Retrieving Textual Evidence for Knowledge Graph Facts

Gonenc Ercan[1,2], Shady Elbassuoni[3], and Katja Hose[1]

[1] Aalborg University, Denmark {gonenc,khose}@cs.aau.dk
[2] Hacettepe University, Informatics Institute Ankara, Turkey
[3] American University of Beirut, Lebanon se58@aub.edu.lb

Abstract. Knowledge graphs have become vital resources for semantic search and provide users with precise answers to their information needs. Knowledge graphs often consist of billions of facts, typically encoded in the form of RDF triples. In most cases, these facts are extracted automatically and can thus be susceptible to errors. For many applications, it can therefore be very useful to complement knowledge graph facts with textual evidence. For instance, it can help users make informed decisions about the validity of the facts that are returned as part of an answer to a query. In this paper, we therefore propose FacTify, an approach that given a knowledge graph and a text corpus, retrieves the top-k most relevant textual passages for a given set of facts. Since our goal is to retrieve short passages, we develop a set of IR models combining exact matching through the Okapi BM25 model with semantic matching using word embeddings. To evaluate our approach, we built an extensive benchmark consisting of facts extracted from YAGO and text passages retrieved from Wikipedia. Our experimental results demonstrate the effectiveness of our approach in retrieving textual evidence for knowledge graph facts.

1 Introduction

Knowledge graphs, such as YAGO [23], DBpedia [1], and Google Knowledge Graph[1], have caused a revolution in information retrieval (IR). These knowledge graphs encode billions of facts, typically in the form of RDF triples[2]. For example, the following two facts from YAGO indicate that the late President John F. Kennedy died in Dallas on November 22, 1963:

```
John_F._Kennedy diedIn Dallas
John_F._Kennedy diedOnDate "1963-11-22"
```

The above set of facts, in the following referred to as a *knowledge subgraph*, provide very concise information about President Kennedy's place and date of death. However, it lacks any context that a user might be interested in. In this

[1] https://developers.google.com/knowledge-graph/
[2] https://www.w3.org/RDF/

paper, we therefore propose FacTify, an approach to complement knowledge subgraphs with *relevant* text passages extracted from a text corpus. For the above mentioned example, a relevant passage extracted from Wikipedia is:

> President Kennedy was assassinated in Dallas, Texas, at 12:30 pm Central Standard Time on Friday, November 22, 1963. He was in Texas on a political trip to smooth over frictions in the Democratic Party between liberals Ralph Yarborough and Don Yarborough (no relation) and conservative John Connally. Traveling in a presidential motorcade through downtown Dallas, he was shot once in the back, the bullet exiting via his throat, and once in the head.

Passages, such as the one above, serve many purposes. First, they provide evidence on the validity of the knowledge graph facts. This is particularly crucial in this era of "Alternative Facts", where misinformation and disinformation are flooding the Internet, and where there is a strong plea for providing more transparency in the way intelligent systems work. Second, the facts in many of the currently existing large-scale knowledge graphs are automatically constructed by making use of Information Extraction and NLP techniques. They are thus susceptible to errors and can have some invalid or inaccurate facts. Providing passages along with knowledge subgraphs allows the users to make informed decisions about the validity of the facts. Third, these passages will also contain additional information related to the knowledge subgraphs; information that might not even be present in the whole knowledge graph. For instance, from the passage shown above, the user can infer that President Kennedy was *assassinated*, and that he was shot twice.

Our task can be formulated as follows: given a knowledge graph and a text corpus, retrieve the *top-k most relevant passages* for a given knowledge subgraph consisting of one or *multiple* facts in the form of RDF triples. In this paper, we define passages as short ones consisting of *three* sentences. However, our approach is applicable to passages of any length. Since our goal is to retrieve short passages, we propose to combine exact matching using the Okapi BM25 model [22] with semantic matching using word embeddings. Word embeddings are distributed vector representations of words, which are capable of capturing semantic and syntactic relations between the words. To evaluate the effectiveness of our approach, we built a benchmark consisting of 56 knowledge subgraphs extracted from YAGO and a pool of passages retrieved from Wikipedia, which were assessed using Figure Eight crowdsourcing platform[3]. Our experimental results indicate that our approach, which combines exact and semantic matching, results in statistically significant improvements in IR effectiveness compared to using each type of matching on its own.

