip-gram model attempts to achieve the reverse of what the CBOW model does by predicting a context from a given set of target words [4]. Even though CBOW and Skip-gram use different formulation techniques, they are simple Neural Networks with a single hidden layer to perform a particular prediction task based on sentence semantics. Weights learned in this hidden layer are called vectors, and each vector in the layer is associated with a single word representing the context of the word, which is also called *embeddings*. In general, word embeddings are used in a wide range of NLP tasks such as



(a) Without length-normalization, query and centroid are mapped far from each other in vector space.



(b) With length-normalization, query and centroid get closer.

Fig. 2. The two dimensional PCA projection of embeddings for the document relevant to the query. The query and a centroid are labeled to show the difference in distance between them, prior and after the Length Normalization.

binary classification, question answering, machine translation, *etc*. However, in this paper, we use it for query-document similarity matching by using its unique property of formulating document meaning.

For all experiments in this paper, we have used vectors generated by the Skip-gram model as embeddings to measure the similarity between query and document. However, the proposed ranking model can work with the vectors generated by the CBOW method as well. A comprehensive introduction to CBOW and Skip-gram models is outside the scope of this study, but more details can be found in [4].

B. Document Length Normalization

As discussed in the previous section, the length of the document influences its ranking, as it allows each word in the document to contribute equally rather than their actual relevancy to the query. Therefore, this leads to an unreasonably higher ranking of short documents and lower-ranking of important documents. The large documents are profoundly affected as with extended vocabulary, it generates centroids, which mapped far from a query in the actual vector space. Fig. 2a shows how the query word and centroid is mapped on distance, compared to Fig. 2b where distance is reduced after normalization.

Therefore, to counter this problem, we normalize the length of documents before calculating a centroid as it removes unnecessary terms, which help in reducing bias among documents with different lengths.

We have applied two different normalization techniques discussed in subsequent sections.

1) Cosine Normalization: Cosine Normalization is a widely used method in the vector space model [12]. It addresses the two main problems, higher tf_s and the number of terms in the document at a time. With increased individual term frequency for w_i it increases its $tf * idf$ value, which increases the penalty on the term weights. As can be seen in Equation 1, the Cosine normalization is calculated using square root of each term frequency where, w_i is the $tf * idf$ weight for a term i:

$$\sqrt{{w_1}^2 + {w_2}^2 + {w_3}^2 + ... + {w_i}^2}. \tag{1}$$

Also, if the document has more terms, the number of individual weights in the Cosine factor (t in the above formula) increases, yielding a higher normalization factor.

When classifying documents into various categories, Cosine normalization tends to favor retrieval of short documents but suppresses the long documents [12]. So, to address this problem, Pivoted Normalization is suggested.

2) Pivoted Normalization: The Pivoted normalization scheme is based on the principle that the probability of retrieval of a document is inversely related to the normalization factor used in the term weight estimation for that document. The higher the value of the normalization factor for a document, the lower chance of retrieval for that document. This relationship suggests that to boost the chances of retrieval for documents of a certain length, we should lower the value of the normalization factor for those documents, and vice-versa.

Figure 3 illustrates the basic idea of Pivoted normalization. In which, the point where the retrieval and relevance curves cross each other is called the pivot. The documents on one side of the pivot are generally retrieved with a higher probability than their relevance probability, and the documents on the other side of the pivot are retrieved with a lower probability than their probability of relevance. A more detail explanation can be found in [12]. However, here we include an explanation of the formula for Pivoted Normalization p:

$$p = (1.0 - slope) * pivot + slope * old\_norma, \tag{2}$$

where, old_norma is a normalized vector to its unit length and parameters $slope$ and $pivot$ are selected after cross-validation.

