Information Processing and Management 56 (2019) 733–755

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Neural embedding-based indices for semantic search

Fatemeh Lashkari[a], Ebrahim Bagheri[*,b], Ali A. Ghorbani[a]

[a] Faculty of Computer Science, University of New Brunswick, Canada
[b] Department of Electrical and Computer Engineering, Ryerson University, Canada

ARTICLE INFO

Keywords:
Semantic search
Information retrieval
Inverted index
Neural embeddings

ABSTRACT

Traditional information retrieval techniques that primarily rely on keyword-based linking of the query and document spaces face challenges such as the *vocabulary mismatch problem* where relevant documents to a given query might not be retrieved simply due to the use of different terminology for describing the same concepts. As such, semantic search techniques aim to address such limitations of keyword-based retrieval models by incorporating semantic information from standard knowledge bases such as Freebase and DBpedia. The literature has already shown that while the sole consideration of semantic information might not lead to improved retrieval performance over keyword-based search, their consideration enables the retrieval of a set of relevant documents that cannot be retrieved by keyword-based methods. As such, building indices that store and provide access to semantic information during the retrieval process is important. While the process for building and querying keyword-based indices is quite well understood, the incorporation of semantic information within search indices is still an open challenge. Existing work have proposed to build one unified index encompassing both textual and semantic information or to build separate yet integrated indices for each information type but they face limitations such as increased query process time. In this paper, we propose to use neural embeddings-based representations of term, semantic entity, semantic type and documents within the same embedding space to facilitate the development of a unified search index that would consist of these four information types. We perform experiments on standard and widely used document collections including Clueweb09-B and Robust04 to evaluate our proposed indexing strategy from both *effectiveness* and *efficiency* perspectives. Based on our experiments, we find that when neural embeddings are used to build inverted indices; hence relaxing the requirement to explicitly observe the posting list key in the indexed document: (a) *retrieval efficiency* will increase compared to a standard inverted index, hence reduces the index size and query processing time, and (b) while retrieval efficiency, which is the main objective of an efficient indexing mechanism improves using our proposed method, *retrieval effectiveness* also retains competitive performance compared to the baseline in terms of retrieving a reasonable number of relevant documents from the indexed corpus.

1. Introduction

One of the core tasks of information retrieval is to identify relevant documents to a given input query in realtime. The relevance of a document to the query is often determined based on some statistical measure of relatedness such as the frequency of the query terms in the document (Zou et al., 2017). As such, search indices have been primarily built based on the notion of inverted indices, which

* Corresponding author.
E-mail addresses: fatemeh.lashkari@unb.ca (F. Lashkari), bagheri@ryerson.ca (E. Bagheri), ghorbani@unb.ca (A.A. Ghorbani).

https://doi.org/10.1016/j.ipm.2018.10.015
Received 27 March 2018; Received in revised form 12 September 2018; Accepted 20 October 2018
Available online 11 January 2019

efficiently store and retrieve the relation between terms and documents and their associated statistical measures of relevance according to a bag of words model (Heinz & Zobel, 2003; Persin, Zobel, & Sacks-Davis, 1996). Existing work in the literature (Ensan & Bagheri, 2017) suggest that inverted indices, based on the bag of words model, suffer from two main issues, among others, namely: (1) the consideration of documents as bag of words loses term ordering and inter-term association at the time of indexing; and (2) the lack of semantic description of terms that can lead to incorrect retrieval for ambiguous terms. To address these two challenges, the research community has already explored ways in which semantic information can be incorporated into the retrieval process, often referred to as *semantic search* (Fernandez et al., 2011; Lee, Min, Oh, & Chung, 2014; Ma, Zhang, Liu, Li, & Yuan, 2016; Mangold, 2007). Early and efficient semantic full-text search engines including Mimir (Tablan, Bontcheva, Roberts, & Cunningham, 2015) and Broccoli (Bast, Bäurle, Buchhold, & Haußmann, 2014), which interrelate semantic information and textual content have already been proposed. Most of such systems maintain separate yet interconnected indices that index the semantic and textual content available in documents. For instance, the work in Li, Li, and Yu (2010) proposes to maintain separate indices for semantic entities as well as terms that are observed in the document corpus. In our earlier work (Lashkari, Ensan, Bagheri, & Ghorbani, 2017), we have shown that it is possible to perform semantic full-text search on textual corpora by maintaining three types of indices including (i) textual indices that store term-document associations, e.g., documents D_1 and D_2 contain the term `Armstrong`; (ii) entity indices, which consist of semantic entity-document relationships, e.g., documents D_2 and D_3 consist of the entity `Neil_Armstrong`[1]; and (iii) semantic entity type indices that store entity type hierarchies, e.g., the fact that `Neil_Armstrong` is an entity of type `Astronaut` as expressed by the triple $<$ `dbr:Neil_Armstrong dbp:type dbr:Astronaut` $>$. The integration of these three indices provides the infrastructure to search documents not only based on document-term relevance but also based on term semantics. This approach can provide enhanced retrieval through the integration of the information in the term, entity and type indices; however, its performance is dependent on the strategy used for joining the various types of indices. An obvious limitation of such an approach is its increased query processing time due to the need to query and integrate information from multiple sources.

The main goal of this paper is to explore the possibility of folding these three types of indices into a single index that incorporates textual, semantic entity and entity type information, collectively. Several authors have already explored the possibility of integrating all three types of information into one index and suggest that it might not be efficient to integrate heterogeneous information into the same inverted index as it will significantly increase query processing time because of the need to discern information types at runtime given the heterogeneity of the index keys (Wang et al., 2009). In order to address this issue, we propose a simple yet intuitive idea of turning the heterogeneous keys of such an inverted index into homogeneous ones. In other words, given heterogeneity of index keys is the main cause for the increase in query processing time, if the keys could become homogeneous in some way, then the increase in query processing time would not happen. Our proposed idea for turning the heterogeneous index keys into homogeneous ones is to systematically embed textual, semantic entity and entity type information into the same *embedding space*, which we show to be achievable using *neural embeddings* in this paper. Neural embeddings (Le & Mikolov, 2014; Mikolov, Yih, & Zweig, 2013) attempt to learn unique and dense yet accurate representation of objects based on the contexts they appear in. There have been recent work on ad hoc document (Ensan, Bagheri, Zouaq, & Kouznetsov, 2017; Guo, Fan, Ai, & Croft, 2016) and entity retrieval (Van Gysel, de Rijke, & Kanoulas, 2016; Van Gysel, de Rijke, & Worring, 2015, 2016b), entity linking (Chen et al., 2017; Moreno et al., 2017; Pappu, Blanco, Mehdad, Stent, & Thadani, 2017), and question answering (Bordes, Weston, & Usunier, 2014; Yang, Lee, Park, & Rim, 2015; Zhou, He, Zhao, & Hu, 2015), among others, that employ neural embeddings to improve task performance compared to the state of the art. In our work, we will show how the three types of heterogeneous information can be embedded within the same space in order to make them homogeneous and hence comparable. Once all the information are embedded within the same space, it is possible to index them using a single inverted index; hence, achieving our objective of indexing heterogeneous information without sacrificing query processing time.

The objectives of our work in this paper are, more succinctly, as follows:

1. We are interested in indexing four heterogeneous types of information, namely textual terms, semantic entities, entity types and the documents themselves within one indexing structure. This will ensure that information can be retrieved from the same index regardless of the type of the information. To this end, we will systematically show how textual terms, semantic entities, entity types and the documents that contain these content can be embedded within the same space using neural embeddings and hence become homogeneous to be indexed within a single inverted index.
2. Based on the idea of using neural embeddings to represent textual terms, semantic entities, entity types and documents, it is our objective to redefine the composition of posting lists within an inverted index such that documents are related to posting list keys based on the notion of *similarity* between documents and terms/entities. As such, in our work, we propose a relaxing condition for posting lists where unlike traditional inverted indices, the documents in each posting list are not guaranteed to explicitly contain the index key but are rather guaranteed to be, semantically-speaking, the nearest neighbors of the index key in the embedding space.
3. From a practical perspective, we would like to show the integration of neural embeddings within the process of building inverted indices can have a positive impact on the efficiency and effectiveness of retrieval. For this purpose, we perform systematic evaluation of our work from both intrinsic and extrinsic perspectives. First, intrinsic evaluation is carried out where the effectiveness and efficiency of the proposed method for building an inverted index is compared to a standard information retrieval benchmark. Second, extrinsic evaluation is carried out where the sensitivity of our proposed method is measured with respect to various model parameters.

[1] http://dbpedia.org/resource/Neil_Armstrong .

The practical significance of this work is that it enables the retrieval of semantically relevant documents within the context of inverted indices, which is currently not supported while maintaining the widely known advantages of inverted indices such as being easily compressible and having fast and efficient intersection operations, just to name a few. The rest of the paper is organized as follows. Section 2 presents an overview of inverted index data structures, neural embedding models and approximate nearest neighbor search. In Section 3, we provide the details of our proposed process for building the integrated inverted index. Section 4 will introduce our evaluation methodology, evaluation corpora and our obtained experimental results. We also offer discussions on the effectiveness and efficiency of the proposed approach. Section 5 reviews the related literature on semantic search and neural models for information retrieval. Finally, the paper is concluded in Section 6 with suggestions for further work.

2. Preliminaries

Our proposed work in this paper will be based on several techniques that we will briefly introduce here in order to facilitate understanding. We encourage interested readers to consult the cited references for more in-depth information.

2.1. Inverted indices

The inverted index is a simple yet efficient data structure, which plays an important role in information retrieval systems (Bast & Weber, 2006; Konow, Navarro, Clarke, & López-Ortíz, 2013). Inverted indices store a list of relevant document identifiers along with some associated measures of relevance for each term in the corpus. The information related to each document in the list is referred to as a *posting* and hence the collection of postings related to a given term is known as a *posting list*. Postings in each posting list can be ordered based on increasing document identifier or decreasing term frequency depending on the query processing strategy or index compression technique. When postings are ordered based on some simple document identifier, the index can only support Boolean retrieval, i.e., Boolean intersection and union, also known as *exact match queries* (Lashkari et al., 2017); otherwise, it can also support ranked retrieval (Konow et al., 2013). Boolean retrieval aims to find all the documents where the query terms appear in without caring about their relevance measures. However, ranked retrieval intends to find the most relevant documents. Fig. 1 provides an overview of the structure of an inverted index whose postings store a document identifier, term frequency and term position in the document. Also, they are sorted based on ascending order of document identifiers. For example, the term *africa* appears in three documents identified by 3, 108 and 205 and is located in positions 1, 99 and 467 of the document that is identified by the identifier 205.

2.2. Neural word and paragraph embeddings

In traditional information retrieval techniques, the relation between a term and a document is often defined based on some measure of relevance such as term frequency-inverse document frequency. On this basis many retrieval models focus on building a

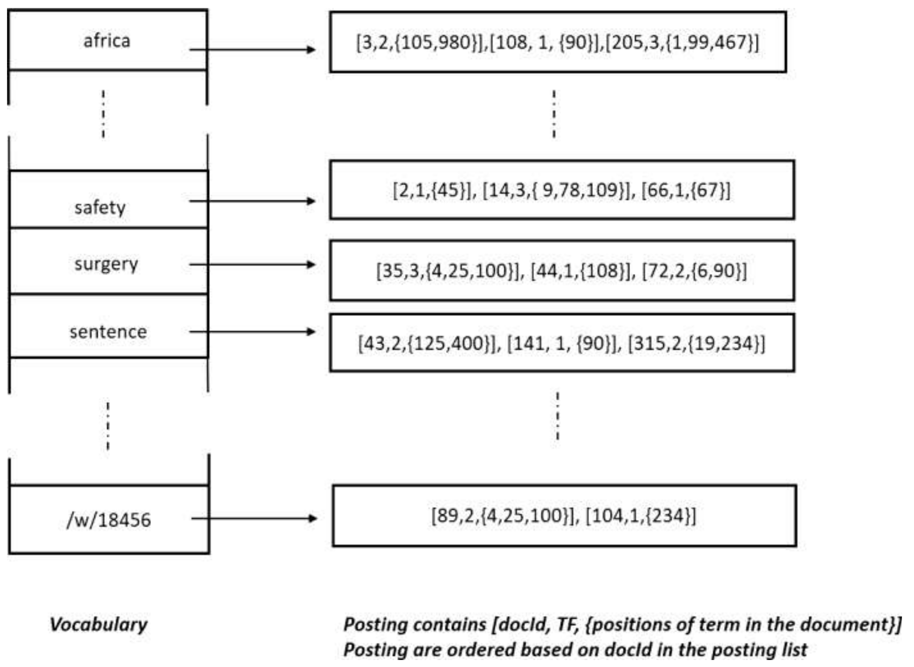

Fig. 1. A sample inverted index whose postings consists of a document identifier in ascending order, term frequency and term position.

vector space based on such measures of relevance. In the vector space, each term/word is represented as a vector whose dimensionality is equal to the vocabulary size. A clear limitation of such an approach is *curse of dimensionality* (Bengio, Ducharme, Vincent, & Janvin, 2003; Indyk & Motwani, 1998) where the large number of dimensions in the feature space cause limitations in building effective classification and retrieval methods. As such, more recent models learn dense yet meaningful vector representations for words in a given corpus. These dense vectors are often known as *word embeddings* (distributed word representations) (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) and if learnt based on a neural network model would be referred to as *neural embeddings*. It has been shown that neural embeddings can maintain syntactic and semantic relations between words (Mikolov, Sutskever et al., 2013). One of the better known methods for learning word embeddings is word2vec that has two variations, namely *skip-gram* (*SG*) and *continuous bag-of-words* (*CBOW*). In both of these variations, word embeddings are learnt by using a three-layer neural network with one hidden layer. Given a sequence of words, a *CBOW* model is trained such that each word can be predicted based on its context, defined as the words surrounding the target word, while *SG* model is trained to predict the surrounding words of the target word. It has been shown in practice that *SG* models have better representation power when the corpus is small while *CBOW* is more suitable for larger datasets and is faster compared to the *SG* model (Mikolov, Sutskever et al., 2013).