In summary, the contributions of this paper can be summarized as follows:

- An approach (FacTify) for retrieving relevant passages for knowledge graph facts as well as knowledge subgraphs consisting of multiple facts.

[3] https://www.figure-eight.com/

– A retrieval method that combines exact matching through the Okapi BM25 ranking model with semantic matching through word embeddings.
– An extensive comparative study of multiple IR models for the above task. The created benchmark containing knowledge subgraphs from YAGO and passages retrieved from Wikipedia has been made publicly available (http://qweb.cs.aau.dk/factify/).

The remainder of this paper is organized as follows. In Section 2, we discuss related work. Section 3 describes our approach to retrieve textual evidence for knowledge graph facts. In Section 4, we give an overview of our benchmark and then present our experimental results. Finally, we conclude and discuss future work in Section 5.

2 Related Work

Our work is related to two different areas. The first consists of IR models used to retrieve passages for knowledge graph facts and the second consists of IR models that make use of word embeddings to rank search results. We discuss each one separately in the following.

2.1 Passage Retrieval for Knowledge Graph Facts

To the best of our knowledge, there are only a handful of approaches for the problem of retrieving text passage for knowledge graph facts [2, 6, 9, 18]. All these approaches, with the exception of [6, 18], operate on single facts only. For instance, the approach in [2] relies on language modeling to rank short passages for a given input fact. It assumes the presence of labels for the components of the facts and ranks the passages based on the passages' probabilities of generating the words in the concatenation of these labels. It then assumes independence between the words and smooths the passage probabilities with document and collection probabilities. The approach, however, operates only on the level of single facts, and it assumes independence between the words representing the fact. Moreover, the approach deploys only exact matching, i.e., semantically related words to those in the fact labels would not play any role in the ranking of passages unless a list of synonyms for terms is provided. In this paper, we experimentally compare our approach to [2] and show that our approach significantly outperforms this approach in terms of retrieving relevant passages for knowledge subgraphs.

DeFacto [9] retrieves webpages that are relevant for a given input fact by issuing several queries to a Web search engine. These queries are generated by verbalizing the fact using natural-language patterns extracted by the BOA framework [10]. Once the webpages have been retrieved, they again use the BOA framework in addition to light-weight NLP techniques to determine whether a webpage contains useful evidence for the given fact. They then use a set of classifiers to predict trustworthiness of the webpages retrieved as well as the fact itself. As can be seen, DeFacto relies heavily on external search engines, NLP,

machine learning and the BOA framework to retrieve textual evidence and estimate their trustworthiness as well as that of the fact. On the other hand, our approach is a general retrieval framework that can seamlessly operate on any knowledge graph and text corpus and does not rely on any external information that might not be available for certain knowledge graphs or corpora.

Similar to our approach, ROXXI [6, 18] can be applied to knowledge subgraphs containing multiple facts. However, it retrieves documents rather than passages as in our case. To do this, ROXXI makes use of textual patterns available in PATTY [20] to rank these documents using a language modeling based approach. Once retrieved, the documents are annotated to highlight the textual patterns of the given facts and a text snippet is generated using those textual patterns. However, ROXXI again relies on external textual patterns, which might not be available in general for many knowledge graphs.

Finally, ExFakt [7] is a framework that generates human-comprehensible explanations for knowledge graph facts form a given text corpus. It uses background knowledge encoded in the form of Horn clauses to rewrite a fact into a set of other easier-to-spot facts. The final output of the framework is a set of *semantic traces* for the fact generated from both the text corpus and the knowledge graph. However, ExFakt does not retrieve text passages for knowledge graph facts as in our case, and it can only support single facts.

