A Word Embedding based Generalized Language Model for Information Retrieval

Debasis Ganguly
ADAPT Centre, School of Computing
Dublin City University
Dublin, Ireland
dganguly@computing.dcu.ie

Dwaipayan Roy
CVPR Unit
Indian Statistical Institute
Kolkata, India
dwaipayan_r@isical.ac.in

Mandar Mitra
CVPR Unit
Indian Statistical Institute
Kolkata, India
mandar@isical.ac.in

Gareth J.F. Jones
ADAPT Centre School of Computing
Dublin City University
Dublin, Ireland
gjones@computing.dcu.ie

ABSTRACT

Word2vec, a word embedding technique, has gained significant interest among researchers in natural language processing (NLP) in recent years. The embedding of the word vectors helps to identify a list of words that are used in similar contexts with respect to a given word. In this paper, we focus on the use of word embeddings for enhancing retrieval effectiveness. In particular, we construct a generalized language model, where the mutual independence between a pair of words (say t and t') no longer holds. Instead, we make use of the vector embeddings of the words to derive the transformation probabilities between words. Specifically, the event of observing a term t in the query from a document d is modeled by two distinct events, that of generating a different term t', either from the document itself or from the collection, respectively, and then eventually transforming it to the observed query term t. The first event of generating an intermediate term from the document intends to capture how well a term fits contextually within a document, whereas the second one of generating it from the collection aims to address the vocabulary mismatch problem by taking into account other related terms in the collection. Our experiments, conducted on the standard TREC 6-8 ad hoc and Robust tasks, show that our proposed method yields significant improvements over language model (LM) and LDA-smoothed LM baselines.

Categories and Subject Descriptors

H.3.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval—*Retrieval models, Relevance Feedback, Query formulation*

General Terms

Theory, Experimentation

SIGIR'15, Aug 9–13, 2015, Santiago, Chile.

http://dx.doi.org/10.1145/2766462.2767780 .

Keywords

Generalized Language model, Word Embedding, Word2Vec

1. INTRODUCTION

Word embedding as technique for representing the meaning of a word in terms other words, as exemplified by the *Word2vec* approach [7]. The embedding of the word vectors enables the identification of words that are used in similar contexts to a specufic word. a list of words that are used in similar contexts with respect to a given word. While word Embedding has gained significant interest among researchers in natural language processing (NLP) in recent years, there has to date been little exploration of the potential for use of these methods in information retrieval (IR).

This paper explores the use of word embeddings of enhance IR effectiveness. We begin with a brief introduction to word embedding techniques and then motivate how can these be applied in IR.

A brief introduction to word embedding. Word embedding techniques seek to embed representations of words. For example, two vectors $\vec{t}$ and $\vec{t'}$, corresponding to the words t and t', are close in an abstract space of N dimensions if they have similar contexts and vice-versa, (i.e. the contexts in turn having similar words) [4]. Use of a cosine similarity measure on this abstract vector space of embedded words can be used to identify a list of words that are used in similar contexts with respect to a given word. These semantically related words may be used for various natural language processing (NLP) tasks. The general idea is to train moving windows with vector embeddings for the words (rather than training with the more conventional word count vectors), and classify the individual windows [2]. This finds application for examples in applications such as POS tagging, semantic role labeling, named-entity recognition and other tasks. The state-of-the-art word embedding approaches involve training deep neural networks with the help of *negative sampling* [7]. It is reported that this process of negative sampling (commonly known as *word2vec*[1]) produces reliable word embeddings in a very efficient manner [7].

Potential use in IR. We now discuss how word embeddings can potentially be helpful in improving retrieval quality. In the context of IR, *vocabulary mismatch*, i.e. the inherent characteristic of using different but semantically similar terms across documents about

[1]The name *word2vec* comes from the name of the software tool released by Micholov et. al. (`https://code.google.com/p/word2vec/`

the same topic or between a query and its relevant documents, is a difficult problem to solve.

