Exploiting Structured Elements to improve Retrieval

Author 1, Author 2, and Author 3

Institute1
Institute 2
Institute 3
{email1,email2}

Abstract. Word embeddings have recently been shown to be effective in improving retrieval. However, existing word embeddings rely on plain text and do not take into consideration the structure of a web document which is traditionally a popular means of extracting features for query-document matching. The underlying assumption of existing word embeddings from plain text documents is that semantically related words occur together. However, these embeddings may not provide a succinct representation of web documents. Webpages provide a lot more meta-information about each word. We can extract multiple properties of a word such as its color, size or location in a webpage. Integration of these properties into existing word embeddings would yield a much richer representation of web documents which can be used to perform several tasks such as clustering, retrieval or spam detection.
In this preliminary work, we exploit these structural properties to create richer embeddings for web documents. We specifically exploit location of words and structure of web page to create these embeddings. We evaluate the utility of these embeddings with retrieval. Our experiments show that structure embeddings clearly outperform plain text word embeddings on four years of Web track TREC corpus.

Introduction

With exponential growth in digital content , an increasing number of users rely on search engines to find information on a daily basis. In order to assist users in finding the right information with least effort, search engines rely heavily on information retrieval. As opposed to the early days of internet when most web documents primarily comprised of textual content, modern day documents are inherently more complex and composed of heterogeneous structural elements including, but not limited to images, tables or entity panels.

Learning comprehensive representations of data is a critical component of any system that ranks documents. Classical retrieval models such as TF-IDF and BM25, use a bag-of-words representation and cannot effectively capture contextual information of a word. Researchers have also used vector representations of text that embed document and query text into a lower dimensional space using LSA[6], PCA[12] or LDA[1] to compute query-document similarity. Several models are also trained with hand crafted features [3,2,4] that capture query specific syntactic or semantic information from the document. These features may rely on frequency, position or proximity of query terms

in web documents. The above approaches have a major shortcoming: they either require manual decison on several hyperparameters (number of dimensions, topics or vectors) or manually designed features. Such approaches may not succinctly represent and exploit query or document information which may yield suboptimial rankers.

With rapid progress in deep learning and distributional semantics, hand-crafted features are no longer required. Neural IR models can now exploit myriad types of large scale pre-trained vector representations to learn a high-dimensional scoring function for a query and document. Recently proposed neural architectures [24,23,18] rely on word embeddings that directly exploit word co-occurance. However, these embeddings do not directly capture the rich structural information available in web documents. Specifically, existing word embeddings fail to capture insights from the presence of and content embedded in structural elements of web pages which may limit the performance of existing rankers.

Given the significant diversity in the content of web documents, in this preliminary study, we go beyond textual content and aim to leverage various heterogeneous structural elements in a web document to create richer embeddings. From a relevance and user satisfaction perspective, we hypothesize that users often spend less cognitive effort in finding information in structured tables or entity panels than in reading through multiple paragraphs in a webpage. Given the importance of structural elements in webpages and their absence from existing embeddings, we posit that training rankers that jointly exploit distributional semantics and word level meta-data will perform better than traditional models.

In this work, we undertake the task of learning richer document representations and propose a novel neural ranking architecture that exploits both co-occurance and structural information associated with every word in the web page. We begin by investigating document composition of TREC Web documents to understand the distribution of structural elements in web corpus. Specifically, we consider four different types of structural elements: (i) images, (ii) tables, (iii) headings, (iv) lists and (v) hyperlinks to understand the interplay between the presence of such elements and document relevance. We propose a neural architecture which jointly exploits structural element embedding and word embeddings to rank documents.

Given the rich representations of documents, we perform preliminary experiments to demonstrate the efficacy of the proposed model. We observed slight but statistical improvements in NDCG over existing baselines. This indicates the advantage of exploiting underlying structure of web pages in retrieval. We posit that proposed architecture, when augmented with more refined information about a webpage would significantly improve retrieval accuracy. In future, we aim to experiment with recurrent networks that directly exploit word or character level information to reduce proposed model's dependency on pre-trained word embeddings.