2.2 Word Embeddings for Information Retrieval

As indicated above, our goal is to retrieve textual passage for a given set of facts. Vocabulary mismatch is a major challenge for many retrieval tasks, including ours. The semantic matching capability of latent models alleviates the vocabulary mismatch problem. Latent models, ranging from latent semantic models [16] to neural network based word embeddings [19], are used extensively in natural language processing and more recently in information retrieval tasks. Nevertheless, building effective document representations using word embeddings is an open research topic.

A basic method to build document vectors from word embeddings is to aggregate the word vectors by summation or averaging [24]. Different aggregation strategies including Fisher Vectors are also used to build document vectors [4,25]. As an alternative, the literature proposes methods that jointly learn the document representations (D2V) with word embeddings [17]. Document vectors are integrated into Skip-gram and CBOW methods [19] by adding an embedding layer for the documents.

While aggregation based methods provide a concise representation of documents, they do not fully take advantage of all pairwise similarities between the words in the documents. A method that makes use of pairwise similarities of words in the documents is the Word Mover's distance (WMD) [15]. WMD casts the similarity problem as a transportation problem, where the words in the first document produce items that should be transported to the second document's words and the cost of transportation is determined by the distance between the words in the embedding space. The production amount of words in the source

document and the capacity of the words in the target document are determined by weights calculated using tf or tf-idf weights of the words. These weights control the amount of pairwise similarity taken into account. If the target words have infinite capacity, then transportation will take place between the most similar word pairs, this version of WMD is often referred to as Relaxed WMD. If the weights in the source document are large, then WMD uses all pairwise similarities.

Guo et al. [11] argue that both aggregate and WMD methods loosely match words in queries, thus causing a concept drift and degrading the performance of word embedding based IR methods. As a solution to this problem, they propose a non-linear scaling function depending on the inverse document frequency of the query word, allowing more strict matching for words with a high idf value, while retaining the ability to semantically match less discriminative words. Galke et al. [8] compare both aggregate and WMD based methods in an information retrieval task. The aggregate method using normalized tf-idf word weights achieves better results in a range of retrieval experiments compared to WMD and D2V methods. However, word embedding methods fail to beat the tf-idf baseline. Brokos et al. [3] use Relaxed WMD for document retrieval in question answering. They report better results using Relaxed WMD for re-ranking the results. Kenter and de Rijke [14] use Relaxed WMD and BM25 based weighting function as a feature in a sentence paraphrase classifier.

In this paper, we assess the applicability of embedding based methods when applied to our task of retrieving text passages for knowledge graph facts. More precisely, we propose hybrid methods that combine embedding based semantic matching with exact matching as we describe next.

3 Retrieving Textual Evidence

As mentioned in the introduction, the problem we are targeting in this paper can be defined as follows. Given a knowledge graph and a text corpus, retrieve the top-k most relevant passages for a given input knowledge subgraph. A knowledge subgraph is a subgraph of the underlying knowledge graph that consists of one or more RDF triples, i.e., facts. A passage is defined as a short text excerpt, for instance consisting of three sentences extracted from the text corpus. Finally, a relevant passage is a passage containing textual evidence that verifies the facts in the knowledge subgraph.

Since knowledge subgraphs are in RDF while passages are in plain text, we use the *labels* (i.e., surface names) of resources in the knowledge graph to transform a knowledge subgraph into a keyword query. For instance, in YAGO, `John_F._Kennedy` is associated with the label *John F. Kennedy*. Similarly, the relation `diedIn` can be represented using the label *died in*. Our approach thus first transforms the knowledge subgraph for which passages are to be retrieved into a keyword query by replacing the RDF triples with their labels. For example, consider the knowledge subgraph

John_F._Kennedy diedIn Dallas
John_F._Kennedy diedOnDate "1963-11-22"

The corresponding keyword query for the above subgraph is *"John F. Kennedy died in Dallas John F. Kennedy died on date 1963-11-22"*. Note that the date "1963-11-22" is a literal and thus does not need to be substituted by a label.