However, the working principle of most standard retrieval models in IR involves an underlying assumption of term independence, e.g. the vector space model (VSM) assumes that the documents are embedded in a mutually orthogonal term space, while probabilistic models, such as the BM25 or the language model (LM) assume that the terms are sampled independently from documents. Standard approaches in IR take into account term association in two ways, one which involves a *global* analysis over the whole collection of documents (i.e. independent of the queries), while the other takes into account *local* co-occurrence information of terms in the top ranked documents retrieved in response to a query. The latter approach corresponds to the relevance feedback step in IR which we do not investigate in this paper. Existing global analysis methods such as the latent semantic indexing (LSA) [3] or latent Dirichlet allocation (LDA) [1] only take into account the co-occurrences between terms at the level of documents instead of considering the context of a term. Since the word embedding techniques that we introduced in the beginning of this section, leverage the information around the local context of each word to derive the embeddings (two words have close vector representations if and only if they are used in similar contexts), we believe that such an approach can potentially improve the global analysis technique of IR leading to better retrieval effectiveness.

The rest of the paper is organized as follows. Section 2 discusses related work. In Section 3, we propose the generalized LM, which is evalaued in Section 4. Finally, Section 5 concludes the paper.

2. RELATED WORK

Latent semantic analysis (LSA) [3] is a global analysis technique in which documents are represented in a term space of reduced dimensionality so as to take into account inter-term dependencies. More recent techniques such as the latent Dirichlet allocation (LDA) represent term dependencies by assuming that each term is generated from a set of latent variables called the *topics* [1]. A major problem of these approaches is that they only consider word co-occurrences at the level of documents to model term associations, which may not always be reliable. In contrast, the word embeddings take into account the local (window-based) context around the terms [7], and thus may lead to better modeling of the term dependencies.

Moreover, most of these global analysis approaches, e.g. LDA, have been applied in IR in an ad-hoc way for re-assigning term weights without explicitly representing the term dependencies as an inherent part of an IR model. For example, an LDA document model (term sampling probabilities marginalized over a set of latent topic variables) is linearly added as a smoothing parameter to the standard LM probability [9], as a result of which the term dependencies are not clearly visible from the model definition. Contrastingly, in this paper, we intend to directly model the term dependencies as a part of an IR model.

3. A GENERALIZED LANGUAGE MODEL

In this section, we propose the generalized language model (GLM) that models term dependencies using the vector embeddings of terms.

3.1 Language Modelling

In LM, for a given query q, documents are returned as a ranked list sorted in decreasing order by the posterior probabilities $P(d|q)$. These posterior probabilities are estimated for each document d during indexing time with the help of the prior probability ($P(q|d)$) according to the Bayes rule [8, 6, 10].

$$P(d|q) = \frac{P(q|d).P(d)}{\sum_{d' \in C} P(q|d').(d')} \propto P(q|d).P(d) = P(d).\prod_{t \in q} P(t|d)$$
$$= \prod_{t \in q} \lambda \hat{P}(t|d) + (1-\lambda)\hat{P}(t|C) = \prod_{t \in q} \lambda \frac{tf(t,d)}{|d|} + (1-\lambda)\frac{cf(t)}{cs} \tag{1}$$

In Equation 1, the set C represents a universe of documents (commonly known as the *collection*), $\hat{P}(t|d)$ and $\hat{P}(t|C)$ denote the maximum likelihood estimated probabilities of generating a query term t from the document d and the collection respectively, using frequency statistics. The probabilities of these two (mutually exclusive) events are denoted by λ and $1-\lambda$ respectively. The notations $tf(t,d)$, $|d|$, $cf(t)$ and cs denote the term frequency of term t in document d, the length of d, collection frequency of the term t and the total collection size respectively.

3.2 Term Transformation Events

As per Equation 1, terms in a query are generated by sampling them independently from either the document or the collection. We propose the following generalization to the model. Instead of assuming that terms are mutually independent during the sampling process, we propose a generative process in which a noisy channel may transform (mutate) a term t' into a term t. More concretely, if a term t is observed in the query corresponding to a document d, according to our model it may have occurred in three possible ways, shown as follows.

- **Direct term sampling**: Standard LM term sampling, i.e. sampling a term t (without transformation) either from the document d or from the collection.
- **Transformation via Document Sampling**: Sampling a term $t'(t' \neq t)$ from d which is then transformed to t by a noisy channel.
- **Transformation via Collection Sampling**: Sampling the term t' from the collection which is then transformed to t by the noisy channel.