While using pivoted normalization, the new term weight Tw for each term T in the document can be written as:

ISSN 2395-8618

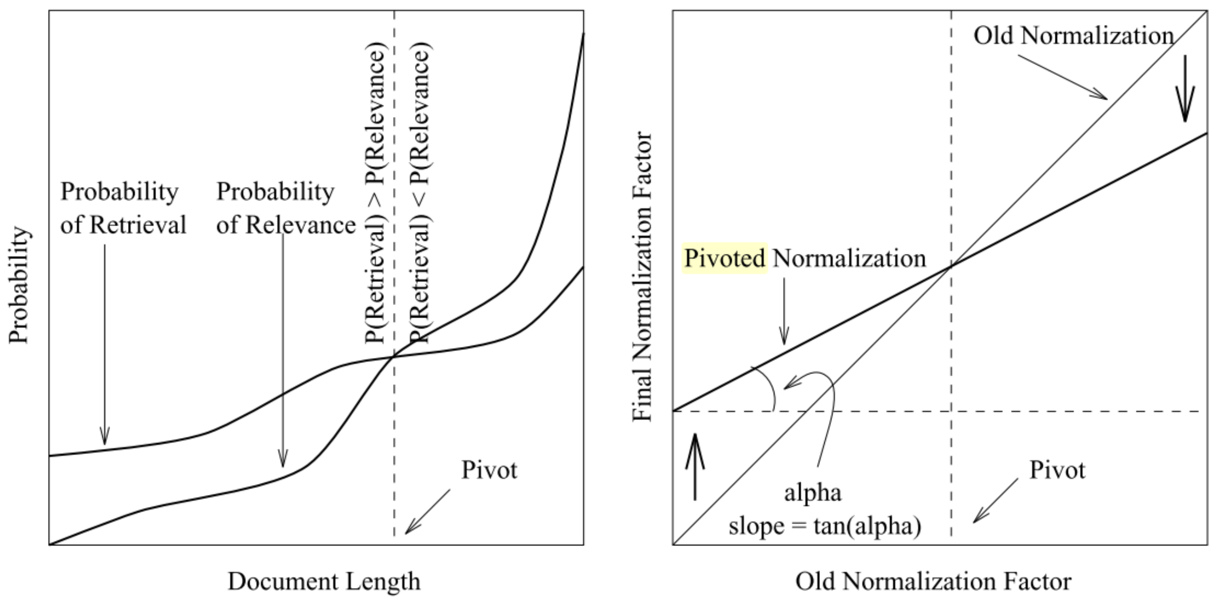


Fig. 3. Pivoted Normalization: The normalization factor for documents for which $P(retrieval) > P(relevance)$ is increased, whereas the normalization factor for documents for which $P(retrieval) < P(relevance)$ is decreased [12].

$$T_w = \frac{tf.idf}{(1.0 - slope) * pivot + slope * old\_norma}. \quad (3)$$

Both normalization techniques discussed above generate a term-frequency score, which signifies the importance of the word in a document. We use this score as a parameter that serves as a threshold, so any word with the value less than a threshold (min_x) is removed from the document before calculating a centroid. In our study, the optimum value for min_x is selected after parameter switching between values 0.01 to 0.1.

In Fig. 2, it can be seen that the cosine distance between a query and the centroid is shorter for normalized document. We are not discussing both normalization techniques comprehensively in this study, but more relevant information can be found in [33], [12].

C. Embedding Space Model

As discussed earlier, differentiating whether a document merely contains a query word or is genuinely relevant to the query topic is a significant challenge. We attempt to address this problem by utilizing word embeddings with a similar approach to [25], wherein a centroid is calculated first for a document, and cosine distance between the query vector and centroid is considered as a similarity measure. However, we also cannot ignore the fact that the presence of a query word in the document is also a piece of strong evidence for relevancy, and embedding space models are weak rankers in the long run [25]. We define a simple ranking function that takes query presence into account while ranking:

$$sim(Q, D) = \overline{L} + cos(\vec{Q}, \vec{D}), \quad (4)$$

where:

$$\overline{L} = \sum_{q_i \epsilon Q} \frac{cos(\vec{Q}, \vec{D}).|q^i|\epsilon D}{|D|}, \quad (5)$$