While word embeddings are able to efficiently and accurately learn embeddings for document words, it is often desirable to learn vector representations for larger portions of documents such as paragraph. For instance, it would be quite useful to have a vector representation for a paragraph that has appeared in a given document. To this end, Le and Mikolov have introduced the notion of *Paragraph Vectors* (*PV*) (Le & Mikolov, 2014) as an extension to *word2vec* to learn relationships between words and paragraphs. In this model, paragraphs are embedded in the same vector space as words and hence their vectors are comparable. *PV* models can capture similarities between paragraphs by learning the embedding space in such a way that similar paragraphs are placed near each other in the space. Similar to the neural embeddings of words, *PV* models are trained based on completely unlabeled paragraph collections and rely solely on word and paragraph positional closeness. *PV* models can be trained based on two variations. The first method is the *Distributed Memory Model* ($PV-DM$), which assumes each paragraph to be a unique token and uses the word vectors in the paragraph context to learn a vector for the paragraph. The second method is the *Distributed Bag-of-Words* ($PV-DBOW$) model, which is trained by maximizing the likelihood of words that are randomly sampled from the paragraph and ignores the order of words in the paragraph (Le & Mikolov, 2014). The generated output of both paragraph models is an embedding space that consists of dense vector representations of words and paragraphs, which are directly comparable and hence homogeneous in nature. The PV-DM has been empirically shown to have superior performance (Le & Mikolov, 2014; Zwicklbauer, Seifert, & Granitzer, 2016). As explained in subsequent sections, we employ the PV-DM model to learn embeddings for terms, entities and types within the same embedding space as to make them comparable.

2.3. Approximate nearest neighbor search

K-nearest neighbor algorithms are a class of non-parametric methods that are used for text classification and regression (Yang, 1999; Yang & Liu, 1999). They find the nearest points in distance to a given query point and are a simple and generalized form of nearest neighbor search, which also is called similarity search, proximity search, or close item search. Applying nearest neighborhood to high-dimensional feature space deteriorates performance because the distance between the nearest neighbors can be high. Therefore, research has concentrated on finding approximations of the exact nearest neighbors (Aumüller, Bernhardsson, & Faithfull, 2017; Indyk & Motwani, 1998). Finding approximate nearest neighbors has be presented in a variety of algorithms such as using kd-trees (Bentley, 1975) or M-trees (Ciaccia, Patella, & Zezula, 1997) by ending the search early or building graphs from the dataset, where each data point is presented with a vertex and its true nearest neighbors are adjacent to the vertex. Other methods use hashing such as *Locality-sensitive hashing* (*LSH*) (Indyk & Motwani, 1998) to project data points into a lower-dimensional space. In our work, approximate nearest neighbor search is used to find the top k approximate nearest documents to a query term. This retrieval method of documents is based on the assumption that the top k nearest documents in embedding space will be the most related set of documents within the corpus to the query term.

3. Proposed approach

As mentioned earlier, traditional information retrieval models primarily focus on term-based measures to find relevance between query terms and the documents in a collection. With the emergence of knowledge graphs and semantic-based ontologies, it has been shown that retrieval performance can be improved if semantic information is taken into account to determine relevance. Most existing work focus on developing semantics-enabled relevance models for retrieval, which have shown significant improvement over term-based retrieval models. With the positive impact of semantic information in retrieval, it is important to consider actual implementation practicality of these approaches. For instance, the SELM model proposed in Ensan and Bagheri (2017) requires the pairwise calculation of the semantic similarity of entities available in the query and all documents of the collection. This model has shown strong retrieval effectiveness; however, little is known about its practical implementation details and how it can be operationalized. As such, our work in this paper is focused on building efficient indexing infrastructure for facilitating semantic information retrieval.

To this end, earlier works have adopted two strategies to be able to index semantic information. The first group of work have attempted to modify the structure of the inverted index to allow for the indexing of multiple heterogeneous information types, including terms, entities and types (Bast & Buchhold, 2013; Bast, Chitea, Suchanek, & Weber, 2007). The second set of work adopt separate but interrelated indices to store different information types (Cheng & Chang, 2010; Lashkari et al., 2017; Tablan et al., 2015). Both of these works suffer in terms of query processing time where the first approach takes longer to find relevant documents as it has to distinguish between the heterogeneous information stored within the same index while the second approach requires multitude of index calls as well as the integration of the results from multiple indices. The main objective of our work is to develop a single index that can store term, entity and type information without the deficiencies of these already existing approaches.

The core idea of our work is based on three fundamental premises as follows:

1. In order to be able to index information within an inverted index without sacrificing query processing time, the index keys need to be of homogeneous nature. Therefore, we need to develop representations of documents, terms, entities and semantic types, that will form the index keys, such that they are homogeneous and comparable. For this purpose, we propose to use the joint embedding of these four different heterogeneous information types within the same embedding space. The embedded representations of these information will be comparable and hence provide the homogeneity required by inverted index keys.
2. Each posting within the inverted index needs to store one additional measure of relevance for the indexed document to the related index key. The measure of relevance in semantic-based information retrieval is often based on some form of semantic similarity or relatedness (Hasibi, Balog, & Bratsberg, 2016; Liu & Fang, 2015; Sánchez, Batet, Isern, & Valls, 2012). Given the fact that we embed documents within the same embedding space as terms, entities and types, it will be possible to calculate the similarity between each document and any of the three types of information through vector similarity, e.g. cosine similarity and Euclidean distance. This is possible because the documents, terms, entities, and types are embedded within the same space by jointly embedding them.
3. Finally, traditionally within an inverted index, documents listed in the posting list of a given index key are guaranteed to consist of at least one mention of the key. In our work, however, we relax this requirement and do not require that all documents in the posting list have to necessarily contain the index key. Instead, we require that each posting list should include the top-k approximate nearest neighbors of the index key. Based on this requirement, it is possible that a document is listed in the posting list of a certain index key even if the document does not have the index key in it; but is, semantically speaking, more similar to the index key compared to those that actually contain the index key.

Based on these premises, the workflow of our work consists of several steps: In the first step, we jointly learn embeddings for terms, entities, types and documents within the same embedding space. Therefore, the vector representation of all this information is homogeneous and comparable; hence, they can all, if necessary, serve as keys for the same inverted index. For each term, entity and type observed in the document collection, we populate its posting list in the inverted index. In order to populate the posting list, we identify the top k most similar documents that are represented as data points within the joint embedding space and add them to the posting list ordered by their degree of vector similarity. The top k most similar documents are retrieved and identified using approximate nearest neighbor search. This process will result in an inverted index whose posting lists have the most k postings and each posting refers to a document that might not necessarily consist of the key of that entry in the inverted index but is guaranteed to be among the top k most similar documents to the key. We provide the details of this process in the following steps:

3.1. Step 1: Jointly learning the embedding space

In order to jointly learn a vector representation for terms, entities, types and documents, the first step is to identify and link textual documents with knowledge graph entities. In order to achieve this, we perform entity linking (Ferragina & Scaiella, 2010; Meij, Balog, & Odijk, 2014) on each document in the corpus, as a result of which a set of relevant entities for the content of that document are identified and linked to some phrases in the document. We will later explain that we have used Freebase and DBpedia entities in our work. Once entities are identified for each document of the corpus, it is possible to find the type of the entity by traversing the type hierarchy within the knowledge graph. Depending on whether the links in the knowledge graph are traversed towards the children or the parents, super-types and sub-types of the immediate type of the entity can be retrieved. On this basis, we retrieve entity type information for the entities in each document. Now, given the annotated document, it would be possible to use paragraph vector models to jointly learn embeddings based on the whole collection. The main reason why we use paragraph vectors and not word vectors is the need to learn joint embeddings for all the types of information that is present including terms, entities, types and documents. The use of paragraph vectors provides us with vector representations for all these elements in the same embedding space; hence, making them comparable to each other. As such it would be possible to compute the distance between documents, terms, entities and types based on their vector representation without having to be concerned with them being different elements because they are embedded systematically within the same space. However, before this can be done, we need to address the challenge that relates to the paragraph vector context window size.

As mentioned earlier, neural embedding models such as paragraph vector models define the context of a term in the form of a number of terms seen before and after the term of interest. While this will work efficiently when dealing with terms, it will not be directly applicable when additional information that did not originally appear in the document need to be considered. In this specific case, each annotated document now consists of an additional set of entities and their types that were not originally a part of the document and hence would not be included in the embedding process unless they are added to the document. In order to add them to the document, there are two considerations that need to be made: i) the position where the entities and types are added: This is important because the position where the entities and types are added will determine their neighboring terms and hence form their context based on which the vector representations of the entities and types are trained. One approach would be to include the entities and entity types immediately after the phrase that is linked to the entity by the entity linking system. For instance, a document such as “Gensim is a robust open-source vector space modeling and topic modeling” is converted to “Gensim */m/*708*mx* */m/*126*mx* is a robust open-source */m/*1278*q* vector space*/m/*498444*q* */m/*09731*q* */m/*171*mx* modeling and topic modeling */m/*393*mx*” which now includes Freebase identifiers. This leads to the second consideration: ii) once additional entity, and type information are added to the original document, term contexts are now different than they originally were. For instance, for a window size of three, the context for the term “Gensim” would have been “is robust open-source” (assuming articles are ignored), whereas the context of the same term in the revised document would be “is */m/*708*mx* */m/*126*mx*”.

To address these two issues, we generate multiple auxiliary documents for each of the original documents. In each auxiliary document, one of the annotated terms is replaced with its corresponding entity or entity type. For instance, one of the auxiliary documents generated for our earlier example would be “*/m/*708*mx* is a robust open-source vector space modeling and topic modeling” where “Gensim” is replaced by its Freebase identifier. Another alternative auxiliary document would be “ */m/*126*mx* is a robust open-source vector space modeling and topic modeling” where “Gensim” is replaced by its entity type. This way, entities and entity types are incorporated into the documents while respecting the context window size and also preserving the term neighborhood of the original document collection. It should be noted that the inclusion of auxiliary documents does not negatively skew the balance of the term co-occurrences because while the frequency of co-occurrences between terms that appear in the same context increases as the number of auxiliary documents increases, the overall frequency of co-occurrences between all other co-occurring terms also increases. This means that the frequency of all co-occurring terms increases similarly due to the inclusion of additional auxiliary documents and as such while the frequency counts will have larger values, they are proportionally approximately similar to when auxiliary documents were not included.

Now, given the newly developed document collection that includes both the original documents as well as the newly added auxiliary documents that include entities and types, we learn vector representations using the Paragraph Vector model on this document collection. The learned vector representations will include embeddings for terms, entities, types and documents as all of these are present as tokens in the newly created document collection. Essentially based on our new document collection, the embedding model does not distinguish between terms, entities and types as they are all placed in the documents as tokens. Therefore, the paragraph vector model learns vector representations for documents and tokens consisting of terms, entities and types. The learnt vectors for all four types of information are in the same space as required.

For the sake of depicting a few examples of how the terms, entities and types are embedded within the same space, Table 1 provides some sample query terms and their nearest neighbors (including terms, entities and types) in the embedding space. Our general observation from the derived embeddings was that narrow domain terms, e.g., Obama and Poker, tend to have much more semantically similar neighbors compared to more general terms such as Game and Music.

3.2. Step 2: Building semantic inverted indices

One of the limitations of earlier work in building inverted indices in terms of including both term and semantic information is the *heterogeneity* of the index keys. We have been able to address this issue by embedding term, entity, and type information within the same embedding space; therefore, all these three types of information can be used as index keys in the inverted index. As such, it is possible to simply build an inverted index that would consist of one posting list for each term, entity, and type that has been observed in the document collection. According to the traditional method for populating the posting list related to each index key, the posting list consists of one posting per those documents where the index key has been observed at least once. As such, all documents of the posting list are guaranteed to contain at least one mention of the index key. In our work, we relax this requirement and allow documents to be listed in the posting list even if they do not explicitly contain the index key. The primary reason for this is based on the empirical observations of relevance. The relevance judgements provided by human experts in the TREC collection, in some cases, contain relevant documents that do not include the query term that is being searched. For this reason, while the presence of the query term is a strong indicator of relevance, it does not necessarily mean that all the other documents that do not have the query term are irrelevant. There are cases where the document is related to the query terms but it does not contain the query terms explicitly. For instance, for a query such as “famous conspiracies”, a document that talks about the Apollo moon landing would be relevant even if the terms “famous” and “conspiracies” do not appear in the document.

The relaxation of the need to explicitly observe the index key allows us to benefit from the semantics embedded in the vector representation of documents, terms, entities, and types. On this basis, we populate the posting list related to a given index key based on the similarity of the index key with the documents in the document collection.

Several studies have measured distance in the embedding space based on the Euclidean distance (Chen, Lin et al., 2015; Hashimoto, Alvarez-Melis, & Jaakkola, 2016; Kusner, Sun, Kolkin, & Weinberger, 2015; Shwartz & Dagan, 2016). For example, Trieu,

Table 1
Sample query terms and their most similar neighbors in the joint embedding space.

Index key	Similar terms and entities
Obama	President, Mother, News, Iowa, Barack_Obama, African_Americans, American, Family
President	Chairman, Obama, United_States_Capitol, President_of_the_United_States, African_Americans
Game	Atari, Battle, Poker, Computer, Japanese, Anaheim, Korean, PlayStation_2
Poker	Casino, Tournament, Game, Internet, Texas, PlayStation_2
Music	Rock, Band, Song, Interview, California, Uranus

Table 2
Description of the corpora and query sets (topics) used in our experiments.