Related Work

Our work spans multiple areas of research. We review literature that addresses construction of word embeddings and their utility in information retrieval.

Fig. 1: Percentage distribution of the presence of different structural elements considered.

Distributional Semantics for IR

While many word embedding models have been proposed recently, the Continuous Bag-of-Words (CBOW) and the Skip-Gram (SG) architectures proposed by Mikolov et al. [17] are arguably the most popular. More recently, a theoretical framework was proposed which estimates query embedding vectors based on the individual embedding vectors of vocabulary terms [23]. In terms of applications, word embeddings have been studied in various IR contexts such as term reweighing [24], cross-lingual retrieval [22] and short text similarity [13]. Beyond word co-occurrence, recent studies have also explored learning text embeddings from click-through data [19], session data [9] and for query prefix-suffix pairs [18]. Finally, Diaz *et al.* [7] highlight the value of locally-training word embeddings retrieval. We explore the use of embeddings to represent contributions from different structural elements comprising document and proposing novel ways of leveraging such representations for improved document retrieval.

Document Representations

Vector based representation of documents is a standard practice in retrieval. A document vector simply encapsulates the importance of each term in the document and is usually highly dimensional. As a consequence, these vectors are sparse. To address the data sparsity issue, several dimensionality reduction approaches have been proposed in the literature. A popular class of methods is based on linear projection, which projects a high-dimensional vector on to a lower dimensional space. A historical approach to linear projection is Principal Component Analysis (PCA) [12], which performs a singular value decomposition (SVD) on a document matrix D of size n $\times$ m, where each row in D is the term vector representation of a document. Latent Semantic Analysis (LSA) [6] is very similar to PCA but performs the SVD using the correlation matrix instead of the covariance matrix, which implies a lower computational cost. Other projection models such as Latent Dirichlet Allocation (LDA) [1] are based on the generative models of text in documents. Another approach, named Explicit Semantic Analysis (ESA) [8], represents each document by its similarities to other documents in a collection. Using a low domain specificity document collection such as Wikipedia, the model has proven to obtain competitive results. Deep architectures have been shown to be highly effective in discovering from training data the hidden structures and features at different levels of abstraction useful for a variety of tasks. Liu *et al.* [16] propose a representation Learning technique which makes use of a Multi-Task Deep Neural Networks for semantic retrieval. Recently, Shen *et al.* [19] present a series of new latent semantic models based on a convolutional neural network (CNN) to learn low-dimensional semantic vectors for search queries and Web documents. The interested reader is directed to Li *et al.* [14] which presents a comprehensible survey summarizing different semantic matching techniques.

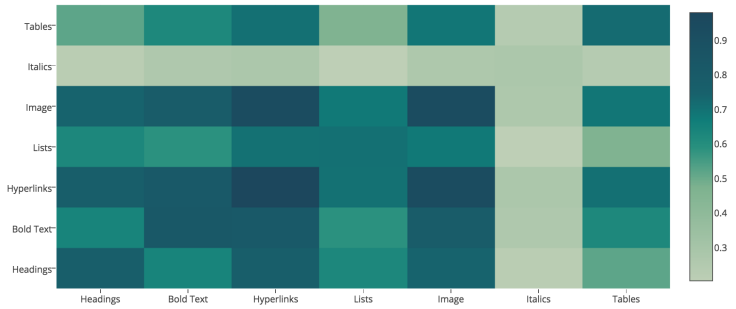


Fig. 2: Element co-occurrence visualized via heatmap.

Exploiting Document Structure

Beyond exact and semantic matching of queries and documents for retrieval, the use of number of different features and elements have been explored by enhanced retrieval techniques. Tombros *et al.* [20] investigate the criteria used by online searchers and presented the relative utility of features: web page content, structure and quality and their importance when assessing the relevance of web pages for information-seeking tasks. Hui *et al.*[10] consider positional information and propose position-aware representations for relevance matching in a neural retrieval framework. Recently [21] found that presence of structural elements in webpage reduce user's overhead in finding required information from a webpage quickly which provides a strong motivation for our work

Methodology

We begin with brief overview of elements used in this work to generate structural embeddings. This is followed by a detailed overview of the model used to jointly score documents and train structure embeddings. We compare the performance of the proposed architecture with some state-of-the-art learning to rank models. Finally, we provide a short description of the datasets and evaluation metrics used in this work.