Transformation via Document Sampling. Let $P(t,t'|d)$ denote the probability of generating a term t' from a document d and then transforming this term to t in the query.

$$P(t,t'|d) = P(t|t',d)P(t'|d) \tag{2}$$

In Equation 2, $P(t'|d)$ can be estimated by maximum likelihood with the help of the standard term sampling method as shown in Equation 1. For the other part, i.e. transforming t' to t, we make use of the cosine similarities between the two embedded vectors corresponding to t and t' respectively. More precisely, this probability of selecting a term t, given the sampled term t', is proportional to the similarity of t with t'. Note that this similarity is independent of the document d. This is shown in Equation 3, where $sim(t,t')$ is the cosine similarity between the vector representations of t and t' and $\Sigma(d)$ is the sum of the similarity values between all term pairs occurring in document d, which being the normalization constant, can be pre-computed for each document d.

$$P(t|t',d) = \frac{sim(t,t')}{\sum_{t'' \in d} sim(t,t'')} = \frac{sim(t,t')}{\Sigma(d)} \tag{3}$$

Consequently, we can write Equation 2 as

$$P(t,t'|d) = \frac{sim(t',t)}{\Sigma(d)} \frac{tf(t',d)}{|d|} \tag{4}$$

Equation 4 favours those terms t's that are not only tend to co-occur with the query term t within d, but are also semantically related to

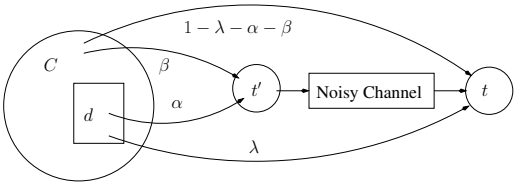


Figure 1: Schematics of generating a query term t in our proposed Generalized Language Model (GLM). GLM degenerates to LM when $\alpha = \beta = 0$.

it. Thus, words that are used in similar contexts with respect to the query term t over the collection, as predicted by the vector embeddings, are more likely to contribute to the term score of t. In other words, Equation 4 takes into account how well an observed query term t *contextually fits* into a document d. A term contextually fits well within a document if it co-occurs with other semantically similar terms. Terms, score high by Equation 4, potentially indicate a more relevant match for the query as compared to other terms with low values for this score.

Transformation via Collection Sampling. Let the complementary event of transforming a term t', sampled from the collection instead of a particular document, to the observed query term t be denoted by $P(t, t'|C)$. This can be estimated as follows.

$$P(t, t'|C) = P(t|t', C).P(t'|C) = P(t|t', C).\frac{cf(t')}{cs} \quad (5)$$

Now $P(t|t', C)$ can be estimated in a way similar to computing $P(t|t', d)$, as shown in Equation 3. However, instead of considering all (t, t') pairs in the vocabulary for computation, it is reasonable to restrict the computation to a small neighbourhood of terms around the query term t, say N_t because taking too large a neighbourhood may lead to noisy term associations. This is shown in Equation 6.

$$P(t|t', C) = \frac{sim(t, t')}{\sum_{t'' \in N_t} sim(t, t'')} = \frac{sim(t, t')}{\Sigma(N_t)} \quad (6)$$

While $P(t, t'|d)$ measures the contextual fitness of a term t in a document d with respect to its neighbouring (in the vector space of embedded terms) terms t' in d, $P(t, t'|C)$, on the other hand, aims to alleviate the vocabulary mismatch between documents and queries in the sense that for each term t in d it expands the document with other related terms t's. From an implementation perspective, $P(t, t'|d)$ reweights existing document terms based on their contextual fit, whereas $P(t, t'|C)$ expands the document with additional terms with appropriate weights.