Collection	Documents	Vocabulary size	TREC topics (Queries)	Max length of queries	Max length of annotated queries
Robust04	528,155	782,799	301–450, 601–700	4	7
ClueWeb09-B-1m	1,073,009	5,910,302	1–200	5	8
ClueWeb09-B-2m	2,186,082	7,791,876	1–200	5	8
ClueWeb09-B-5m	5,006,963	13,666,170	1–200	5	8
Pooled Baselines	249,334	1,870,151	1–200	5	8

Tran, and Tran (2017) proposed a new method for news classification by using pre-trained embeddings based on Twitter content. The authors measure the semantic distance between two embedding vectors using three direct distance metrics L1, Euclidean distance and cosine similarity. Their experiments demonstrated that the semantic distance between two vectors based on Euclidean distance provides the best accuracy. Furthermore, most approximate nearest neighbor search algorithms support Euclidean distance; therefore, in our work, we adopt the inverse of the Euclidean distance between the vector representations of the index key and the document as the *measure of relevance*. For an index key $k = (k_1, \ldots, k_n)$ and a document $d = (d_1, \ldots, d_n)$ that are embedded in the same n-dimensional space, the relevance of d for k, $rel(k, d)$, is calculated as follows:

$$rel(k, d) = (\epsilon + \sqrt{(k_1 - d_1)^2 + \ldots + (k_n - d_n)^2})^{-1}$$

Based on this vector-based measure, the relevance of each document to the index key can be computed. This will range from the most relevant, which would have a relevance of infinity to the least similar which would have a relevance of zero. For each index key, all documents in the corpus can be inversely ordered and included in the relevant posting list. Now, given the fact that a significant number of documents in the corpus are completely unrelated to an index key, it would not be reasonable to include all documents in each of the posting lists. For this reason, the top-k most similar documents can be selected to be included in each posting list.

Based on this strategy, the length of the posting list would depend on the size of the chosen k. In order to find the top-k most similar documents, it is possible to perform *approximate nearest neighbor search* (Indyk & Motwani, 1998) that significantly reduces the computational time of the similarity calculations. In our work, we use locality-sensitive hashing (*LSH*) and random projections (Indyk & Motwani, 1998). The approximate nearest neighbor search for every index key finds and retrieves k documents that are most relevant to the index based on vector similarity. As mentioned earlier, the size of each posting list can be at most k and the postings in each posting list are not guaranteed to contain the index key but are, with an accurate *approximation*, the most similar documents to the index key based on the learnt embeddings.

4. Experimental setup

In this section, we introduce the datasets, gold standards and the implementation details used for running experiments and then subsequently, in the next section, we cover the research questions that were explored for evaluating our work.

4.1. Datasets and gold standards

In our experiments, we benefited from three widely adopted document collections within the information retrieval community: (i) TREC *Robust04*, which is a small news dataset; (ii) *ClueWeb09-B*, is a large Web collections from which we chose 1, 2 and 5 million random documents from among the first 50 million English pages of the corpus; and (iii) Pooled Baselines documents from ClueWeb09-B where the top-100 related retrieval documents are extracted from three widely cited retrieval baselines, namely EQFE (Dalton, Dietz, & Allan, 2014), the RM (Lavrenko & Croft, 2001) and SDM (Metzler & Croft, 2005). We divide the ClueWeb09-B document collection into three document collections, which contain one million, two million and five million documents in order to evaluate the effect of document collection size on efficiency and effectiveness.

We use Freebase annotations of the ClueWeb Corpora (FACC1) as the semantic annotations of the ClueWeb09 documents. We use TagMe (Ferragina & Scaiella, 2010) to perform annotations for TREC Robust04 since there are no public annotations for this collection. In order to do so, we created a locally installed version of TagMe on our local server and ran each document through the service, which produced a set of entity links to Wikipedia entries. This way, the set of Wikipedia entities appearing in each document would be automatically derived. To prune unreliable entities, we set TagMe's confidence value to the recommended value of 0.1. The motivations for choosing the TagMe annotation engine is a study (Cornolti, Ferragina, & Ciaramita, 2013), which shows that TagMe is among the better performing annotation engines for different types of documents, e.g., Web pages and Tweets. Also, TagMe is open source and provides publicly accessible API. The datasets and the used topics (queries) in our experiments are summarized in Table 2. The selected topics needed to also be semantically annotated for which we use TagMe to annotate the related TREC queries for each document corpus. It should also be noted that in our experiments, we removed both stopwords and HTML tags from stemmed queries and stemmed documents. Fig. 2 shows a visualization of the document collections based on the embedding of the terms, entities and documents in the embedding space, developed using the *t-Distributed Stochastic Neighbor Embedding* (t-SNE) technique (Maaten &

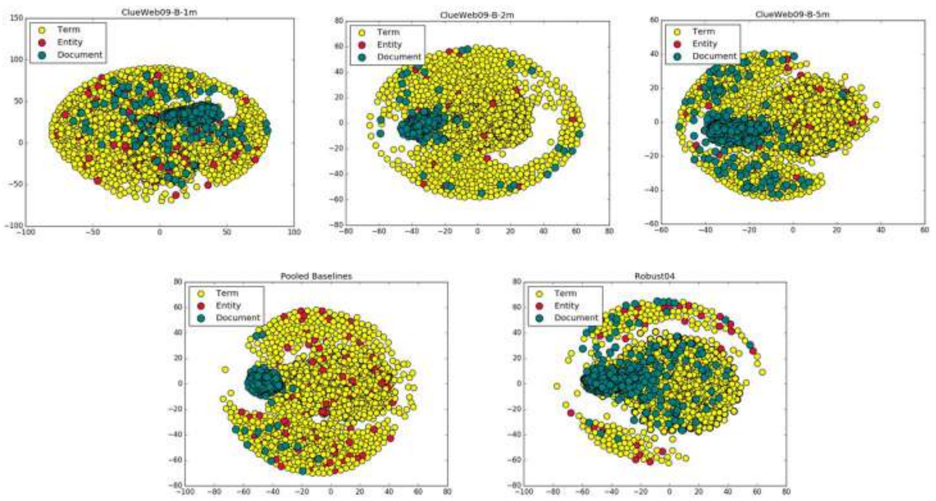

Fig. 2. The joint embedding space of all document collections for terms, entities, types and documents is visualized in scatter plots with t-SNE.

Hinton, 2008).[2]

In terms of the comparative baseline used in our experiments, we used Indri[3] with its default parameter settings. Indri (Strohman, Metzler, Turtle, & Croft, 2005) is a widely adopted information retrieval toolkit developed to simplify evaluation over standard text collections from evaluation forums, e.g., *TREC, CLEF, NTCIR*. For the sake of comparison, we used Indri to index the various text corpora that are listed in Table 2 and also to process the queries listed in the same table. The obtained results for the baseline are based on this installation of Indri.

4.2. Implementation details

Our experiments were performed on a 2x2.7GHz Eight-Core Intel Xeon Processor with 20MB Cache–E5-2680 with 256GB of memory running Ubuntu 16. The paragraph vector models were built using Gensim (Řehůřek & Sojka, 2010), which offers word embeddings and PV model implementation, which follows the model introduced by Le and Mikolov (2014). In the experiments, both sampling techniques, namely Hierarchical Softmax and sampling were considered and are implemented according to Mikolov, Sutskever et al. (2013). To evaluate the effect of the other parameters of the paragraph vector model, we selected three dimensions: 300, 400 and 500 and two context window sizes 5 and 10. Thus, several variation of our proposed index was built based on the above different parameter settings. To distinguish between all the variations, which were built based on different context window sizes and different sampling methods, we defined simple abbreviations to refer to each variation as presented in Table 3. For instance, the abbreviation *W5* refers to the variation of our semantic inverted index whose paragraph vector model has been trained with a context window size of 5 and based on Negative Sampling and likewise *W10-HS* refers to the variation of our work that has a context window size of 10 with Hierarchical Softmax sampling.

For the purpose of performing approximate nearest neighbor search on our learnt embedding space, we employed Annoy,[4] which is a C+ +library with Python bindings. We chose to have 2000 trees in the binary tree forest of Annoy as explained earlier. Also, in order to show the impact of k, three values for k were experimented. We refer to these values of k as k_1, k_2 and k_3, which are equivalent to 0.05%, 0.1%, 0.2% of the number of documents in the document collection, respectively. In addition, the ratio of the average length of posting list for baseline indices to the average length of posting list of the proposed semantic indices is less than 15 for all selected document collections. The posting list size ratio based on k_1, k_2 and k_3 for baseline indices to the proposed semantic indices for all document collections is presented in Table 4. The actual values for k depending on the document collection are presented in Table 5.

5. Empirical evaluations

In this section, we first introduce the research questions and then proceed with the reporting of our findings with regards to each research question.

[2] https://lvdmaaten.github.io/tsne/ .
[3] http://www.lemurproject.org/indri.php .
[4] https://github.com/spotify/annoy .

Table 3
Abbreviations used to refer to the different variation of our work.

Sampling method	Window size	
	5	10
Hierarchical Softmax	W5-HS	W10-HS
Negative Sampling	W5	W10

Table 4
The ratio of the average length of baseline Indri index posting list to the average length of the proposed indexing approach for different values of *k*.

Collection	Posting list size ratio based on:		
	k_1	k_2	k_3
Robust04	1	2	4
ClueWeb09-B-1m	2	4	8
ClueWeb09-B-2m	4	8	15
ClueWeb09-B-5m	2	4	8
Pooled Baselines	4	8	15

Table 5
The length of the posting list depending on the value for *k*.

Collection	Length of posting list		
	k_1 (0.05%)	k_2 (0.1%)	k_3 (0.2%)
Robust04	2700	5400	5000
ClueWeb09-B-1m	5000	10,000	20,000
ClueWeb09-B-2m	10,000	20,000	40,000
ClueWeb09-B-5m	26,000	52,000	104,000
Pooled Baselines	1250	2500	5000

5.1. Research questions

We systematically evaluate our proposed indexing approach based on several research questions (*RQ*) as follows:

*RQ*1. How does the proposed indexing approach compare with the traditional inverted index from an *efficiency* perspective, i.e., space usage and query processing time?

*RQ*2. How does the proposed indexing approach perform when contrasted with the traditional inverted index from an *effectiveness* point of view, i.e., the number of retrieved relevant results?

*RQ*3. How do the parameters of our proposed indexing approach such as embedding space dimensions, context window size and the size of the posting lists, which is dependent on the value for *k* in top k approximate nearest neighbors impact *efficiency* and *effectiveness*?

5.2. RQ 1. Evaluating the efficiency of the proposed approach

Efficiency and Effectiveness are two main desirable characteristics of an information retrieval system (Catena, Macdonald, & Ounis, 2014). In this context, efficiency evaluates the performance of the retrieval systems by considering index storage space (index size) and query processing time (*QPT*). Effectiveness; however, evaluates the quality of the retrieval by looking at how many relevant documents are retrieved for any given query. Striking the right balance between efficiency and effectiveness is important for the usefulness and usability of an information retrieval system.

In the first research question, we evaluate the efficiency of the proposed indexing mechanism. The efficiency of an index is often evaluated based on two measures: the index size and its QPT. Therefore, one can compare the efficiency of two comparable indexing mechanisms by comparing their index size and *QPT* when applied on the same textual corpora and for the same query set. In order to perform comparative analysis of the performance of our proposed index, we indexed the corpora listed in Table 2 both using our proposed indexing method as well as an installation of Indri. The configurations used in *RQs* 1 and 2 are shown in Table 6. For instance, for the Robust04 collection, we used a window size of 5 with Negative Sampling, the length of posting list to be k_3 according to Table 5 and an embedding dimension of 500. We will discuss later how the values in Table 6 are decided in RQ 3. We measured the amount of storage space required for each of the indices (index size) as well as the amount of time required by each of the indices to process the queries related to each collection. The results are reported in Table 7.

Table 6
The configurations of our proposed approach used in *RQs* 1 and 2.

Collection	Configuration		
	Window size	Posting list size	Embedding dimension
Robust04	W5	k_3	D500
ClueWeb09-B-1m	W10	k_3	D500
ClueWeb09-B-2m	W10	k_3	D500
ClueWeb09-B-5m	W10	k_3	D500
Pooled Baselines	W5	k_3	D500

Table 7
Comparative analysis of the efficiency of our proposed indexing strategy compared to Indri.

Collection	Indri		Proposed approach		Efficiency difference	
	Size	QPT	Size	QPT	Δ Size	Δ QPT
Robust04	904.2 MB	39.3s	552.8 MB	26.9s	+38.86%	+31.55%
ClueWeb09-B-1m	3.2GB	1m54.3s	1.05GB	54.7s	+67.18%	+52.14%
ClueWeb09-B-2m	10.7GB	5m20.3s	2.06GB	2m23.6s	+80.74%	+55.16%
ClueWeb09-B-5m	18.9GB	11m33.3s	12.46GB	4m54.5s	+34.07%	+57.52%
Pooled Baselines	1.9GB	45.1s	962.4 MB	32.1s	+49.34%	+28.82%

As seen in the table, our proposed indexing approach has a better efficiency compared to the baseline Indri index in terms of both index size and QPT. The improved efficiency of our approach can be consistently observed across all five corpora and for different query sets; hence, it shows *robust* performance regardless of the corpus size, corpus characteristics and the input query set. The improved performance of our proposed indexing approach is primarily due to two characteristics of our proposed approach: i) the method does not enforce that all documents that have the index key to be present in the related posting list, and ii) the size of the posting lists depends on the size of *k* that is selected in the top k approximate nearest neighbor search technique. We will later show in *RQ* 3 how the size of *k* and other parameters related to learning the embedding model impact efficiency and effectiveness.