where:

$$cos(\vec{Q}, \vec{D}) = \frac{1}{|Q|} \sum_{q_i \epsilon Q} \frac{{q_i}^T \overline{D}}{||q_i|| ||\overline{D}||}, \quad (6)$$

where:

$$\overline{D} = \frac{1}{|D|} \sum_{d_i \epsilon \overline{N}(D)} \frac{\vec{d_i}}{||D||}, \ (if\ d_i > min\_x), \quad (7)$$

where, $\overline{N}(D)$ is a length-normalization for document D and min_x is a threshold value.

Here, $cos(\vec{Q}, \vec{D})$ is an absolute relevancy score between query Q and document D calculated using cosine similarity across each query term q_i and centroid $\overline{D}$. The value of $\overline{D}$ is the mean of all word vectors in document D, which serves as single embedding. A function $\overline{N}$ is applied on document D before calculating centroid $\overline{D}$. The $\overline{N}$ is a length-normalization function such as Pivoted or Cosine.

$\overline{L}$ serves as linear ranker which considers a count of query term q^t in document D and multiplies it with the cosine score

ISSN 2395-8618

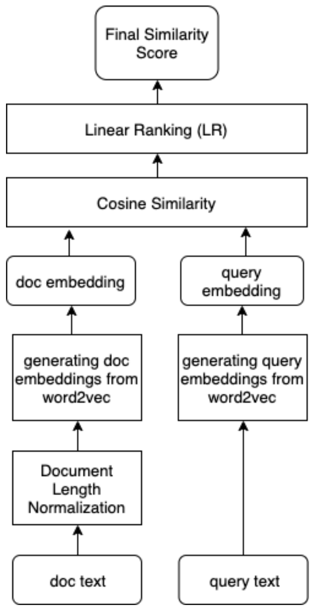

Final Similarity Score
Linear Ranking (LR)
Cosine Similarity
doc embedding
query embedding
generating doc embeddings from word2vec
generating query embeddings from word2vec
Document Length Normalization
doc text
query text

Fig. 4. The architecture of the embedding based model to rank query-document based on length normalization, word embeddings, and linear ranking.

$cos(\vec{Q},\vec{D})$. The reason behind considering a $cos(\vec{Q},\vec{D})$ score in a $\overline{L}$ is to control it from over-ranking documents with multiple query terms but no actual relevancy. In $sim(Q|D)$ linear ranking component is completely ignored if the query count is 0, and in such a case, only the cosine score is considered to decide document rank.

Fig. 4 shows a graphical representation of the proposed model, in which the query and document similarity score is generated by a stack process of length-normalization, inferring embeddings, and ranking. A cosine score is used as a similarity measure, later combined with the linear score as shown in Equation 5. The next section presents the experiments with four different embedding spaces along with Length normalization and Linear Ranking.

IV. Experiments

We compare the performance of normalized ranking function with a baseline based on *word2vec*, which is not utilizing any normalization techniques beforehand. We have also utilized the Dual Embedding Space Model with additional experiments [25]. We have not performed any assessment of our approach over any other term-frequency based baselines, as our primary goal is to study the impact of document length and linear ranking on top of simple embeddings based methods.

In our experiments, we considered every document in the dataset as a candidate for each query, which is aligned with the traditional approach of IR, where the model needs to retrieve a relevant set of documents from a single collection.

A. Datasets

We have used the TREC Blog06 dataset for experiments [13]. The Blog06 dataset is a collection of Blog feeds, Permalinks, and Homepage documents covering a wide area of topics such as news, sports, politics, health, and entertainment. The University of Glasgow collected this dataset for 11 weeks from December 2005 to February 2006. The combined size of collected feeds and permalinks documents is around 3 million. For assessment purposes, a set of 50 queries with a relevant judgment has also provided with the dataset.