Now, our original task of retrieving passages for a given knowledge subgraph is transformed into retrieving passages for the obtained keyword query representing the knowledge subgraph. We propose to combine two types of IR models to achieve this task. The first utilizes exact matching using term frequencies and the second employs semantic matching using word embeddings. Exact matching relies on term frequencies to rank query results. It thus assumes that the query terms will appear in some form in the documents. While this is an adequate assumption when retrieving long documents, short passages, such as the ones we are concerned with in this paper, might not miss some of the query terms but can still be highly relevant. For example, the word "`married`" in a query might not appear at all in some of the relevant passages that contain the words "`wife`" or "`husband`" instead.

A remedy to this vocabulary mismatch problem is to make use of word embeddings. Word embeddings are distributed vector representation of words, capable of capturing semantic and syntactic relations between the words. They have been successfully employed in various natural language processing tasks, such as question-answering, document classification, and more recently in IR [8, 15, 17]. On the other hand, word embedding based approaches are susceptible to concept drift as a word can have a high similarity to collocated, hypernyms and meronyms as well as synonyms. Recent results hint at these challenges [8, 11].

Hence, we propose a hybrid model that combines both exact and semantic matching to retrieve relevant passages for knowledge graph facts. This is inspired by the results from previous work on other IR tasks such as biomedical document retrieval for question answering [3] or monolingual and cross-lingual information retrieval [24], in which a hybrid approach, such as the one we propose, has been shown to achieve improvements in IR effectiveness. In the next two subsections, we explain each type of model separately, and then finally describe our hybrid model that combines both.

3.1 Exact Matching

For exact matching we use the OKAPI BM25 [22] retrieval model. Ad-hoc information retrieval experiments in early TREC shared tasks show that the BM25 retrieval model is one of the most effective IR models. The BM25 model uses term frequencies to estimate the relevance of passages as follows.

$$BM25(S,Q) = \sum_{q \in Q} IDF(q) \frac{f(q,S)(k_1+1)}{f(q,S) + k_1(1 - b + b\frac{|S|}{avgsl})} \tag{1}$$

where $f(q_i, S)$ is the frequency of the query word q_i in passage S, $avgsl$ is the average passage length, and $IDF(q)$ is the inverse document frequency calcu-

lated using Equation 2. The terms k_1 and b are free parameters weighing the normalization and frequency components of the scoring function.

$$IDF(q) = log \frac{N - df(q) + 0.5}{df(q) + 0.5} \tag{2}$$

3.2 Semantic Matching

Word embeddings represent words as d-dimensional dense vectors. The similarity or distance between the vectors of words in the embedding space measures the semantic relatedness between them. Pre-trained word embeddings are typically built from a large corpus and have a vocabulary formed of frequent words in that corpus. The vocabulary of the word embeddings can contain both the upper and lower case form of a word. In addition, word vectors for some common phrases might be available. In the following, we describe several embedding based models that can be used to rank passages given a knowledge subgraph.

IWCS. The IDF re-weighted word centroid similarity (IWCS) model [8] uses the word embedding vectors to construct a d-dimensional vector representing a passage. In the IWCS model, the word vectors of the given text are aggregated into a single vector using a linear weighted combination of its word vectors. The weight of each word is determined by the tf-idf weighting. Let $t(S, w)$ be the L_2 normalized tf-idf weight for word w in the passage S and $\overrightarrow{w}$ be the word vector. A single vector for passage S can then be computed as follows:

$$\overrightarrow{S} = \sum_{w \in S} \overrightarrow{w} \times t(S, w) \tag{3}$$

The centroid vector for the query $\overrightarrow{Q}$ can also be constructed in a similar manner. Finally, to rank a passage S with respect to a given query Q, we utilize the cosine similarity between these two centroids, i.e., $IWCS(S, Q) = cosine(\overrightarrow{S}, \overrightarrow{Q})$.