Theoretically speaking, there is a clear dependency between index size and QPT whereby smaller indices result in faster response time to queries as there are less information in the index to be processed for each query. This is observable in Table 7 as well. However, it is quite likely that smaller indices do not consist of all the relevant documents and hence cannot retrieve all the possible relevant documents for an input query. Therefore, it is important to explore whether the improvement in index size and QPT provided by our approach translates into poorer/better performance in terms of retrieving relevant documents or not. As such, the next research question (*RQ*2) explores the effectiveness of our proposed approach.

5.3. RQ 2. Evaluating the effectiveness of the proposed approach

We measure effectiveness based on the number of relevant documents retrieved for the queries related to the document corpora. Therefore, in our comparisons with the baseline, we compute how many relevant documents are returned by the baseline compared to the number of documents returned by our proposed index. Within the information retrieval literature, several researchers have proposed that effectiveness can be measured in terms of metrics such as mean average precision (MAP) and normalized discounted cumulative gain (nDCG@k), which primarily focus on whether relevant documents are placed at the top of the retrieved list or not (Zuccon, Koopman, Bruza, & Azzopardi, 2015). Intuitively, a better retrieved list would be the one that consists of relevant documents to the query being placed at the top of the list. Therefore, these metrics are sensitive to not only the relevance of the documents but also their ranking. In retrieval systems, the *indexing mechanism* is responsible for making sure that relevant documents are available to be retrieved while *ranking algorithms*, which work based on the content inside the index, are responsible for putting the content available in the next in the best possible order. For this reason, the best metric to evaluate the effectiveness of an indexing method is the number of relevant documents that it would return for a given query while the best metrics for evaluating a ranking method would be to see how well the retrieved documents by the index are ranked in order. As such, the comparison of our proposed approach, which is an indexing method and not a ranking method, and Indri is based on the number of retrieved relevant documents.

We would like to provide further insight as to why we compare our work with Indri and not with any of the state of the art semantics-based retrieval models such as Ensan and Bagheri (2017), Dalton et al. (2014), Xiong, Callan, and Liu (2017). There are two primary reasons for this: (1) as mentioned earlier, our proposed method is an *indexing mechanism* whose objective to index and maintain the maximum number of relevant documents while the mentioned state of the art techniques are *retrieval methods* that focus on ranking and hence are less focused on maintaining comprehensiveness and are primarily engaged in making sure that the most relevant documents are placed at the top of the retrieved list. Therefore, the objective of our work and these methods is different. (2) The mentioned techniques operate primarily by re-ranking results obtained from a keyword-based retrieval system such as Lavrenko and Croft (2001), Metzler and Croft (2005) and therefore, do not maintain or provide a separate list of relevant documents. As such,

Table 8
The number of relevant documents retrieved by each index.

Collection	Number of relevant documents retrieved by Indri (percentage of relevant documents)	Number of relevant documents retrieved by our approach (percentage of relevant documents)	Effectiveness Difference
Robust04	10,131	13,178	+23.12%
ClueWeb09-B-1m	179	157	−12.29%
ClueWeb09-B-2m	1028	998	−2.91%
ClueWeb09-B-5m	1587	1471	−7.30%
Pooled Baselines	6309	6080	−3.62%

the maximum number of relevant documents retrieved by these methods is equivalent to the number of relevant documents retrieved by the baseline keyword-based techniques already implemented in Indri. For this reason, and given the fact that the focus of our work is maximizing the coverage of relevant documents in the index and not on ranking the relevant documents, we compare our work with Indri and not ranking methods.

The results reported in Table 8 show the number of relevant documents retrieved based on Indri and our proposed approach. For instance, as indicated in the table, Indri is able to retrieve 6309 relevant documents based on the Pooled Baselines collection, while our proposed approach has been able to return 6080 relevant documents. It should be noted that both recall and precision metrics will have comparable performance based on the number of relevant retrieved documents by each method. The reason is that recall is defined as the number of relevant documents retrieved in the context of all relevant documents. The number of relevant documents per query is constant and the same for both methods and hence is a constant value. On the other hand, precision is the number of relevant documents in the list of all retrieved documents. In this case, since our retrieval happens based on top-k most similar documents, the size of the retrieved set is also a constant value. Therefore, the behavior of recall and precision is similar and primarily dependent on the number of relevant retrieved documents.

There are several observations that can be made based on the results in Table 8. The first observation indicates that both approaches retrieve a reasonable number of relevant documents for all five document collections. The second observation is that while both approaches are close in the percentage of relevant documents that they can retrieve, they differ in their performance depending on the collection. For the variations of the ClueWeb9B collection, the Indri index returns a higher number of relevant documents while on the Robust04 collection, our approach provides better results.

We now further compare the performance of our proposed approach with Indri on a per query basis. It is important to see whether the comparative effectiveness results reported in Table 8 are also consistently observed in each query. For this reason, we compare the number of relevant retrieved documents by our approach compared to Indri in Fig. 3 (for ClueWeb09). The y-axis of the diagrams is the difference between the number of relevant documents retrieved by our approach and Indri. Therefore, positive lines in the chart

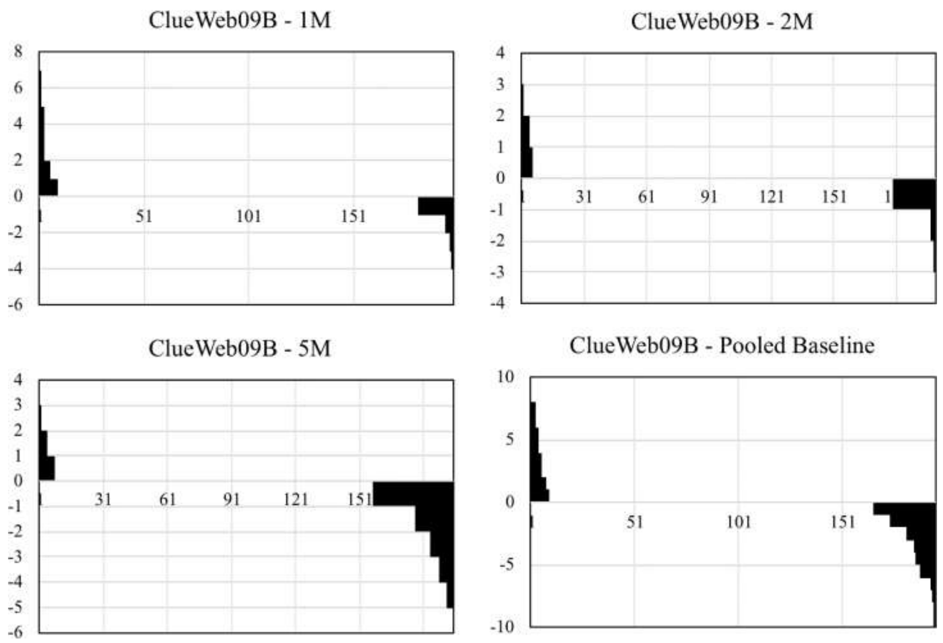

Fig. 3. The comparative performance of the effectiveness of our proposed approach against Indri on a per query basis on ClueWeb09.

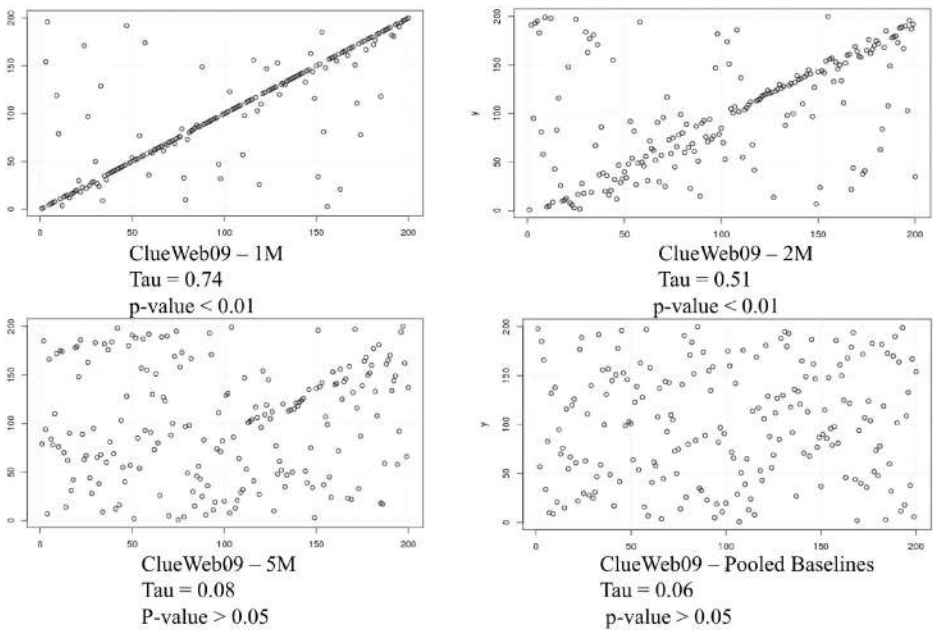

Fig. 4. Kendall's rank correlation between the ranked list of queries based on the number of relevant documents retrieved for our approach compared to Indri.

denote those queries for which our approach retrieved more relevant documents and the negative values are those queries for which Indri retrieves more relevant documents. The queries are sorted in descending order for better visual understanding. The blank queries are those queries that had the same number of relevant documents by both our approach and indri. As seen in the figure, for the majority of the queries, the number of relevant documents retrieved by both approaches are the same and there are only few queries that have slightly different performance.

Now, in order to better understand the behavior of each index, we measure Kendall's rank correlation coefficient based on the queries for each approach sorted by the number of relevant documents retrieved by our approach compared to Indri. The higher the rank correlation is, the more similar the performance of the approaches would be. Fig. 4 visualizes the rank correlations. The figure shows that when the number of relevant documents to queries are low (ClueWeb09 - 1M and 2M) that the performance of our proposed approach and Indri as well as the rank of the queries are quite correlated; however, as more relevant documents become available the correlation of the two approaches drops and the correlation is no longer statistically significant (ClueWeb09 - 5M and Pooled Baselines). This observation needs to be interpreted in the context of the findings of Fig. 3. As seen in Fig. 3, the two approaches have a similar performance on the query level on all three variations of the ClueWeb09 document collection (blank space in the middle of the diagram showing queries that were tied in terms of number of relevant documents retrieved by each of the approaches), the divergence of the rank correlation of queries on ClueWeb09 - 5M and Pooled Baselines means that our approach is retrieving a different set of documents compared to Indri for the queries. In other words, while the two approaches retrieve similar number of relevant documents for each query, the relevant documents that are retrieved are not necessarily overlapping in both approaches and become complimentary as the size of the document collection grows. This is also similarly observed for the Robust04 document collection in Fig. 5.

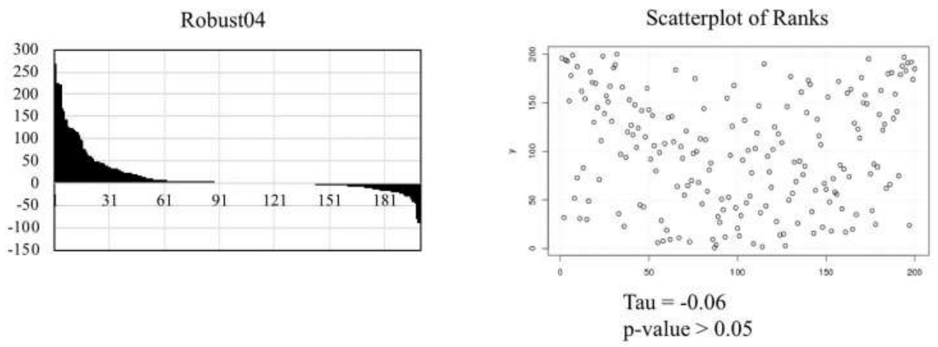

Fig. 5. The comparative performance of the effectiveness (left) and Kendall's rank correlation (right) for our approach compared to Indri on the Robust04 document collection.

Table 9
Kendall's rank correlation between the ranked list of queries based on the number of relevant documents retrieved by our approach compared to when neural embeddings are learnt solely based on terms.

	CW09-B-1m	CW09-B-2m	CW09-B-5m	Pooled Baselines	Robust04
Kendall's Tau	0.539	0.427	0.425	0.019	0.044

We additionally explored whether the retrieval effectiveness exhibited by our proposed approach is due solely to the characteristics of the neural embedding technique or the additional inclusion of type and entity information also played a role. In order to examine this, we used the best configuration obtained in our previous experiments shown in Table 6, to train an embedding model based solely on the textual content of the document collections without the inclusion of type and entity information. The trained embedding model was then used to build the index. Given the index, we retrieved the related documents to each query and then ranked the queries based on the number of relevant documents retrieved. The ranked list of queries was then compared to the ranked list of queries obtained from our proposed approach based on Kendall's rank correlation measure. A highly correlated set of ranked queries would show that our proposed approach and the index based on embeddings trained solely on textual content are similar and as such the inclusion of entity and type information does not play an important role in the process. The findings are reported in Table 9. Based on the correlations reported in this table and compared to the rank correlations observed between our approach and the Indri indices (shown in Figs. 4 and 5), it can be seen that the correlation between our proposed approach and Indri is higher than when compared to the embeddings trained solely based on textual content, indicating that type and entity information included in our approach do in fact play a substantial role in retrieval effectiveness. Furthermore, it is important to point out that not only does the inclusion of the type and entity information impact retrieval effectiveness, but also enables other upstream document ranking models, which work based on entities and types, such as Ensan and Bagheri (2017) and Hasibi et al. (2016), to be built on top of our proposed approach. This is something that is not possible based on Indri indices or embeddings trained solely based on textual content.