B. word2vec Model

We trained a Skip-gram based *word2vec* model with parameter settings of 200 dimensions, 100-word count, windows size of 10, and negative sampling of 5. We used around 500,000 Blog06 feed documents for training. Before training, we performed cleaning by removing unnecessary HTML tags and stop words. We also normalized cleaned text by lower-casing all text content and replacing tokens such as "UK" with "england" and "US/USA" to "united states of america". Moreover, we also stripped out non-English text from documents and eliminated documents with non-English content. There are two variants of the *word2vec* model, Skip-gram and CBOW. However, we train the Skip-gram model only considering the applicability of the proposed approach over the CBOW model also, as both models produce qualitatively and quantitatively similar embeddings.

C. Length Normalization and Linear Ranking

Document Length Normalization is a core component of our ranking function. While measuring the cosine similarity between document centroid and query vector, we first normalized it with Pivoted or Cosine methods. In Pivoted normalization, we run our experiments on multiple combinations of Pivot and Slope value to obtain an absolute threshold. Fig. 5 shows how MAP varying at the various Pivot-Slope combinations. Also, to provide a more subjective baseline, we run the same set of experiments using the Cosine Normalization method.

During the experiment with Pivoted Normalization, the *min_x* parameter was set to 0.05. For Cosine, we used 0.05 in D-OUT + Q-OUT and D-OUT + Q-IN. For D-IN + Q-IN and D-IN + Q-OUT, we have used 0.1.

Experimenting with different *min_x* values makes sense as each embedding space generates different sets of vectors and so its interaction gives different results. Moreover, we have also implemented various *min_x* combinations in the Pivoted Normalization approach, selecting 0.05 as a threshold after parameter switching. We have not assessed a *min_x* parameter comprehensively. However, this parameter is completely dynamic, and its value depends on the vocabulary size of the dataset. Therefore, parameter switching is required to determine the optimal threshold.

ISSN 2395-8618

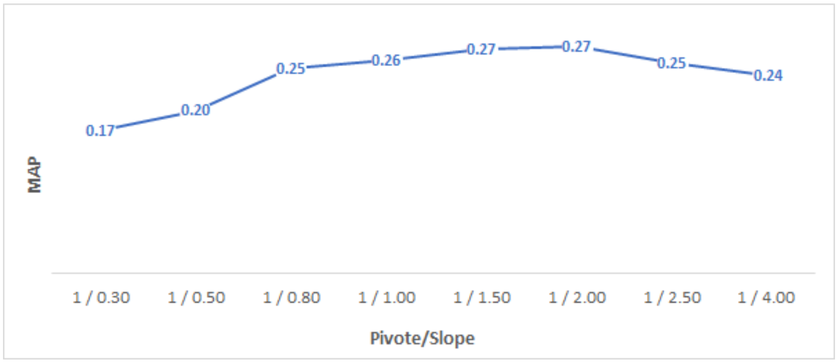


Fig. 5. MAP trend when tweaking Pivote/Slope values. Best results achieved with values between 1/1.50 to 1/2.00.

In our core ranking function, one of the parameters is the count of query words in the document, which is considered strong evidence of relevancy. We conducted experiments, including this parameter set, which improved the overall MAP considerably. Throughout all of our experiments related to Pivoted Length Normalization, including LR, we have used value 1/1.80 for Pivot and Slope.

D. *IN - OUT Embeddings Space*

The CBOW and Skip-gram models of *word2vec* contains two separate embedding spaces (IN and OUT). As shown in Figure 6, these embedding spaces are a set of weights from different layers. Weights for input to hidden layer are called IN embedding, and weights for hidden to output layer are called OUT embeddings [4]. By default, *word2vec* discards OUT embeddings at the end of the training. However, in this study, we keep both weights to utilize them as embedding and infer vectors from cross embedding spaces. Because, relevant few studies have found that words that appear in similar context get pushed closer to each other within IN and OUT embedding space, therefore cosine similarities in IN-IN and OUT-OUT embeddings are higher for words that are typically similar, whereas in IN-OUT cosine similarities are higher for words that often co-occur in training corpus [25].