Structural Elements

Web pages comprise of several structural elements. However, in this work we primarily study role of a small subset of structural elements in a webpage. We consider seven elements listed below.

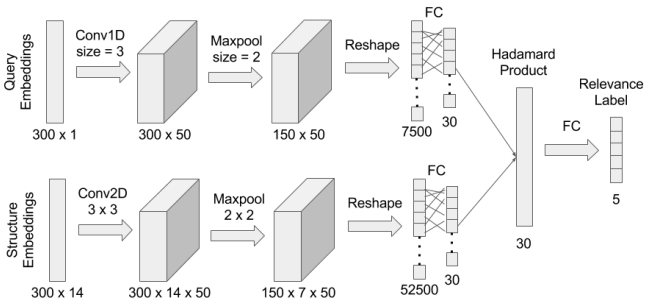
1. **Headings:** Human beings use document or section headings as anchors or entry points [] to paragraphs. Often headings summarize information conveyed by the text in the forthcoming paragraph, and can be treated as a separate element.
2. **Tables:** The presence and contents of any tabular data in the document (table layout) is categorized in the Tables element. Certain type of information e.g. statistics, are easy to find in a table than scanning an entire document. In this work, we consider entire content in a table for learning its low-dimensional representation.

3. **Bold and Italics:** Any italicized or bold text is treated as a separate element. Often certain phrases are marked in bold text to convey their importance. All bold textual content is treated as a separate element.
4. **Hyperlinks:** Any reference to data (within the same document or across different web documents) that the reader can directly follow either by clicking, tapping, or hovering is considered as a hyperlink. We consider the anchor text of all hyperlinks present in a document under this element.
5. **Lists:** Authors often create ordered or unordered lists of content to summarize key points which in turn makes it easier for readers to parse information quickly. All textual content present in such lists is considered under the Lists structural element.
6. **Images:** Information items that are of a non-textual form, i.e., pictures or GIFs or videos comprise the Images entity category. In this work, we use meta-data contained in the image element to create embeddings.

It is important to analyze how these elements are distributed in a web corpus to approximate the quality of embeddings. Sparsely distributed elements that contain less textual content may not yield useful representations. Thus, we first analyze the distribution of above listed elements in Clueweb corpus labeled in TREC Web track 2011-2014. Figure 1 presents the distribution of each of the seven structural elements listed above. We observe that hyperlinks and images are most common with over 90% web documents containing at least one hyperlink or image. Specialized formats such as tables and lists are found in over 70% of web documents.

In terms of formatting of textual content, bold texts and headings are much more commonly found in documents than italicized text, with less than 30% documents containing italicized text. In addition to the popularity of each element (Figure 1), we also look at the element co-occurrence patterns using a heatmap of co-occurring elements in Figure 2. We observe that Images are accompanied more by Hyperlinks in web documents than by Tables. Overall our analysis indicates myriad usage of structural elements in web documents in clueweb corpus. This re-emphasizes that these elements, if exploited, may indeed be useful in retrieval.

Fig. 3: Network architecture



Retrieval with structure embeddings

For preliminary analysis we rely on a simple convolutional neural network architecture that predicts both relevance of a document with respect to a search query and its structural embedding. The network architecture with each layer, input and output size is given in Figure 3. Given Q queries, D documents in corpus and S structural elements in a webpage, the model projects a query and document into an embedding space before scoring.

The proposed network has two input vectors: query and document. We represent each query $q_i \in Q$ query with 300[1] dimensional vector. In this experiment, we represent each document $d_j \in D$ with a stacked two-dimensional matrix, where each row is the embedding of $s_k \in S$ structural element.