Average of Query IWCS. While the IWCS model was shown to be effective in some IR benchmarks [8], it is difficult to encode diverse relationships between word vectors in the d-dimensional space with simple aggregation. In order to test the validity of this hypothesis, we propose a model that unfolds the query and use the average similarity of query words to the passage centroid as an estimate of relevance as follows.

$$QIWCS(S, Q) = \frac{1}{|Q|} \sum_{w \in Q} cosine(\overrightarrow{S}, \overrightarrow{w}) \times t(Q, w) \tag{4}$$

We refer to such model as QIWCS.

Pairwise Similarity. As a final word embedding based model, the average pairwise similarities between the query word vectors $\overrightarrow{q}$ and the passage word vectors $\overrightarrow{w}$ can be used as follows.

$$PairWise(S,Q) = \sum_{w \in S} \sum_{q \in Q} cosine(\overrightarrow{q}, \overrightarrow{w}) \times t(Q,q) \times t(S,w) \tag{5}$$

We will refer to this model as PairWise in the remainder of this paper.

3.3 FacTify Model

Our approach combines both exact and semantic matching as follows. Given a knowledge subgraph transformed into a keyword query Q and a passage S, `FacTify` utilizes a hybrid model that ranks a passage S based on a linear combination of its score using BM25 and a word embedding based one as follows:

$$FT(S,Q) = \alpha BM25(S,Q) + (1-\alpha) Embedding(S,Q) \tag{6}$$

where $FT(S,Q)$ is the score of passage S for query Q using `FacTify`'s hybrid model, $BM25(S,Q)$ is the BM25 score as computed using Equation 1, $Embedding(S,Q)$ is an embedding based similarity (Section 3.2), and α is a weighting parameter. This results in three different hybrid models: FT-IWCS, FT-QIWS, and FT-PairWise, which result from combining BM25 with IWCS, QIWCS, and PairWise.

4 Evaluation

In this section, we present the results of experimentally comparing `FacTify` with its different hybrid models to each other and to several baselines. We start by describing the baselines and the evaluation setup, then give an overview of our benchmark, and finally present our evaluation results.

4.1 Baselines

We compare `FacTify` to three different baselines. The first baseline is the plain BM25 model (see Equation 1) without combining it with word embeddings. The second baseline is a recent approach proposed by Bhatia et al. [2], which is based on language models (LM). This model assumes that the words in a fact are conditionally independent and uses the probability $P(Q|S)$ to rank a passage S based on its probabilities of generating the knowledge subgraph query Q as in Equation 8.

$$LM(Q,S) = P(Q|S) \propto \prod q \in Q P(q|S) \tag{7}$$

$$P(q|S) = \lambda_1 P(q|\theta_S) + \lambda_2 P(q|\theta_D) + \lambda_3 P(q|\theta_C) \tag{8}$$

where q is a query keyword, λ_1, λ_2, and λ_3 are weighting parameters. The probability of a word given a passage $P(q|S)$ is defined as a mixture of three different language models. The first is the passage language model θ_S, the second is the language model of the document from which the passage was extracted θ_D, and the third is the collection language model θ_C, where the collection is the whole text corpus. The language models are estimated using maximum likelihood estimators with Laplacian smoothing, where V is the vocabulary, D is the document containing the passage S, and C refers to the collection.

$$P(q|\theta_S) = \frac{f(q,S)+1}{|S|+|V|} \tag{9}$$

$$P(q|\theta_D) = \frac{f(q,D)+1}{|D|+|V|} \tag{10}$$

$$P(q|\theta_C) = \frac{f(q,C)}{|C|} \tag{11}$$

Note that this model uses two smoothing methods, Laplacian and Jelinek-Mercer smoothing, where the latter is achieved through the collection language model. Since the length of passages $|S|$ is relatively small with respect to $|V|$ and $|D|$, the Laplacian smoothing often weakens the signal encoded in $P(q|\theta_S)$. As a third baseline, we therefore propose a modification of the LM model that does not employ Laplacian smoothing, which we refer to as LM-noLap. Only Equations 9 and 10 are modified by removing the additive terms used for Laplacian smoothing.