Combining the Events. Finally, for putting all the events together in the LM generation model, let us assume that the probability of observing a query term t without the transformation process (as in standard LM) be λ. Let us denote the probability of sampling the query term t via a transformation through a term t' sampled from the document d with α, and let and the complementary probability of sampling t' from the collection be then β, as shown schematically in Figure 1. The LM term generation probability in this case can thus be written as shown in Equation 7. This is a generalized version of the standard LM, which we now henceforth refer to as generalized language model (GLM), that takes into account term relatedness with the help of the noisy channel transformation model, which in turn uses the word embeddings to derive the likelihood of term transformations. Note that the GLM degenerates to standard LM by setting α and β to zero, i.e. not using the transformation model in the term generation process.

$$P(t|d) = \lambda P(t|d) + \alpha \sum_{t' \in d} P(t, t'|d)P(t') + \beta \sum_{t' \in N_t} P(t, t'|C)P(t') + (1 - \lambda - \alpha - \beta)P(t|C) \quad (7)$$

Table 1: Dataset Overview

TREC disks	Qry set	Fields	Qry Ids	Avg. qry length	Avg. # rel. docs
4 & 5	TREC 6	title	301-350	2.48	92.22
	TREC 7	title	351-400	2.42	93.48
	TREC 8	title	401-450	2.38	94.56
	Robust	title	601-700	2.88	37.20

3.3 Implementation Outline

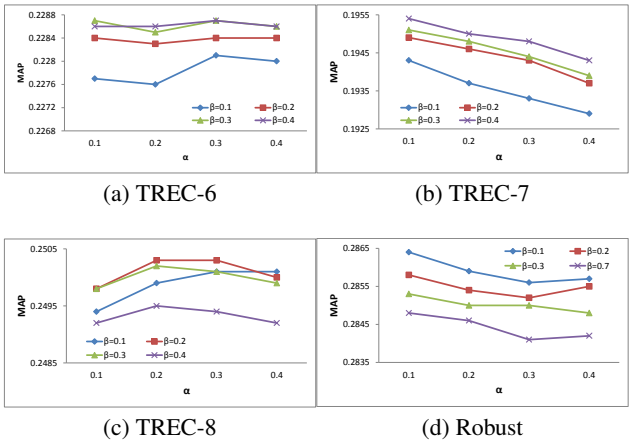
An efficient approach to get the neighbours of a given term is to store a pre-computed list of nearest neighbours in memory for every word in the vocabulary. After this step, for each document d in the collection, we iterate over term pairs (t, t') and assign a new term-weight to the term t representing the document sampling transformation according to Equation 4. Then we iterate again over every term t in d and use the pre-computed nearest neighbours of t (N_t) to compute a score for the collection sampling transformation, as shown in Equation 6. To account for the fact that these transformation probabilities are symmetrical, we add the term t' to d. Note that it is not required to add the term t' in case of the document sampling transformation event because t' is already present in d.

4. EVALUATION

Experimental Setup. Our experiments were conducted on the standard TREC ad hoc tasks from TREC 6, 7, 8 and the Robust track. Information about the document and the query sets is outlined in Table 1. We implemented GLM using the Lucene[2] IR framework. As one of our baseline retrieval models, we used standard LM with Jelinek Mercer smoothing [6, 10], which is distributed as a part of Lucene. Additionally, we also used LM with LDA smoothing [9] as our second baseline to compare against. In contrast to [9], which reports retrieval results with LDA smoothed LM (LDA-LM) on individual document subsets (and their corresponding relevance judgements) from the TREC collection as categorized by their sources, i.e. the "LA Times" and the "Financial Times", we instead executed LDA on the whole TREC collection. The rationale for using LDA as a baseline is that analogous to our model, LDA also attempts to model term dependencies by taking into account latent variables (called the topics). This baseline was also implemented in Lucene.

Parameters. The parameter λ of the LM baseline was empirically set to 0.2 (after varying it within a range of $[0.1, 0.9]$). This value of λ for the TREC collection agrees with the observations reported in [6]. According to the findings of [9], the number of topics in LDA, i.e. K, was set to 800. As prescribed in [5], we set the LDA hyper-parameters α and β (note that these are different from the GLM parameters) to $50/K$ and 0.01 respectively. Obtaining effective word embedding is an integral part of the GLM. The word embeddings for the experiments reported in this section were obtained on the TREC document collection with the parameter settings as prescribed in [7], i.e., we embedded the word vector in a 200 dimensional space, and used continuous bag-of-words

[2] `http://lucene.apache.org/core/`



(a) TREC-6 (b) TREC-7

(c) TREC-8 (d) Robust

Figure 2: Effect of varying the GLM parameters α and β on the MAP values for the TREC query sets.

with negative sampling. The neighbourhood N_t of the GLM (see Equation 7) was set to 3, i.e., for each given term in a document, we consider adding at most 3 related terms from the collection.