We further explore our observations based on Kendall's rank correlation by identifying the hardest and easiest queries for our approach and Indri. We define hard queries for some method (method being our approach or Indri) to be those queries that have the least number of relevant documents retrieved for them by that method. Conversely, we define easy queries to be those that have the most number of relevant retrievals by that method (our approach or Indri). Table 10 (easiest query shown on the top row) and Table 11 (hardest query placed at the top row) show the sorted list of queries for our approach and Indri. As seen in the tables, the two approaches have a similar set of queries identified as easy queries but their hard queries do not have as much overlap. This reinforces our observations based on Kendall's rank correlation and the difference in retrieval effectiveness that while overall the two approaches retrieve similar number of relevant documents but their effectiveness is complementary to each other, showing that our approach can retrieve relevant documents that would otherwise not be retrieved by Indri.

As indicated earlier, the tradeoff between effectiveness and efficiency is also an important consideration in designing information retrieval systems. RQs 1 and 2 explore these two aspects independently and hence it is important to analyze them in tandem. Within the Robust04 dataset, our proposed approach provides improvement in terms of both effectiveness and efficiency. Furthermore, on the Pooled Baselines collection, our approach provides significant improvement in terms of efficiency (index size and QPT) over the baseline; however, it shows a similar performance in terms of effectiveness. It is clear that in such a case, our proposed approach would be favored in a competitive information retrieval systems. Finally, on the ClueWeb09B variations, regardless of the size of the corpus, the Indri index provides better effectiveness while our proposed approach provides better efficiency. The clear tradeoff between effectiveness and efficiency can be seen in the ClueWeb09B collections. We believe that our proposed approach is suitable for cases where: (1) QPT is of significant importance because our work is able to provide at least 50% speedup for the ClueWeb09B collection; and (2) storage space is of importance for storing the index. Given the abundance of memory, this might not seem to be an important consideration; however, when caching or embedded systems considerations are taken into account, a smaller index could be of more help in efficiently retrieving relevant documents. It is important to point out that recent studies (Teevan, 2017) have shown that for two competitive retrieval systems, QPT can be the determining factor for overall user satisfaction even when one of the retrieval systems is providing slightly weaker retrieval effectiveness. As such, the speedup and space utilization provided by our approach can be a strong advantage when considering that its effectiveness is still competitive to the baseline on the ClueWeb09B collection and competitive or better for the Pooled Baselines and Robust04 collections.

5.4. RQ 3. Impact of model parameters on effectiveness and efficiency

The impact of the model parameters can be considered from the perspective of the embedding model variations as well as the impact of the size of the posting lists. We systematically explore the impact of these parameters in the following:

5.4.1. Impact of embedding parameters

The performance of our proposed indexing strategy depends on how well the learnt embedding model can capture the semantics and relationship between terms, entities, types and documents. Research has already shown that the parameters of the embedding space training can impact the quality of the embedding (Zuccon et al., 2015). As such, we have empirically evaluated the impact of these parameters on the performance of our proposed index. There are primarily three parameters within the training process: (1)

Table 10
The list of *easy queries* for our approach compared to Indri. Shared queries between two methods are denoted by **bold face**.

	ClueWeb09 - 1M		ClueWeb09 - 2M		ClueWeb09 - 5M		ClueWeb09 - Pooled		Robust04	
	Ours	Indri	Ours	Indri	Ours	Indri	Ours	Indri	Ours	Indri
1	**dangers of asbestos**	**dangers of asbestos**	**inuyasha**	**inuyasha**	**inuyasha**	**inuyasha**	**fact on uranus**	**fact on uranus**	**Implant Dentistry**	computer viruses
2	**cheap internet**	**voyager**	**obama family tree**	**obama family tree**	**dangers of asbestos**	**dangers of asbestos**	**inuyasha**	**inuyasha**	Radio Waves and Brain Cancer	**encryption equipment export**
3	**voyager**	**cheap internet**	**fact on uranus**	**fact on uranus**	**fact on uranus**	**worm**	**tornadoes**	**tornadoes**	**Income Tax Evasion**	tax evasion indicted
4	**interview thank you**	**cell phones**	**tornadoes**	**tornadoes**	**obama family tree**	**voyager**	**dangers of asbestos**	**dangers of asbestos**	**encryption equipment export**	**Implant Dentistry**
5	**cell phones**	used car parts	**figs**	**figs**	**worm**	**fact on uranus**	**espn sports**	**espn sports**	Poliomyelitis and Post-Polio	food stamps increase
6	**bellevue**	**bellevue**	**tangible personal property tax**	**south africa**	tornadoes	**obama family tree**	**euclid**	**diabetes education**	Hubble Telescope Achievements	**Income Tax Evasion**
7	lower heart rate	**interview thank you**	**south africa**	**tangible personal property tax**	**voyager**	south africa	**diabetes education**	**euclid**	Endangered Species (Mammals)	exotic animals import

Table 11
The list of *hard queries* for our approach compared to Indri. Shared queries between two methods are denoted by **bold face**.

	ClueWeb09 - 1M		ClueWeb09 - 2M		ClueWeb09 - 5M		ClueWeb09 - Pooled		Robust04	
	Ours	Indri	Ours	Indri	Ours	Indri	Ours	Indri	Ours	Indri
1	**rice**	er tv show	**bobcat**	income tax return online	**income tax return online**	the current	**to be or not to be that is the question**	arkadelphia health club	R&D drug prices	Educational Standards
2	**satellite**	the wall	milwaukee journal sentinel	sewing instructions	**to be or not to be that is the question**	defender	**memory**	angular cheilitis	poaching, wildlife preserves	Falkland petroleum exploration
3	rincon puerto rico	raffles	**earn money at home**	kiwi	**earn money at home**	wilson antenna	**earn money at home**	the current	obesity medical treatment	journalist risks
4	iowa food stamp program	titan	**rice**	**bobcat**	rincon puerto rico	avp	rincon puerto rico	defender	food/drug laws	ocean remote sensing
5	rock art	earn money at home	rincon puerto rico	milwaukee journal sentinel	fybromyalgia	**income tax return online**	map	**to be or not to be that is the question**	Great Britain health care	blood-alcohol fatalities
6	credit report	**rice**	getting organized	**earn money at home**	map	**to be or not to be that is the question**	kcs	**memory**	supercritical fluids	drug legalization benefits
7	elliptical trainer	**satellite**	to be or not to be that is the question	**rice**	ps 2 games	**earn money at home**	djs	**earn money at home**	**sick building syndrome**	**sick building syndrome**

Table 12
Impact of sampling strategies on retrieval effectiveness (k_3 and $D500$).

Collection	Number of relevant documents based on the relevance judgement qrel file	Number of relevant documents retrieved (context window size of 5)		Number of relevant documents retrieved (context window size of 10)	
		Negative sampling	Hierarchical Softmax	Negative sampling	Hierarchical Softmax
Robust04	17,412	13,178	13,129	12,647	12,651
ClueWeb09-B-1m	212	156	130	157	148
ClueWeb09-B-2m	1050	990	984	998	989
ClueWeb09-B-5m	1666	1459	1436	1471	1458
Pooled Baselines	6390	6196	6085	6024	5964

sampling strategy; (2) context window size; and (3) embedding dimension. As mentioned earlier in the paper, the results reported in research questions *RQs* 1 and 2 are based on the findings of this section with the best performing trained models.

We first explore whether and how much the sampling strategies impact the performance of the proposed index. There are two main types of sampling strategies namely, Negative Sampling and Hierarchical Softmax. In order to study the impact of the sampling strategy, we systematically studied the various combinations of parameter values for context window size and embedding dimension in combination with the sampling strategies. Due to space constraint, we only report the values for the parameter values k_3, $D500$ and context window sizes of 5 and 10 but note that the other variations show similar behavior. Our observations with regards to the impact of both effectiveness and efficiency regardless of the dimensionality of the embeddings and the context window size was that both sampling strategies show very competitive performance for effectiveness and efficiency. Table 12 summarizes the number of relevant documents that are retrieved based on the proposed approach depending on whether Negative Sampling or Hierarchical Softmax was employed. As seen in the table, the number of relevant documents retrieved by each of the sampling strategies is very close to each other with Negative Sampling having a slight edge over Hierarchical Softmax. On the other hand and for efficiency as shown in Fig. 6, the performance of the proposed indexing strategy is very similar for both Negative Sampling and Hierarchical Softmax. Our conclusion based on the observations made in the experiments is that the sampling strategy does not impact the performance of the indexing mechanism and as such is not an issue of consideration. In the rest of our experiments, we adopted Negative Sampling due to its slightly better performance on retrieval effectiveness.

Furthermore, several researchers (Le & Mikolov, 2014; Zuccon et al., 2015) have already shown that neural embeddings can be sensitive to context window size as it is this parameter that determines which terms, entities and types are considered adjacent in practice and hence would end up having similar vector representations. A larger context window size might result in creating semantic association between terms, entities and types that are not in fact related, while a smaller context window size might miss to make correct associations between related terms available in the corpus. In order to evaluate the impact of context window size, we selected the sampling strategy to be Negative Sampling, as discussed in the previous paragraph, and systematically varied the dimension size between 300, 400 and 500. Based on these settings, we evaluated whether a change in context window size between 5 and 10 impacted retrieval effectiveness and efficiency. Fig. 7 reports on our findings of the impact of context window size on retrieval effectiveness. We find that the characteristics of the document collection influences the optimal size of the context window. In our five document collections, the three variations of the ClueWeb09-B corpus favored a larger context window size while the Robust04 and Pooled Baselines collections were inclined to better retrieval effectiveness with a smaller context window size. This can be explained based on the characteristics of the document collection as mentioned earlier. The difference between the ClueWeb document collections and the Robust04 and Pooled Baselines collections is on *size of the collection*. Our findings indicate that larger context window sizes will show better retrieval effectiveness performance when used on collections that have a large document corpus size; on the contrary, smaller-sized collections will show better effectiveness with a small context window size.

From the perspective of retrieval efficiency, both in terms of index size shown in Fig. 8 and QPT shown in Fig. 9, the impact of context window size is consistent. Larger context window size results in both an increase in index size as well as increased QPT. This finding is expected as a larger context window size will lead to a larger number of similar vectors in the embedding space and

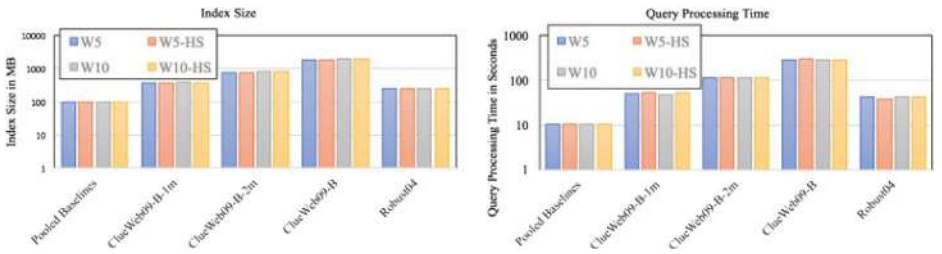

Fig. 6. Impact of sampling strategies on retrieval efficiency.

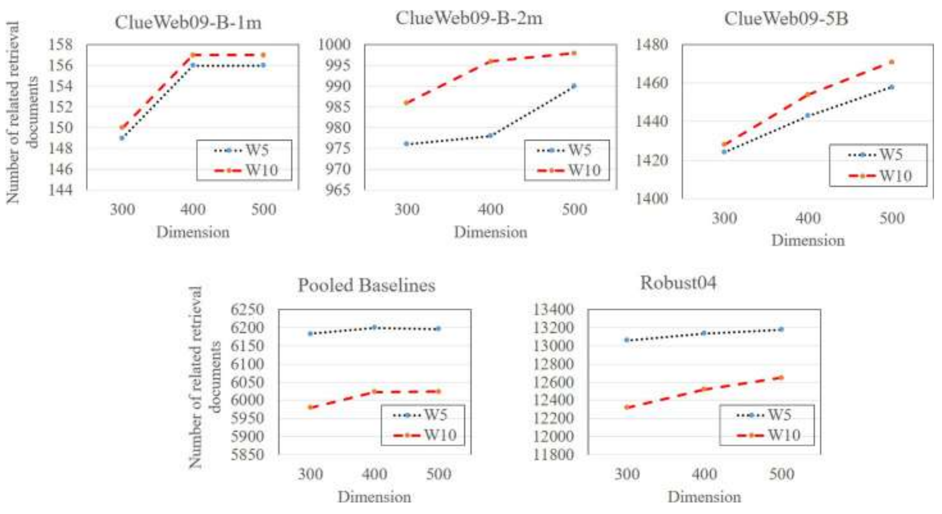

Fig. 7. Impact of context window size and embedding dimension on retrieval effectiveness.

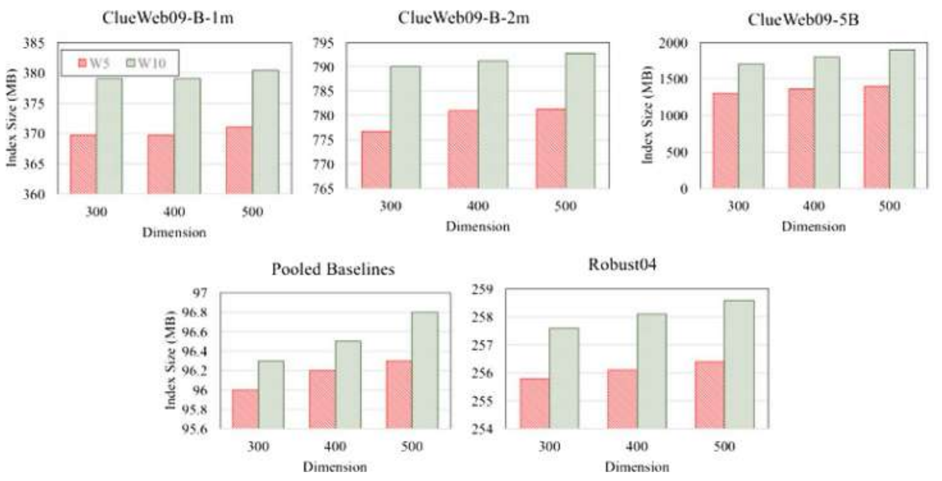

Fig. 8. Impact of context window size and embedding dimension on retrieval efficiency (index size).