Table 1: Query and label distribution of 2011-2014 Web track

Year	#q	#docs	#rel
2011	50	19381	3157
2012	50	16102	3523
2013	50	14531	4149
2014	50	14610	5665

Each query vector is first passed through one-dimensional convolutional layer with 50 filters, kernel size of 3 and stride of 1. However, each document vector is first passed through a two-dimensional convolutional layer with 50 filters, a kernel size of 3X3, and a stride of 1.

While, output of query convolution layer is further passed into a size 2 pooling layer, output of document convolution layer is passed into size 2x2 pooling layer. This is followed by reshaping and a final fully-connected layer that produces a single real-valued vector. All the nodes in the model use the rectified linear unit (relu) function for non-linearity.

We compute the posterior probability of document relevance given a query from the semantic relevance score between them through a softmax function in Equation 1.

$$p(R_{d_j} = r|q_i) = \frac{e^{\beta_r * f(q_i, d_j)}}{\sum_{0<c<K} e^{\beta_c * f(q_i, d_j)}} \tag{1}$$

where $P(R_{d_j} = r|q_i)$ is the probability that relevance grade R_{d_j} of document d_j for query q_i is r. K represents different relevance grades in training data.

Given M query, document pair and corresponding relevance label as training data, we optimize the cross-entropy loss function in Equation 2.

$$\mathcal{L}(x;\theta) = -\frac{1}{n} \sum_{0<n<M} \sum_{0<c<K} [r_j ln(\hat{r}_j) + (1 - r_j) ln(1 - \hat{r}_j)] \tag{2}$$

[1] 300 dimensional embeddings are standard in practice [18,23]

Table 2: $NDCG@1$ and $NDCG@10$ for 2011-2014

	NDCG@1				NDCG@10			
System	2011	2012	2013	2014	2011	2012	2013	2014
SVM_r	**0.25**	**0.16**	0.23	0.17	**0.26**	0.13	0.27	0.23
$LMart$	0.23	0.10	0.22	**0.27**	0.24	**0.16**	0.30	**0.29**
$SEmbed_{all}$	$0.21^{\Downarrow}$	$0.11^{\Downarrow}$	**0.24***	$0.25^{\Uparrow}$	**0.26**	0.15^{*}	**0.32**$^{\Uparrow}$	**0.29**$^{\Uparrow}$
$SEmbed_{head}$	0.05	**0.10**$^{\Downarrow}$	**0.16**$^{\Downarrow}$	$0.13^{\Downarrow}$	$0.08^{\Downarrow}$	**0.14**$^{\Downarrow}$	**0.17**$^{\Downarrow}$	$0.16^{\Downarrow}$
$SEmbed_{table}$	**0.08**	$0.05^{\Downarrow}$	$0.09^{\Downarrow}$	$0.13^{\Downarrow}$	$0.07^{\Downarrow}$	$0.09^{\Downarrow}$	$0.14^{\Downarrow}$	$0.19^{\Downarrow}$
$SEmbed_{link}$	0.06	$0.05^{\Downarrow}$	$0.12^{\Downarrow}$	**0.24**$^{\Uparrow}$	**0.10**$^{\Downarrow}$	$0.10^{\Downarrow}$	$0.15^{\Downarrow}$	**0.21**$^{\Downarrow}$
$SEmbed_{list}$	0.05	$0.05^{\Downarrow}$	$0.10^{\Downarrow}$	$0.14^{\Downarrow}$	$0.09^{\Downarrow}$	$0.11^{\Downarrow}$	$0.15^{\Downarrow}$	$0.17^{\Downarrow}$
$SEmbed_{img}$	0.03	$0.05^{\Downarrow}$	$0.13^{\Downarrow}$	$0.15^{\Downarrow}$	$0.08^{\Downarrow}$	$0.10^{\Downarrow}$	$0.16^{\Downarrow}$	$0.20^{\Downarrow}$

[2]p-val: $^{*} \leq 0.05$, $^{\Uparrow} \leq 0.001$, $^{\Downarrow} \leq 0.001$ over SVMRank with paired t-test

where r_j and $\hat{r}_j$ are actual and predicted relevance labels of a document for a query respectively. K represents different relevance grades in training data.