Results. First, we varied the GLM parameters, namely α and β within the range $[0.1, 0.4]$ so as to ensure that $\alpha + \beta + \lambda < 1$ (λ being set to 0.2) for all the query sets used in our experiments. The results are shown in Figure 2. It can be seen that the optimal values of α and β depend on the query set, e.g. for the TREC 8 query set (Figure 2c, the optimal results are obtained for $(\alpha, \beta) = (0.3, 0.2)$, whereas this combination does not produce the optimal results for the other query sets. It can be observed that a reasonable choice for these parameters is in the range $[0.2, 0.3]$, which means imparting more or less uniform weights to all the term generation events, namely α, β and λ. In Table 2, we show the optimal results obtained with GLM for each individual query set and compare the results with the baselines, i.e. the LM and the LDA-LM. It can be observed that for each query set, GLM significantly[3] outperforms the baselines. It turns out that the LDA-LM (almost) consistently outperforms the standard LM. However, the results (as measured by the percentage gains in comparison to standard LM) do not seem to be as high as reported in [9] (about 3% as compared to about 8%). We believe that the reason for this is due to the diversity in the LDA topics caused by the news articles from different sources.

From Table 2, we observe that GLM consistently and significantly outperforms both LM and LDA-LM for all the query sets. Not only does it increase the recall values in comparison to LM, but it also increases precision at top ranks by always outperforming LDA in terms of MAP. Although LDA achieves higher recall than GLM in two cases (TREC-6 and Robust), the higher recall in the case of LDA does not significantly increase the MAP, which is indicative of the fact that the precision at top ranks does not improve. For GLM however, an increase in the recall value is always associated with a significant increase in MAP as well, which indicates that precision at top ranks remains relatively stable in comparison to LDA.

5. CONCLUSIONS AND FUTURE WORK

We proposed a generalized version of the language model for IR. Our model considers two possible cases of generating a term from either a document or the collection and then changing it to another term after passing it through a noisy channel. The term transformation probabilities of the noisy channel, in turn, are computed by making use of the distances between the word vectors embedded in an abstract space. We argue that this model has two fold advantage, firstly it is able to estimate how well a term fits in the context of a document, and secondly it is able to decrease the vocabulary gap by adding other useful terms to a document. Empirical evaluation shows that our method significantly outperforms the standard LM and LDA-LM. Possible future work will be to investigate compositionality of terms from the vector embeddings of words.

[3]Measured by Wilcoxon statistical significance test with 95% confidence.

Table 2: Comparative performance of LM, LDA and GLM on the TREC query sets.

Topic Set	Method	Metrics		
		MAP	GMAP	Recall
TREC-6	LM	0.2148	0.0761	0.4778
	LDA-LM	0.2192	0.0790	**0.5333**
	GLM	**0.2287**	**0.0956**	0.5020
TREC-7	LM	0.1771	0.0706	0.4867
	LDA-LM	0.1631	0.0693	0.4854
	GLM	**0.1958**	**0.0867**	**0.5021**
TREC-8	LM	0.2357	0.1316	0.5895
	LDA-LM	0.2428	0.1471	0.5833
	GLM	**0.2503**	**0.1492**	**0.6246**
Robust	LM	0.2555	0.1290	0.7715
	LDA-LM	0.2623	**0.1712**	**0.8005**
	GLM	**0.2864**	0.1656	0.7967

Acknowledgement. This research is supported by SFI through the CNGL Programme (Grant No: 12/CE/I2267) in the ADAPT Centre (www.adaptcentre.ie) at Dublin City University, and by a grant under the SFI ISCA India consortium.

6. REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March 2003.

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, Nov. 2011.

[3] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[4] Y. Goldberg and O. Levy. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722, 2014.

[5] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences (PNAS)*, 101(suppl. 1):5228–5235, 2004.

[6] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, Center of Telematics and Information Technology, AE Enschede, 2000.

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS '13*, pages 3111–3119, 2013.

[8] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR*, pages 275–281. ACM, 1998.

[9] X. Wei and W. B. Croft. LDA-based document models for ad-hoc retrieval. In *SIGIR '06*, pages 178–185, 2006.

[10] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, Apr. 2004.