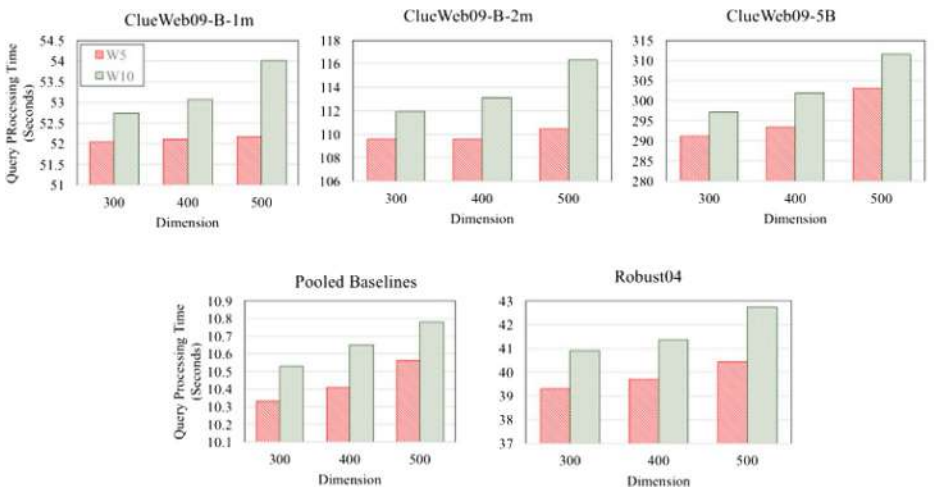

Fig. 9. Impact of context window size and embedding dimension on retrieval efficiency (QPT).

therefore there will be much more similar terms, entities and types per given index key in the proposed index leading to increased index size and QPT. There is clearly a tradeoff between effectiveness and efficiency as shown in the contrast between Fig. 7 and Figs. 8 and 9. Therefore, the choice of the best context window size will depend on the requirements of the retrieval system. In our experiments reported in *RQs* 1 and 2, the choice of the size of the context window selected and reported in Table 6 was motivated by the tradeoff between efficiency and effectiveness and giving higher importance to effectiveness. It should be noted that in a resource-constrained environment or scenarios where extremely fast response time is required, one might prefer to prioritize efficiency over effectiveness.

We also explore the impact of embedding dimensions on both effectiveness and efficiency. The dimension of the vectors in the embedding space is often related to its representation power where a larger size vector has the ability to capture and represent a wider variance of information. With this in mind, if the number of data points that need to be represented in the embedding space are not too high, a larger embedding dimension is not required in practice. Our findings are inline with this theoretical foundation. Fig. 7 shows that for the larger document collections, namely the 2 million and 5 million ClueWeb corpora, the increase in dimension size leads to improved retrieval effectiveness while in the other collections, an increase in the dimension size does not necessarily lead to meaningful improvements in effectiveness. Similar to context window size, larger embedding dimensions lead to increased index size and QPT; hence reinforcing the importance of deciding on the dimension of the embeddings depending on the requirements of the retrieval system. As mentioned in the previous paragraph, the choices for the embedding dimension reported in Table 6 were motivated by prioritizing effectiveness over efficiency in our work.

In our proposed indexing strategy, the size of the posting lists is determined based on the number of nearest neighbors that are retrieved for the posting list key. As such the number of postings in a posting list can range up to the maximum number of documents in the corpus ranked based on their vector similarity to the posting list key. This is in contrast to the size of the posting lists in inverted indices where the number of postings is determined solely based on the number of documents that include the posting list key. As such, we investigate the impact of the posting list size on retrieval effectiveness and efficiency. To this end, we systematically changed the size of the posting lists based on k_1, k_2 and k_3 as shown in Table 5. The values for k indicate the radius size of the nearest neighborhood search and determines how many most similar data points to the posting list key need to be retrieved and placed in the posting list. In these experiments, the parameter values used for training the embedding model was based on the settings shown in Table 6.

5.4.2. *Impact of neighborhood radius*

Fig. 10shows that the increase in the size of the neighborhood radius positively impacts retrieval effectiveness. This is an expected outcome given the fact that a larger posting list has a higher likelihood of containing relevant documents. In other words, if all available documents in the corpus are included in a posting list, the chances of retrieving all relevant documents would be perfect. However, as shown for the other model parameters, this comes at the cost of retrieval efficiency as shown in Fig. 11. The larger the neighborhood radius is, the larger the index size would become as a result of a larger posting list and the slower the retrieval process will be. The relative importance of retrieval effectiveness versus retrieval efficiency would be an important consideration in determining the best size of neighborhood radius. To clearly visualize the tradeoff, Fig. 12 shows how changing the value of k impacts retrieval effectiveness and efficiency in the Robust04 collection.

As seen in Fig. 12, a smaller size for k (k_1) leads to the retrieval of only 55.8% of the relevant documents while a larger k (k_3), which is four times larger than k1, retrieves 75.4%. The first observation is that the increase in the neighborhood radius does not linearly increase retrieval effectiveness. Second, the increase in the size of the posting list from k_1 to k_3 leads to a decreased retrieval time by four orders of magnitude, which is a considerable slow down on retrieval efficiency considering that retrieval time has significant impact on user satisfaction. There have been some studies that have suggested that users are primarily interested in the retrieved results that are presented to them at the very top of the result list. In such circumstances, adopting a small k such as k_1 might be a reasonable decision as retrieval time would be improved significantly. However, on the other hand in critical domains, e.g., patent search, where retrieving all possible relevant documents is of utmost importance and retrieval time is less important, adopting a larger k such as k_3 would be a better choice.

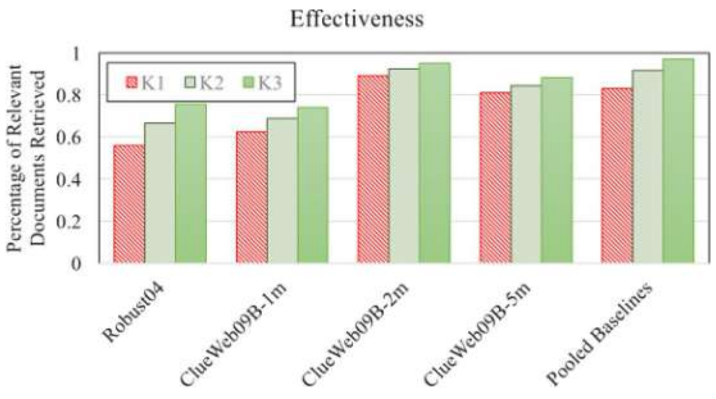

Fig. 10. Impact of nearest neighborhood radius on retrieval effectiveness.

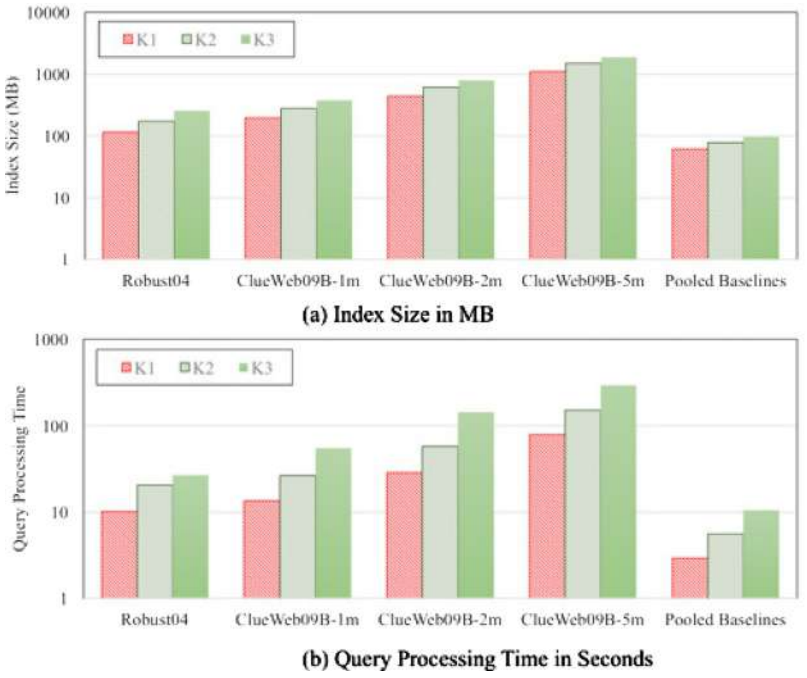

Fig. 11. Impact of nearest neighborhood radius on retrieval efficiency.

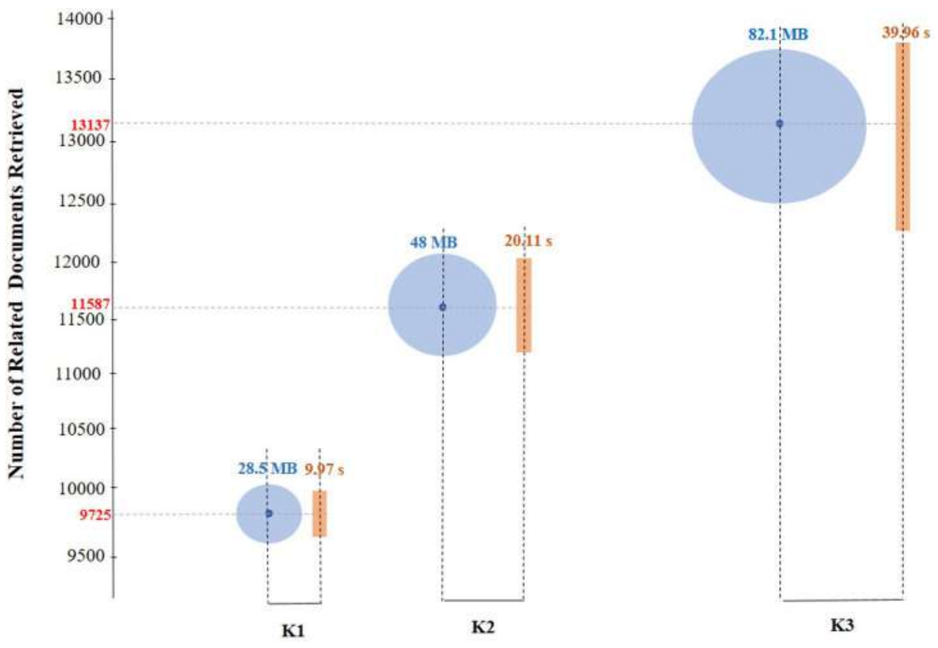

Fig. 12. The tradeoff between effectiveness and efficiency based on different neighborhood radius sizes on Robust04.

5.5. Summary of findings

Our experiments have shown that it is possible to build an effective and efficient index based on the similarity of the vectors jointly learnt for terms, entities, types and documents. We found that there is a tradeoff between retrieval effectiveness and retrieval efficiency and model parameters and corpus characteristics can influence this tradeoff. Our main findings are:

1. Our proposed indexing strategy shows noticeable improvement in terms of retrieval efficiency, i.e., index size and QPT, as shown in Table 7 across all document collections regardless of the characteristics of the document collection;

2. From a retrieval effectiveness perspective, our proposed approach shows competitive or lower performance compared to the baseline inverted index on the ClueWeb'09 document collection while it shows improved performance on the Robust04 document collection as shown in Table 8.
3. We found that the parameters used to train the joint embedding space can influence retrieval effectiveness and efficiency. First, an increased embedding dimension size can improve retrieval effectiveness on large document collections while higher dimension sizes do not necessarily lead to better performance for smaller collections as reported in Fig. 7. Furthermore, as expected, larger embedding dimension sizes negatively impact retrieval efficiency. The length of the context window parameter can influence retrieval effectiveness but has inverse effect depending on the size of the document collection.
4. Finally, our experiments showed that the neighborhood radius size impacts the tradeoff between retrieval effectiveness and efficiency. A larger neighborhood radius will result in a higher number of relevant retrieved documents but at the same increases index size and QPT. The inverse relation between retrieval effectiveness and efficiency is not linear and improved effectiveness could result in 4*x* worst performance on efficiency.

It is important to note that the tradeoff between retrieval efficiency and effectiveness depends on the objectives of the target domain for which the retrieval system is being designed and the characteristics of the document collection. The findings of our experiments assist in choosing the right parameters for an efficient and effective embedding-based index that does not necessarily operate based on explicit term-document association and works according to the similarity of the embedding representation of terms, entities, types and documents.

6. Related work

The information retrieval community has recently become engaged in using neural methods for improving retrieval performance. The objective is to use neural methods as a relevance estimation function to interrelate document and query spaces. Neural models have primarily been used for determining relevance even for cases when query and document collections do not share the same vocabulary set. For example, several researchers have used neural models to estimate relevancy by jointly learning representations for queries and documents (Guo et al., 2016) whereas some other researchers have used neural models for inexact matching by comparing the query with the document directly in the embedding space (Huang et al., 2013; Nalisnick, Mitra, Craswell, & Caruana, 2016) or through the semantic expansion of the query (Diaz, Mitra, & Craswell, 2016; Kuzi, Shtok, & Kurland, 2016; Zamani & Croft, 2016). In this context, neural word embeddings have been aggregated through a variety of approaches, such as averaging the embeddings and non-linear combinations of word vectors (e.g., Fisher Kernel Framework (Clinchant & Perronnin, 2013)) (Le & Mikolov, 2014). The majority of these works are focused on proposing more accurate relevance *ranking* methods and as such differ from our work, which is primarily for indexing relevant documents. The main difference lies in the fact that ranking methods use a set of documents retrieved by a base retrieval method and re-rank them based on some relevance function, while our work serves as the underlying indexing mechanism for maintaining the list of relevant documents that can then be used in ranking methods.