We generate initial query and document embeddings using glove[3]. We generate embedding for each query by computing the average of its term embeddings. We create two dimensional document embeddings by stacking average term embeddings of the seven structural elements listed above.

The output of fully connected layer just after reshaping layer in Figure 3 would represent low dimensional structure embeddings for each document. Thus, in future, we shall investigate the utility of these embeddings for tasks other than retrieval.

Dataset

We rely on TREC Web collection (2011-2014) [5] for our experiments. Table 1 shows the distribution of relevance labels per year.

We use 3 TREC Web datasets (150 queries)[4] to train and one dataset (50 queries) to test its effectiveness. For example, results for 2014 TREC web queries are obtained by training on 2011-2013 TREC web queries as training data.

Baselines

Several algorithms exist in the literature that optimize for relevance. In recent years, learning-to-rank approaches such as RankSVM [3] and LambdaMart [2] have proved to be effective in optimizing metrics such as NDCG. Thus, in this work we compare the proposed model with both SVMRank and LambdaMart.

LambdaMart [2] (**LMart**) relies on boosted regression trees to optimize non-smooth IR metrics such as NDCG. It has been extensively used in ranking tasks and also won Yahoo! Learning to rank challenge [4] in 2010. SVMRank [3] ($\boldsymbol{SVM_r}$), a pairwise max-margin approach is used to learn a function to rank documents for a query. Given a set of document labels, query and document feature vectors, SVMRank optimizes

[3] https://nlp.stanford.edu/projects/glove/

[4] Further divided into 4:1 ratio for training and validation sets

the objective function learns a hyperplane that enforces ordering among relevant and non-relevant documents.

Above approaches require hand-crafted features. We rely on features used in Letor [15], a popular dataset which consists of several query, document text and structure specific features. For this preliminary work, we computed 68 features to train and test these models.

Evaluation metric

There are several metrics that can be used to evaluate effectiveness of an IR system. In this work we rely on Normalized Discounted Cumulative Gain (NDCG) [11] to evaluate different systems. NDCG score at position n is calculated as follows:

$$NDCG@n = Z_n \sum_{j=1} n \frac{2^{y_j} - 1}{log_2 j + 1} \tag{3}$$

where j is the position in the document list, y_j is the label (relevance or effort) of the document d_j in the list and Z_n is the normalization factor which is the oracle discounted cumulative gain at position n so that a perfect list gets NDCG score of 1.

Results and Discussion

The performance of the proposed model is given in Table 2. We report NDCG@1 and NDCG@10 for preliminary analysis. We also train and compare performance of model trained on each structural element. In this work, we compare models trained on headings ($SEmbed_{head}$), tables ($SEmbed_{table}$), lists ($SEmbed_{list}$), images ($SEmbed_{img}$) and hyperlinks ($SEmbed_{link}$).

The proposed network architecture ($SEmbed_{all}$) performs significantly better than SVMRank trained on letor based features on three datasets (2012-2014). Statistical significance was computed using a paired t-test. These improvements suggest the importance of structural elements in web search. We posit that proposed architecture infers important features for each structure element automatically which is captured manually in features used in SVMRank or LambdaMart. An interesting exploration in the future would be to train and evaluate existing learning-to-rank models with pre-computed embeddings learnt via $SEmbed_{all}$.

We also trained a single network for each structural element to investigate its relative importance in retrieving relevant documents. It is expected that individual structural element will not perform better than model trained on combination of all elements. All structural models ($SEmbed_{head}$ - $SEmbed_{img}$) perform significantly lower than SVMRank baseline. However, their performance on all datasets shows that headings are the most useful amongst all other elements in retrieving relevant documents. On all datasets, $SEmbed_{head}$ performs better other models. This is followed by outlinks ($SEmbed_{link}$) and images ($SEmbed_{img}$) respectively. Tables ($SEmbed_{table}$) and lists ($SEmbed_{list}$), however, are not as useful as we had expected. In future, we shall investigate the differences in embeddings of each element to understand their behaviour and impact on query-document matching in greater depth.