4.2 Evaluation Setup

For the BM25 and the LM-based models, both the queries and the passages were tokenized, words were lower-cased and the passages were transformed into bags of words and indexed using Lucene[4]. For the hybrid models which make use of word embeddings, we used a Gazetteer approach to tokenize the text into word vectors. A Directed Acyclic Finite State Automata (DAFSA) was built for the vocabulary of the word embeddings using the construction algorithm of Daciuk et al. [5]. A word or phrase in the text was then mapped to a word vector with the closest form. Although any word embedding can be used, GloVe word embeddings pre-trained with 840 billion word corpora[5] and with word vector dimensions $d = 300$ was used [21].

The parameters of the Okapi BM25 were set as $k_1 = 1.2$, and $b = 0.75$, which are the default parameters optimized using the TREC dataset [13]. The linear weighted combination parameter α for the hybrid models in Equation 6 was empirically set to 0.2, the linear weighted combination parameters for the language model approaches were set as $\lambda_1 = 0.6$, $\lambda_2 = 0.2$, and $\lambda_3 = 0.2$ as suggested by the authors of that model [2].

[4] http://lucene.apache.org/

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

Table 1: Sample knowledge subgraphs and their corresponding keyword queries

Knowledge Subgraph	Keyword Query
Albert_Einstein graduatedFrom ETH_Zurich	Albert Einstein graduated from ETH Zurich
Marie_Curie wasBornIn Warsaw	Marie Curie was born in Warsaw
Marie_Curie was BornOn "1867-11-07"	Marie Curie was born on date "1867-11-07"
JHenry_Kissinger hasWonPrize Nobel_Peace_Prize	Henry Kissinger has won prize Nobel Peace Prize
Le_Duc_Tho hasWonPrize Nobel_Peace_Prize	Le Duc Tho has won prize Nobel Peace Prize
Dalai_Lama hasWonPrize Nobel_Peace_Prize	Dalai Lama has won prize Nobel Peace Prize
Kofi_Annan hasWonPrize Nobel_Peace_Prize	Kofi Annan has won prize Nobel Peace Prize

4.3 Benchmark

To evaluate the effectiveness of FacTify (Section 3), we constructed a benchmark consisting of 56 different knowledge subgraphs extracted from the large knowledge graph YAGO [23]. The knowledge subgraphs are composed of both single facts, for example Cuba hasCapital Havana, or multiple related facts such as Albert_Einstein wasBornIn Ulm; Albert_Einstein wasBornOnDate 1879. The average number of facts per knowledge subgraph in our benchmark is 1.41. Table 1 shows several sample knowledge subgraphs and their corresponding keyword queries.

For each knowledge subgraph in our benchmark, the top 20 passages using our three hybrid models and all the baselines were retrieved. Each passage was assessed for relevance using Figure Eight crowdsourcing platform[6], on a three-level scale: relevant, somehow relevant, or irrelevant. A passage was considered relevant if the annotator was capable to verify *all* the facts from the knowledge subgraph in the passage. If the annotator could only verify part of the knowledge subgraph, say a single fact for multi-fact subgraphs, or if the passage just implied the facts in the knowledge subgraph, the passage was marked as somehow relevant.

Overall, we had a pool of 4,145 unique passages for our 56 knowledge subgraphs. Each passage was annotated as described above by three different annotators. The inter-rater agreement between the three annotators was 0.39 using Fleiss' Kappa. A final relevance score for each passage was computed using a standard majority voting. For 329 passages, there was no consensus among the three annotators and these were thus assigned the mean of the three relevance levels (i.e., they were considered *somehow* relevant). Our benchmark is publicly available at http://qweb.cs.aau.dk/factify/.