With the motivation from the above study, we experimented with four different embedding space variants, called IN-IN, IN-OUT, OUT-OUT, and OUT-IN. For example, in the IN-IN approach, vectors for query words are taken from IN embedding, and vectors for the document are extracted from OUT embeddings. Similarly, vectors for query words are taken from IN embeddings and document words from OUT embeddings in the IN-OUT approach.

In general, IN-IN and OUT-OUT embeddings are likely to behave similarly [25]. However, in experiments, we saw further improvement in results with OUT-OUT combination outperforming any other combinations.

V. Results

We performed Length Normalization based evaluation on different combinations of embedding space, as presented in

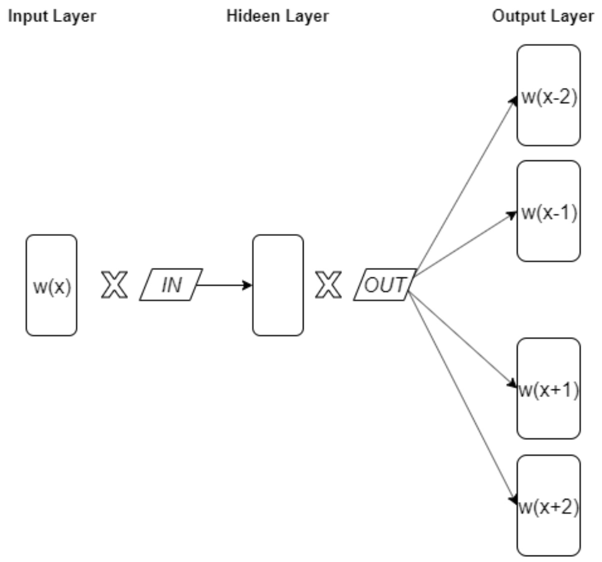


Fig. 6. The architecture of the Skip-gram model. *IN* and *OUT* are the two matrices of weight learned during training and corresponded to the IN and OUT embeddings.

Table I. We ran our experiment using a set of 48 queries out of 50 from the Blog06 dataset, ignoring two queries as an embedding for them was missing. We have significantly improved over-all baselines with Length Normalization, as shown in Table I. Combined approach of LN and LR improved performance significantly in each combination of embedding spaces. The combination of D-OUT + Q-OUT overall performed well with a MAP improving to 0.24 and 0.26 in Cosine and Pivoted Normalization. Interestingly, MAP in each combination of embedding space increases significantly with the LN function. However, the average difference between LN and LR is notably lower.

The Pivoted Normalization outperformed all other methods in terms of MAP. However, while checking its effect on the individual query, we have noticed an unstable trend with different Pivot and Slope combinations, which is shown in Fig. 7. The possible explanation to these dissimilarities is the termination of essential words from the document during

ISSN 2395-8618

TABLE I

MAP results comparing Length Normalization (LN) and Linear Ranking (LR) with baseline. LN, along with LR using D-OUT + Q-OUT, performs significantly better over all baselines. There is also a significant improvement over baseline, even without the LR component.

Embedding Space	Baselines		LN		LN + LR	
	Simple	LR	Cosine	Pivoted	Cosine	Pivoted
D-IN + Q-IN	0.09	0.10	0.16	0.21	0.18	0.23
D-IN + Q-OUT	0.10	0.08	0.19	0.22	0.20	0.22
D-OUT + Q-OUT	**0.20**	**0.21**	**0.24**	**0.26**	**0.30**	**0.31**
D-OUT + Q-IN	0.13	0.13	0.16	0.19	0.21	0.20

Length Normalization. Tweaking parameter min_x can solve this problem to some extent.