Conclusion

Word embeddings have become a popular means to represent textual information in low-dimensional space. Several models have been proposed recently that exploit these embeddings to train neural architectures that perform significantly better than traditional models that relied on manually coded features. We argue that models based on word embeddings ignore a key aspect of web documents i.e. their structure. Each word in a web page can also be represented using its structural meta-data such as its location, size or style in webpage. Existing deep models do not encode structural information in a webpage in any form which traditionally has been used extensively to design query or document specific features.

In this work, we tried to address the above shortcoming of existing models by proposing a simple neural framework that also encodes structural information of web page text to perform query-document matching. Our network takes query and document vectors (along with structure information) as input to predict document relevance. Preliminary experiments show that our approach performs better than existing learning-to-rank models trained on hand crafted features. We posit that our model automatically learns important textual and structural features to score a document against a search query. In future it would be worth investigating whether structural embeddings would be useful in tasks such as webpage clustering or classification.

References

1. D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
2. C. J. Burges. From ranknet to lambdarank to lambdamart: An overview. Technical report, June 2010.
3. Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. SIGIR '06. ACM.
4. O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. In O. Chapelle, Y. Chang, and T.-Y. Liu, editors, *Proceedings of the Learning to Rank Challenge*, volume 14 of *Proceedings of Machine Learning Research*, pages 1–24, Haifa, Israel, 25 Jun 2011. PMLR.
5. K. Collins-Thompson, C. Macdonald, P. Bennett, F. Diaz, and E. M. Voorhees. Trec 2014 web track overview. Technical report, DTIC Document, 2015.
6. 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*, 41(6):391, 1990.
7. F. Diaz, B. Mitra, and N. Craswell. Query expansion with locally-trained word embeddings.
8. E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJcAI*, volume 7, pages 1606–1611, 2007.
9. M. Grbovic, N. Djuric, V. Radosavljevic, F. Silvestri, and N. Bhamidipati. Context-and content-aware embeddings for query rewriting in sponsored search. In *SIGIR 2015*.
10. K. Hui, A. Yates, K. Berberich, and G. de Melo. Position-aware representations for relevance matching in neural information retrieval. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 799–800. International World Wide Web Conferences Steering Committee, 2017.

11. K. Järvelin and J. Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. SIGIR '00. ACM, 2000.
12. I. T. Jolliffe. Principal component analysis and factor analysis. In *Principal component analysis*, pages 115–128. Springer, 1986.
13. Kenter and de Rijke. Short text similarity with word embeddings. In *CIKM 2015*.
14. H. Li, J. Xu, et al. Semantic matching in search. *Foundations and Trends® in Information Retrieval*, 7(5):343–469, 2014.
15. T. Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *SIGIR '07: Proceedings of the Learning to Rank workshop in the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007.
16. X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-Y. Wang. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *HLT-NAACL*, pages 912–921, 2015.
17. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS 2013*.
18. B. Mitra and N. Craswell. Query auto-completion for rare prefixes. In *CIKM 2015*.
19. Y. Shen, X. He, and J. Gao. Learning semantic representations using convolutional neural networks for web search. In *WWW 2014*.
20. A. Tombros, I. Ruthven, and J. M. Jose. How users assess web pages for information seeking. *Journal of the American society for Information Science and Technology*, 56(4):327–344, 2005.
21. M. Verma, E. Yilmaz, and N. Craswell. On obtaining effort based judgements for information retrieval. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 277–286. ACM, 2016.
22. I. Vulić and M.-F. Moens. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *SIGIR 2015*.
23. H. Zamani and W. B. Croft. Estimating embedding vectors for queries. In *ICTIR 2016*.
24. G. Zheng and J. Callan. Learning to reweight terms with distributed representations. In *SIGIR 2015*.