[6] https://www.figure-eight.com/

Table 2: Average NDCG of the evaluated IR models

Model	NDCG@20	NDCG@10	NDCG@5	NDCG@1
FT-PairWise	**72.98**	**74.64**	**76.31**	**81.07**
FT-IWCS	71.90	73.01	75.00	80.00
FT-QIWCS	71.76	73.49	74.97	80.00
BM25	71.72	73.22	74.78	80.00
LM	31.59	32.55	35.05	31.07
LM-noLap	67.05	68.57	71.07	76.79

4.4 Results

To evaluate the effectiveness of our proposed models, we use the Normalized Discounted Cumulative Gain (NDCG) [12]. NDCG is a common metric for IR effectiveness that takes into consideration the rank of relevant results and allows the incorporation of different relevance levels. In addition, we also report the Mean Reciprocal Rank (MRR) and precision for the relevant passages. MRR and precision use binary relevance and allow us to evaluate the effectiveness of the different models in retrieving relevant passages which can *fully* verify all the facts in a knowledge subgraph (i.e, considering somehow relevant passages as irrelevant). Particularly, MMR evaluates whether the top-ranked results is relevant or not. Precision, on the other hand, measures the percentage of relevant results retrieved up to a particular rank.

In Table 2, we report the average NDCG at different ranks over all the knowledge subgraphs in our benchmark. Overall, the hybrid model FT-PairWise achieves the highest NDCG for all ranks. It significantly outperforms the plain BM25 model and LM-noLap model in ranks 5 to 20 (p-value < 0.05). When comparing the different hybrid models, the model that makes use of word embeddings without any aggregation, i.e., FT-PairWise, achieves the highest NDCG. Recall that FT-IWCS aggregates the word vectors of both the query and the passage, while FT-QIWCS aggregates only the words in the passage, and PairWise does not perform any aggregation. This result confirms our initial hypothesis that taking the sum of the word vectors results in information loss and can degrade the retrieval effectiveness. When the word embedding methods IWCS, QIWCS, and PairWise are used alone without combining them with the BM25 model, their performance was as low as 0.35 (NDCG@20).

Comparing the two language model baselines LM and LM-noLap, the LM model was significantly improved when the Laplacian smoothing was removed from the ranking model (i.e., the LM-noLap model). This result supports our intuition that using two smoothing techniques weakens the signal from the passage language model and gives more weight to the document one. Furthermore, having the document language model in the retrieval function improves the effectiveness for the top ranked passage, but tends to retrieve irrelevant passages from documents with high similarity to the query terms. The Okapi BM25 model achieved higher NDCG values compared to both the LM and LM-noLap models.

Table 3 shows the effectiveness of the evaluated models when it comes to retrieving *only* relevant passages (i.e., considering somehow relevant passages

Table 3: Average MRR and precision of the evaluated IR models

Methods	MRR	Precision@20	Precision@10	Precision@5	Precision@1
FT-PairWise	**75.97**	**50.03**	**54.82**	**58.93**	**66.07**
FT-IWCS	72.77	49.64	53.21	56.79	60.71
FT-QIWCS	72.75	49.29	53.93	56.43	60.71
BM25	72.80	49.38	53.57	55.71	60.71
LM	35.84	18.30	20.00	24.29	21.43
LM-noLap	74.96	46.25	52.14	57.50	**66.07**

as irrelevant). MRR measures the average rank of relevant passages. Both FT-PairWise and LM-noLap achieve higher MRR scores than BM25. On the other hand, the two other hybrid models FT-IWCS and FT-QIWCS did not show consistent improvements over BM25 when retrieving only relevant passages. We also report the precision for all models, and again the FT-PairWise and LM-noLap models were the most effective in retrieving a relevant passage in the top rank (i.e., Precision@1) for 66% of the knowledge subgraphs in our benchmark. When considering lower ranks, the precision of LM-noLap drops, even below that of BM25 when considering the top 10 passages. On the other hand, FT-PairWise was able to retrieve more relevant passages at all ranks. We conjuncture again that this is a consequence of relying on the document level language model, which limits the top ranking passages to only relevant documents, penalizing passages retrieved from documents that might not be entirely about the resources mentioned in the subgraph, but that might nonetheless contain relevant passages.