Many information retrieval techniques are based on Language models, which are concerned with modeling the distribution of sequences of words in a corpus or a natural language. Researchers have shown that the probability distribution over sequences of words can be effectively capture through a neural model leading to the so-called *neural language models* (Ganguly, Roy, Mitra, & Jones, 2015). In neural language models, the input words are modeled as vectors whose values are gradually trained using the error back-propagation algorithm in order to maximize the training set log-likelihood of the terms that have been seen together in neighboring sequences. While earlier ideas for building neural language model was based on feed-forward neural network architectures (Bengio et al., 2003), later models focused on recurrent neural networks as the underlying architecture as they provide the means to capture sequentially extended dependencies (Mikolov, Karafiát, Burget, Černockỳ, & Khudanpur, 2010; Mikolov, Kombrink, Burget, Černockỳ, & Khudanpur, 2011). Several authors have also proposed that Long Short-Term Memory-based neural networks would be a suitable representation for learning neural language models as they are robust in learning long term dependencies (Sak, Senior, & Beaufays, 2014; Sundermeyer, Schlüter, & Ney, 2012). From an application perspective, there has been work that explores the possibility of learning neural multimodal language models that can condition text on images and also images on text for bi-directional retrieval. The work by Djuric, Wu, Radosavljevic, Grbovic, and Bhamidipati (2015) introduces a hierarchical neural language model that consists of two neural networks where one is used to model document sequences and one to learn word sequences within documents. This model is relevant to the work in this paper as it dynamically learns embedding representations for both words and documents simultaneously, which relates to the aspect of our work that learn homogeneous representations for terms, entities, types and documents.

Other areas in information retrieval such as entity retrieval that is concerned with finding the most relevant entity from the knowledge graph to an input query (Hu, Huang, Deng, Gao, & Xing, 2015; Jameel, Bouraoui, & Schockaert, 2017) and entity disambiguation that focuses on finding the correct sense of an ambiguous phrase (Fang, Zhang, Wang, Chen, & Li, 2016; Moreno et al., 2017; Yamada, Shindo, Takeda, & Takefuji, 2016a) have used neural representations for more efficient retrieval performance. Other application areas such as query reformulation including both query rewriting (Grbovic, Djuric, Radosavljevic, Silvestri, & Bhamidipati, 2015) and query expansion (Fernández-Reyes, Hermosillo-Valadez, & Montes-y Gómez, 2018), which are used to increase retrieval efficiency, have employed neural representations, and more specifically neural embeddings. One of the main advantages of applying neural models in these contexts is the possibility of overcoming vocabulary mismatch where the embedding representations allow for soft matching between the query and search spaces, be it documents, entities or disambiguation options. This has been the advantage observed in our work in this paper as well where the embedding representation of terms, entities, types and documents go beyond the hard matching of exact terms and entities within documents and similarity/relevance is calculated based on the similarity of the learnt embeddings.

Given our work considers the integration of multiple information types into the same embedding space, it is important to cover related work that have attempted to train joint embedding models as well. One of the earlier works to jointly embed two types of information was the work by Chen, Xu, Liu, Sun, and Luan (2015), which jointly learns embedding representations at character and word levels in the Chinese language. The authors showed this was important due to the compositionality of the structural components of words in the Chinese language. Wang, Zhang, Feng, and Chen (2014) and Yamada, Shindo, Takeda, and Takefuji (2016b) considered embedding words and entities into the same continuous vector space for the sake of named entity disambiguation. The joint embedding model considers both the knowledge base hierarchical structure as well as the word co-occurrence patterns. Toutanova et al. (2015) also learn a similar joint embedding between words and knowledge base entities but instead in the context of the knowledge base completion task. Our work focuses on a similar problem but with attention specifically towards learning a joint embedding representation for terms, entities, types and documents in the context of building semantic inverted indices, which to our knowledge has not been attempted in the past.

7. Concluding remarks

In this paper, we have explored the possibility of building inverted indexes for semantic search engines not based on term occurrence in documents but based on the similarity of the vector representation of terms, entities, types and documents jointly learnt based on neural embeddings. To this end, we have relaxed the main requirement of inverted indices, which require each posting related to a posting list to include the posting list key. Instead, we include postings in the posting list based on the degree of similarity of the document to the posting list key within the embedding space. We employ approximate nearest neighbor search to populate the proposed inverted index structure based on the jointly learnt embedding space. We have evaluated our work based on five publicly available and well accepted document collections and their related standard TREC query sets (topics) and compared its retrieval effectiveness and efficiency with a state of the art retrieval system. Summarily, we find that our approach shows noticeable improvement over the baseline in terms of retrieval efficiency and has better retrieval effectiveness on document collections Robust04 and Pooled Baseline. Furthermore, we systematically explore how different model parameters impact retrieval effectiveness and efficiency and methodically present our findings.

A potential venue for future work will be to improve paragraph vector models by applying vector norms (Ai, Yang, Guo, & Croft, 2016). This strategy has shown to be related to both word frequency and document structures. This way, the effectiveness of our proposed index could be improved by integrating the language model and paragraph models because existing work have shown increased retrieval performance by integrating these two approaches (Ganguly et al., 2015; Zou et al., 2017).

As another area for future work, we plan to explore the effect of combining the presented method for finding top k documents with the Word Movers Distance (WMD) similarity function (Sugawara, Kobayashi, & Iwasaki, 2016) on the effectiveness of our proposed semantic index. WMD is a document-level similarity function which calculates the minimum traveling distance from the matching words of one document to another. Since, this technique was shown to be effective in document classification (Kenter & de Rijke, 2015), it might also have the potential to improve retrieval. Furthermore, Kim, Fiorini, Wilbur, and Lu (2017) have proposed word embeddings based on WMD for calculating similarity between query and documents when no exact matches are found between a query and a document. They showed an increase in mean average precision compared to a BM25 baseline and hence is another sign of the potential usefulness of WMD.

References

Ai, Q., Yang, L., Guo, J., & Croft, W. B. (2016). *Analysis of the paragraph vector model for information retrieval. Proceedings of the 2016 ACM international conference on the theory of information retrievalICTIR '16*New York, NY, USA: ACM133–142. https://doi.org/10.1145/2970398.2970409.

Aumüller, M., Bernhardsson, E., & Faithfull, A. (2017). *Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. Similarity search and applications – 10th international conference, SISAP 2017, Munich, Germany, October 4–6, 2017, proceedings*34–49. https://doi.org/10.1007/978-3-319-68474-1_3.

Bast, H., Bäurle, F., Buchhold, B., & Haußmann, E. (2014). *Semantic full-text search with broccoli. The 37th international ACM SIGIR conference on research and development in information retrieval, SIGIR '14, Gold Coast, QLD, Australia – July 06–11, 2014*1265–1266. https://doi.org/10.1145/2600428.2611186.

Bast, H., & Buchhold, B. (2013). *An index for efficient semantic full-text search. Proceedings of the 22nd ACM international conference on information & knowledge managementCIKM '13*New York, NY, USA: ACM369–378. https://doi.org/10.1145/2505515.2505689.

Bast, H., Chitea, A., Suchanek, F., & Weber, I. (2007). *Ester: Efficient search on text, entities, and relations. Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrievalSIGIR '07*New York, NY, USA: ACM671–678. https://doi.org/10.1145/1277741.1277856.

Bast, H., & Weber, I. (2006). *Type less, find more: Fast autocompletion search with a succinct index. Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrievalSIGIR '06*New York, NY, USA: ACM364–371. https://doi.org/10.1145/1148170.1148234.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research, 3*(Feb), 1137–1155.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications ACM, 18*(9), 509–517. https://doi.org/10.1145/361002.361007.

Bordes, A., Weston, J., & Usunier, N. (2014). *Open question answering with weakly supervised embedding models. Joint European conference on machine learning and knowledge discovery in databases.* Springer165–180.

Catena, M., Macdonald, C., & Ounis, I. (2014). *On inverted index compression for search engine efficiency. Advances in information retrieval – 36th European conference on IR research, ECIR 2014, Amsterdam, The Netherlands, April 13–16, 2014. proceedings*359–371. https://doi.org/10.1007/978-3-319-06028-6_30.

Chen, H., Wei, B., Liu, Y., Li, Y., Yu, J., & Zhu, W. (2017). Bilinear joint learning of word and entity embeddings for entity linking. *Neurocomputing, https://doi.org/10.1016/j.neucom.2017.11.064*.

Chen, X., Xu, L., Liu, Z., Sun, M., & Luan, H.-B. (Xu, Liu, Sun, Luan, 2015a). *Joint learning of character and word embeddings. IJCAI*1236–1242.

Chen, Z., Lin, W., Chen, Q., Chen, X., Wei, S., Jiang, H., & Zhu, X. (Lin, Chen, Chen, Wei, Jiang, Zhu, 2015b). *Revisiting word embedding for contrasting meaning. Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing of the Asian* *federation of natural language processing, ACL 2015, July 26–31, 2015, Beijing, China, volume 1: Long papers*106–115.

Cheng, T., & Chang, K. C.-C. (2010). *Beyond pages: Supporting efficient, scalable entity search with dual-inversion index. Proceedings of the 13th international conference on extending database technologyEDBT '10*New York, NY, USA: ACM15–26. https://doi.org/10.1145/1739041.1739047.

Ciaccia, P., Patella, M., & Zezula, P. (1997). *M-tree: An efficient access method for similarity search in metric spaces. Proceedings of the 23rd international conference on very large data basesVLDB '97*San Francisco, CA, USA: Morgan Kaufmann Publishers Inc426–435.

Clinchant, S., & Perronnin, F. (2013). *Aggregating continuous word embeddings for information retrieval. Proceedings of the workshop on continuous vector space models and their compositionality*100–109.

Cornolti, M., Ferragina, P., & Ciaramita, M. (2013). A framework for benchmarking entity-annotation systems. In D. Schwabe, V. A. F. Almeida, H. Glaser, R. A. Baeza-Yates, & S. B. Moon (Eds.). *22nd international world wide web conference, WWW '13, Rio de Janeiro, Brazil, May 13–17, 2013* (pp. 249–260). International World Wide Web Conferences Steering Committee / ACM.

Dalton, J., Dietz, L., & Allan, J. (2014). *Entity query feature expansion using knowledge base links. Proceedings of the 37th international ACM SIGIR conference on research &#38; development in information retrievalSIGIR '14*New York, NY, USA: ACM365–374. https://doi.org/10.1145/2600428.2609628.

Diaz, F., Mitra, B., & Craswell, N. (2016). *Query expansion with locally-trained word embeddings. Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016, August 7–12, 2016, Berlin, Germany, volume 1: Long papers*.

Djuric, N., Wu, H., Radosavljevic, V., Grbovic, M., & Bhamidipati, N. (2015). *Hierarchical neural language models for joint representation of streaming documents and their content. Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee248–255.

Ensan, F., & Bagheri, E. (2017). *Document retrieval model through semantic linking. Proceedings of the tenth ACM international conference on web search and data miningWSDM '17*New York, NY, USA: ACM181–190. https://doi.org/10.1145/3018661.3018692.

Ensan, F., Bagheri, E., Zouaq, A., & Kouznetsov, A. (2017). *An empirical study of embedding features in learning to rank. Proceedings of the 2017 ACM on conference on information and knowledge managementCIKM '17*New York, NY, USA: ACM2059–2062. https://doi.org/10.1145/3132847.3133138.

Fang, W., Zhang, J., Wang, D., Chen, Z., & Li, M. (2016). *Entity disambiguation by knowledge and text jointly embedding. Proceedings of the 20th SIGNLL conference on computational natural language learning, conll 2016, Berlin, Germany, August 11–12, 2016*260–269.

Fernandez, M., Cantador, I., Lopez, V., Vallet, D., Castells, P., & Motta, E. (2011). Semantically enhanced information retrieval: An ontology-based approach. *Web Semantics: Science, Services and Agents on the World Wide Web, 9*(4), 434–452.

Fernández-Reyes, F. C., Hermosillo-Valadez, J., & Montes-y Gómez, M. (2018). A prospect-guided global query expansion strategy using word embeddings. *Information Processing and Management, 54*(1), 1–13.

Ferragina, P., & Scaiella, U. (2010). *Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). Proceedings of the 19th ACM international conference on information and knowledge managementCIKM '10*New York, NY, USA: ACM1625–1628. https://doi.org/10.1145/1871437.1871689.

Ganguly, D., Roy, D., Mitra, M., & Jones, G. J. (2015). *Word embedding based generalized language model for information retrieval. Proceedings of the 38th international ACM SIGIR conference on research and development in information retrievalSIGIR '15*New York, NY, USA: ACM795–798. https://doi.org/10.1145/2766462.2767780.

Grbovic, M., Djuric, N., Radosavljevic, V., Silvestri, F., & Bhamidipati, N. (2015). *Context- and content-aware embeddings for query rewriting in sponsored search. Proceedings of the 38th international ACM SIGIR conference on research and development in information retrievalSIGIR '15*New York, NY, USA: ACM383–392. https://doi.org/10.1145/2766462.2767709.

Guo, J., Fan, Y., Ai, Q., & Croft, W. B. (2016). *A deep relevance matching model for ad-hoc retrieval. Proceedings of the 25th ACM international on conference on information and knowledge managementCIKM '16*New York, NY, USA: ACM55–64. https://doi.org/10.1145/2983323.2983769.