Considering a count of query words in the document is also a significant factor. We performed a set of experiments with a linear ranking over embeddings space combined with normalization techniques (Table I). Improvement with LR is not significant in a combination of each embedding space. However, it has performed well with D-OUT and Q-OUT when combined with LN. In Table I with D-OUT & Q-OUT, we achieved a MAP of 0.31 with Pivoted Length Normalization and LR, resulting in an improvement of approx. 47% over aligned baselines.

We have not reported any experiments using term-frequency based methods such as BM25, as our primary goal is to study the possible impact of Length Normalization over embedding based methods. Instead, we used an embedding based method without normalization as our baseline.

VI. Conclusion

In this paper, we investigated the problem of lower-ranking of long documents in embeddings based methods. We presented a *word2vec* based embedding model with Length Normalization of documents and performed experiments with two different normalization techniques, Cosine and Pivoted, and found that Pivoted normalization improves MAP significantly. Based on our results, we also cannot ignore the fact that the presence of the query term in the document is strong evidence of relevancy, and combining it with LN can improve the ranking. To implement this idea, we have also applied LR along with LN. With these combined techniques, we achieved up to 47% improvement over baseline.

However, Length Normalization can cause a change of Average Precision for an individual query when altering Pivot and Slope values. One possible reason for this could be the loss of essential terms after normalization as it considers term-frequency and not a semantic relation.

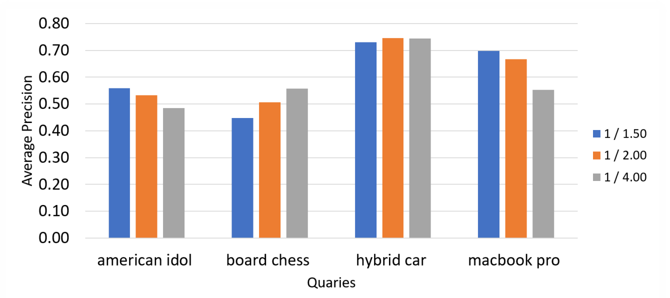


Fig. 7. Inconsistency in AP for individual queries while attempting to normalize documents by tweaking pivot & slope values.

References

[1] P. B. Richard Baeza-Yates and F. Chierichetti, "Essential web pages are easy to find," *Proceedings of the 24th International Conference on World Wide We*, vol. Pages 97-107, 2015.

[2] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, Apr 2009.

[3] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia, Jun. 2013, pp. 746–751. [Online]. Available: https://www.aclweb.org/anthology/N13-1090

[4] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13, USA, 2013, pp. 3111–3119. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999792.2999959

[5] Y. Qi, D. Sachan, M. Felix, S. Padmanabhan, and G. Neubig, "When and why are pre-trained word embeddings useful for neural machine translation?" *Association for Computational Linguistics*, pp. 529–535, Jun. 2018. [Online]. Available: https://www.aclweb.org/anthology/N18-2084

[6] Y. Shen, W. Rong, N. Jiang, B. Peng, J. Tang, and Z. Xiong, "Word embedding based correlation model for question/answer matching," *CoRR*, vol. abs/1511.04646, 2015.

[7] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *CoRR*, vol. abs/1405.4053, 2014. [Online]. Available: http://arxiv.org/abs/1405.4053

[8] D. Roy, "Word embedding based approaches for information retrieval," *Seventh BCS-IRSG Symposium on Future Directions in Information Access*, pp. 1–4, 2014.

[9] H. Zamani and W. B. Croft, "Embedding-based query language models," in *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ser. ICTIR '16, New York, NY, USA, 2016, pp. 147–156. [Online]. Available: http://doi.acm.org/10.1145/2970398.2970405

[10] ——, "Estimating embedding vectors for queries," in *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ser. ICTIR '16, New York, NY, USA, 2016, pp. 123–132. [Online]. Available: http://doi.acm.org/10.1145/2970398.2970403

[11] G. Zuccon, B. Koopman, P. Bruza, and L. Azzopardi, "Integrating and evaluating neural word embeddings in information retrieval," in *Proceedings of the 20th Australasian Document Computing Symposium*, ser. ADCS '15, New York, NY, USA, 2015, pp. 12:1–12:8. [Online].