Hashimoto, T. B., Alvarez-Melis, D., & Jaakkola, T. S. (2016). Word embeddings as metric recovery in semantic spaces. *Transactions of the Association for Computational Linguistics, 4*, 273–286.

Hasibi, F., Balog, K., & Bratsberg, S. E. (2016). *Exploiting entity linking in queries for entity retrieval. Proceedings of the 2016 ACM international conference on the theory of information retrievalICTIR '16*New York, NY, USA: ACM209–218. https://doi.org/10.1145/2970398.2970406.

Heinz, S., & Zobel, J. (2003). Efficient single-pass index construction for text databases. *Journal of the American Society for Information Science and Technology, 54*(8), 713–729.

Hu, Z., Huang, P., Deng, Y., Gao, Y., & Xing, E. P. (2015). *Entity hierarchy embedding. Annual meeting of the association for computational linguistics, ACL 2015*1292–1300.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013). *Learning deep structured semantic models for web search using clickthrough data. Proceedings of the 22nd ACM international conference on information & knowledge managementCIKM '13*New York, NY, USA: ACM2333–2338. https://doi.org/10.1145/2505515.2505665.

Indyk, P., & Motwani, R. (1998). *Approximate nearest neighbors: Towards removing the curse of dimensionality. Proceedings of the thirtieth annual ACM symposium on theory of computingSTOC '98*New York, NY, USA: ACM604–613. https://doi.org/10.1145/276698.276876.

Jameel, S., Bouraoui, Z., & Schockaert, S. (2017). *Member: Max-margin based embeddings for entity retrieval. Proceedings of the 40th international ACM SIGIR conference on research and development in information retrievalSIGIR '17*New York, NY, USA: ACM783–792. https://doi.org/10.1145/3077136.3080803.

Kenter, T., & de Rijke, M. (2015). *Short text similarity with word embeddings. Proceedings of the 24th ACM international on conference on information and knowledge managementCIKM '15*New York, NY, USA: ACM1411–1420. https://doi.org/10.1145/2806416.2806475.

Kim, S., Fiorini, N., Wilbur, W. J., & Lu, Z. (2017). Bridging the gap: Incorporating a semantic similarity measure for effectively mapping pubmed queries to documents. *Journal of Biomedical Informatics, 75*, 122–127. https://doi.org/10.1016/j.jbi.2017.09.014.

Konow, R., Navarro, G., Clarke, C. L., & López-Ortíz, A. (2013). *Faster and smaller inverted indices with treaps. Proceedings of the 36th international ACM SIGIR conference on research and development in information retrievalSIGIR '13*New York, NY, USA: ACM193–202. https://doi.org/10.1145/2484028.2484088.

Kusner, M. J., Sun, Y., Kolkin, N. I., & Weinberger, K. Q. (2015). *From word embeddings to document distances. Proceedings of the 32nd international conference on machine learning, ICML 2015, Lille, France, 6–11 July 2015*957–966.

Kuzi, S., Shtok, A., & Kurland, O. (2016). *Query expansion using word embeddings. Proceedings of the 25th ACM international on conference on information and knowledge managementCIKM '16*New York, NY, USA: ACM1929–1932. https://doi.org/10.1145/2983323.2983876.

Lashkari, F., Ensan, F., Bagheri, E., & Ghorbani, A. A. (2017). Efficient indexing for semantic search. *Expert Systems With Applications, 73*, 92–114. https://doi.org/10.1016/j.eswa.2016.12.033.

Lavrenko, V., & Croft, W. B. (2001). *Relevance based language models. Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrievalSIGIR '01*New York, NY, USA: ACM120–127. https://doi.org/10.1145/383952.383972.

Le, Q. V., & Mikolov, T. (2014). *Distributed representations of sentences and documents. Proceedings of the 31th international conference on machine learning, ICML 2014, Beijing, China, 21–26 june 2014*1188–1196.

Lee, J., Min, J.-K., Oh, A., & Chung, C.-W. (2014). Effective ranking and search techniques for web resources considering semantic relationships. *Information Processing and Management, 50*(1), 132–155.

Li, X., Li, C., & Yu, C. (2010). *Entityengine: Answering entity-relationship queries using shallow semantics. Proceedings of the 19th ACM international conference on information and knowledge managementCIKM '10*New York, NY, USA: ACM1925–1926. https://doi.org/10.1145/1871437.1871766.

Liu, X., & Fang, H. (2015). Latent entity space: A novel retrieval approach for entity-bearing queries. *Information Retrieval Journal, 18*(6), 473–503.

Ma, B., Zhang, N., Liu, G., Li, L., & Yuan, H. (2016). Semantic search for public opinions on urban affairs: A probabilistic topic modeling-based approach. *Information Processing and Management, 52*(3), 430–445. https://doi.org/10.1016/j.ipm.2015.10.004.

Maaten, L.v.d., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research, 9*(Nov), 2579–2605.

Mangold, C. (2007). A survey and classification of semantic search approaches. *IJMSO, 2*(1), 23–34. https://doi.org/10.1504/IJMSO.2007.015073.

Meij, E., Balog, K., & Odijk, D. (2014). *Entity linking and retrieval for semantic search. Proceedings of the 7th ACM international conference on web search and data miningWSDM '14*New York, NY, USA: ACM683–684. https://doi.org/10.1145/2556195.2556201.

Metzler, D., & Croft, W. B. (2005). *A markov random field model for term dependencies. Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrievalSIGIR '05*New York, NY, USA: ACM472–479. https://doi.org/10.1145/1076034.1076115.

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., & Khudanpur, S. (2010). *Recurrent neural network based language model. Eleventh annual conference of the international speech communication association*.

Mikolov, T., Kombrink, S., Burget, L., Černockỳ, J., & Khudanpur, S. (2011). *Extensions of recurrent neural network language model. Acoustics, speech and signal processing (ICASSP), 2011 ieee international conference on.* IEEE5528–5531.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (Sutskever, Chen, Corrado, Dean, 2013a). *Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems 26: 27th annual conference on neural information processing systems 2013. Proceedings of a meeting held December 5–8, 2013, Lake Tahoe, Nevada, United States.*3111–3119.

Mikolov, T., Yih, W., & Zweig, G. (Yih, Zweig, 2013b). *Linguistic regularities in continuous space word representations. Human language technologies: Conference of the North American chapter of the association of computational linguistics, proceedings, June 9–14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*746–751.

Moreno, J. G., Besançon, R., Beaumont, R., D'hondt, E., Ligozat, A., Rosset, S., et al. (2017). *Combining word and entity embeddings for entity linking. The semantic web – 14th international conference, ESWC 2017, Portorož, Slovenia, May 28–June 1, 2017, proceedings, part I*337–352. https://doi.org/10.1007/978-3-319-58068-5_21.

Nalisnick, E., Mitra, B., Craswell, N., & Caruana, R. (2016). *Improving document ranking with dual word embeddings. Proceedings of the 25th international conference companion on world wide webWWW '16 Companion*Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee83–84. https://doi.org/10.1145/2872518.2889361.

Pappu, A., Blanco, R., Mehdad, Y., Stent, A., & Thadani, K. (2017). *Lightweight multilingual entity extraction and linking. Proceedings of the tenth ACM international conference on web search and data miningWSDM '17*New York, NY, USA: ACM365–374. https://doi.org/10.1145/3018661.3018724.

Persin, M., Zobel, J., & Sacks-Davis, R. (1996). Filtered document retrieval with frequency-sorted indexes. *JASIS, 47*(10), 749–764.

Řehůřek, R., & Sojka, P. (2010). *Software Framework for Topic Modelling with Large Corpora. Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks.* Valletta, Malta: ELRA45–50 http://is.muni.cz/publication/884893/en

Sak, H., Senior, A., & Beaufays, F. (2014). *Long short-term memory recurrent neural network architectures for large scale acoustic modeling. Fifteenth annual conference of the international speech communication association*.

Sánchez, D., Batet, M., Isern, D., & Valls, A. (2012). Ontology-based semantic similarity: A new feature-based approach. *Expert Systems with Applications, 39*(9), 7718–7728.

Shwartz, V., & Dagan, I. (2016). *Cogalex-v shared task: Lexnet - integrated path-based and distributional method for the identification of semantic relations. Proceedings of the 5th workshop on cognitive aspects of the Lexicon, cogalex@coling 2016, Osaka, Japan, December 12, 2016*80–85.

Strohman, T., Metzler, D., Turtle, H., & Croft, W. B. (2005). *Indri: A language model-based search engine for complex queries. Proceedings of the international conference on intelligent analysis2. Proceedings of the international conference on intelligent analysis* Amherst, MA, USA2–6.

Sugawara, K., Kobayashi, H., & Iwasaki, M. (2016). *On approximately searching for similar word embeddings. Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016, August 7–12, 2016, Berlin, Germany, Volume 1: Long papers*.

Sundermeyer, M., Schlüter, R., & Ney, H. (2012). *Lstm neural networks for language modeling. Thirteenth annual conference of the international speech communication association*.

Tablan, V., Bontcheva, K., Roberts, I., & Cunningham, H. (2015). Mímir: An open-source semantic search framework for interactive information seeking and discovery. *Journal of Web Semantics, 30*, 52–68. https://doi.org/10.1016/j.websem.2014.10.002.

Teevan, J. (2017). *Search, re-search. Keynote at the 39th European conference on information retrieval, Aberdeen, Scotland [Accessed: 2018 01 21]* http://teevan.org/publications/

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., & Gamon, M. (2015). *Representing text for joint embedding of text and knowledge bases. Proceedings of the 2015 conference on empirical methods in natural language processing*1499–1509.

Trieu, L. Q., Tran, H. Q., & Tran, M.-T. (2017). *News classification from social media using twitter-based doc2vec model and automatic query expansion. Proceedings of the eighth international symposium on information and communication technologySoICT 2017*New York, NY, USA: ACM460–467. https://doi.org/10.1145/3155133.3155206.

Van Gysel, C., de Rijke, M., & Kanoulas, E. (de Rijke, Kanoulas, 2016a). *Learning latent vector spaces for product search. Proceedings of the 25th ACM international on conference on information and knowledge managementCIKM '16*New York, NY, USA: ACM165–174. https://doi.org/10.1145/2983323.2983702.

Van Gysel, C., de Rijke, M., & Worring, M. (2015). *Semantic entities. Proceedings of the eighth workshop on exploiting semantic annotations in information retrievalESAIR '15*New York, NY, USA: ACM1–2. https://doi.org/10.1145/2810133.2810139.

Van Gysel, C., de Rijke, M., & Worring, M. (de Rijke, Worring, 2016b). *Unsupervised, efficient and semantic expertise retrieval. Proceedings of the 25th international conference on world wide webWWW '16*Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee1069–1079. https://doi.org/10.1145/2872427.2882974.

Wang, H., Liu, Q., Penin, T., Fu, L., Zhang, L., Tran, T., et al. (2009). Semplore: A scalable ir approach to search the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web, 7*(3), 177–188.

Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). *Knowledge graph and text jointly embedding. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*1591–1601.

Xiong, C., Callan, J., & Liu, T.-Y. (2017). *Word-entity duet representations for document ranking. Proceedings of the 40th international ACM SIGIR conference on research and development in information retrievalSIGIR '17*New York, NY, USA: ACM763–772. https://doi.org/10.1145/3077136.3080768.

Yamada, I., Shindo, H., Takeda, H., & Takefuji, Y. (Shindo, Takeda, Takefuji, 2016a). *Joint learning of the embedding of words and entities for named entity disambiguation. Proceedings of the 20th SIGNLL conference on computational natural language learning, CONLL 2016, Berlin, Germany, August 11–12, 2016*250–259.

Yamada, I., Shindo, H., Takeda, H., & Takefuji, Y. (Shindo, Takeda, Takefuji, 2016b). *Joint learning of the embedding of words and entities for named entity disambiguation. The signll conference on computational natural language learning (CONLL)*.

Yang, M.-C., Lee, D.-G., Park, S.-Y., & Rim, H.-C. (2015). Knowledge-based question answering using the semantic embedding space. *Expert Systems with Applications, 42*(23), 9086–9104.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval, 1*(1–2), 69–90.

Yang, Y., & Liu, X. (1999). *A re-examination of text categorization methods. Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrievalSIGIR '99*New York, NY, USA: ACM42–49. https://doi.org/10.1145/312624.312647.

Zamani, H., & Croft, W. B. (2016). *Embedding-based query language models. Proceedings of the 2016 ACM international conference on the theory of information retrievalICTIR '16*New York, NY, USA: ACM147–156. https://doi.org/10.1145/2970398.2970405.

Zhou, G., He, T., Zhao, J., & Hu, P. (2015). *Learning continuous word embedding with metadata for question retrieval in community question answering. Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing of the Asian federation of natural language processing, ACL 2015, July 26–31, 2015, Beijing, China, Volume 1: Long papers*250–259.

Zou, B., Lampos, V., Liang, S., Ren, Z., Yilmaz, E., & Cox, I. (2017). *A concept language model for ad-hoc retrieval. Proceedings of the 26th international conference on world wide web companionWWW '17 Companion*Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee885–886. https://doi.org/10.1145/3041021.3054209.

Zuccon, G., Koopman, B., Bruza, P., & Azzopardi, L. (2015). *Integrating and evaluating neural word embeddings in information retrieval. Proceedings of the 20th australasian document computing symposiumADCS '15*New York, NY, USA: ACM12:1–12:8. https://doi.org/10.1145/2838931.2838936.

Zwicklbauer, S., Seifert, C., & Granitzer, M. (2016). *Robust and collective entity disambiguation through semantic embeddings. Proceedings of the 39th international ACM SIGIR conference on research and development in information retrievalSIGIR '16*New York, NY, USA: ACM425–434. https://doi.org/10.1145/2911451.2911535.