ISSN 2395-8618

Available: http://doi.acm.org/10.1145/2838931.2838936

[12] A. Singhal, G. Salton, M. Mitra, and C. Buckley, "Document length normalization," *Information Processing and Management*, vol. Volume 32, Issue 5, 1996.

[13] I. Ounis, C. Macdonald, M. de Rijke, G. Mishne, and I. Soboroff, "Overview of the TREC 2006 blog track," *Proceedings of the Fifteenth Text Retrieval Conference, TREC 2006*, 01 2006.

[14] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[15] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944937

[16] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08, New York, NY, USA, 2008, pp. 160–167. [Online]. Available: http://doi.acm.org/10.1145/1390156.1390177

[17] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *The Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944966

[18] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[19] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014. [Online]. Available: http://aclweb.org/anthology/D14-1162

[20] J. Devlin, M. C. ands Kenton Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[21] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *CoRR*, vol. abs/1802.05365, 2018. [Online]. Available: http://arxiv.org/abs/1802.05365

[22] S. Clinchant and F. Perronnin, "Aggregating continuous word embeddings for information retrieval," *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pp. 100–109, 2013.

[23] S. Balaneshin-kordan and A. Kotov, "Embedding-based query expansion for weighted sequential dependence retrieval model," in *Proceedings*

[29] F. Diaz, B. Mitra, and N. Craswell, "Query expansion with locally-trained word embeddings," *CoRR*, vol. abs/1605.07891, 2016. [Online]. Available: http://arxiv.org/abs/1605.07891

of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ser. SIGIR '17, New York, NY, USA, 2017, pp. 1213–1216. [Online]. Available: http://doi.acm.org/10.1145/3077136.3080764

[24] D. Ganguly, D. Roy, M. Mitra, and G. J. Jones, "Word embedding based generalized language model for information retrieval," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '15, New York, NY, USA, 2015, pp. 795–798. [Online]. Available: http://doi.acm.org/10.1145/2766462.2767780

[25] B. Mitra, E. T. Nalisnick, N. Craswell, and R. Caruana, "A dual embedding space model for document ranking," *CoRR*, vol. abs/1602.01137, 2016. [Online]. Available: http://arxiv.org/abs/1602.01137

[26] D. Roy, D. Ganguly, M. Mitra, and G. J. F. Jones, "Representing documents and queries as sets of word embedded vectors for information retrieval," *CoRR*, vol. abs/1606.07869, 2016. [Online]. Available: http://arxiv.org/abs/1606.07869

[27] M. K. Huth2 and Christopher, "Evaluating the impact of word embeddings on similarity scoring in practical information retrieval," *Zenodo*, pp. 585–596, 2017.

[28] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, "WMD: From word embeddings to document distances," *Proceedings of The 32nd International Conference on Machine Learning*, vol. 37, pp. 957–966, 2015.

[30] A. Imani, A. Vakili, A. Montazer, and A. Shakery, "Deep neural networks for query expansion using word embeddings," *CoRR*, vol. abs/1811.03514, 2018. [Online]. Available: http://arxiv.org/abs/1811.03514

[31] M. Grbovic, N. Djuric, V. Radosavljevic, and N. Bhamidipati, "Search retargeting using directed query embeddings," in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW '15 Companion, New York, NY, USA, 2015, pp. 37–38. [Online]. Available: http://doi.acm.org/10.1145/2740908.2742774

[32] P.-S. Huang and J. Gao, "Learning deep structured semantic models for web search using clickthrough data," *ACM International Conference on Information and Knowledge Management (CIKM)*, October 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/learning-deep-structured-semantic-models-for-web-search-using-clickthrough-data/

[33] A. W. Gerard Salton and C.S.Yang, "A vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 18(11), pp. 613–620, November 1975.