4.5 Discussion

When considering the results of the LM and LM-noLap models, our observation is that their performance is degraded mostly due to the document language model included in their ranking models. Especially when the Laplacian smoothing is used, the passage-level language model is smoothed more harshly as the length of the passages are relatively smaller compared to the vocabulary size. This gives significantly more weight to the document language model. Wikipedia contains articles listing information such as birth dates or various profiles grouped by an aspect (e.g., U.S Presidents). These articles repeat certain words like *born* or numbers in dates with high frequencies, causing a high probability for generating these specific words. The original model LM is affected by this component and retrieve passages from these documents containing only a portion of the words with high frequencies. Removing the Laplacian smoothing alleviates this problem, as the passage-level language model will have more weight and the frequencies will be leveled-out. Moreover, the document-level language model improves the ranking of passages extracted from relevant documents, for example for a subgraph about `Abraham_Lincoln`, the LM model retrieves passages from his Wikipedia article in the top ranks. While this strategy improves the accuracy of the top 1 result, it degrades the accuracy of the following passages in the result set as typically the same fact is not repeated in the same document multiple times.

Table 4: Top ranked passages retrieved by FacTify for some knowledge subgraphs

Query	Top Ranking Passage
Albert_Einstein graduatedFrom ETH_Zurich	ETH Zurich has produced and attracted many famous scientists in its short history, including Albert Einstein. More than twenty Nobel laureates have either studied at ETH or were awarded the Nobel Prize for their work achieved at ETH. Other alumni include scientists who were distinguished with the highest honours in their respective fields, amongst them Fields Medal, Pritzker Prize and Turing Award winners.
Allianz_Arena isLocatedIn Munich	Allianz provided naming rights for the Allianz Arena, a football stadium in the north of Munich, Germany. The two professional Munich football clubs Bayern Munich have played their home games at Allianz Arena since the start of the 2005/06 season. TSV 1860 München have played their home games at Allianz Arena until the 2016/17 season.
Adolf_Hitler created Mein_Kampf	Mein Kampf (My Struggle) is a 1925 autobiographical book by Nazi Party leader Adolf Hitler. The work describes the process by which Hitler became antisemitic and outlines his political ideology and future plans for Germany. Volume 1 of Mein Kampf was published in 1925 and Volume 2 in 1926.
Henri_Becquerel hasWonPrize Nobel_Prize Marie_Curie hasWonPrize Nobel_Prize	His grandparents, Marie and Pierre Curie together with Henri Becquerel won the Nobel Prize in Physics in 1903 for their study of radioactivity. Marie also won the Nobel Prize in Chemistry in 1911. Joliot's parents, Irène Joliot-Curie and Frédéric Joliot-Curie, won the Nobel Prize in Chemistry in 1935 for their discovery of artificial radioactivity.

On the other hand, since our proposed hybrid models do not rely on any document-level information, this allows them to retrieve passages from various Wikipedia articles and that might not be strongly associated with one particular resource in the knowledge subgraph. For example, for the subgraph BarackObama graduatedFrom ColumbiaUniversity a relevant passage from the Wikipedia article titled "College Transfer" is retrieved stating that Barack Obama transferred from Occidental College to Columbia University. Similar observations can be made for all the knowledge subgraphs in our benchmark, as facts are usually repeated in different articles with different contextual information, rather than just only appearing in the most relevant articles. We thus advocate that incorporating document-level information should be limited when retrieving passages to complement knowledge subgraphs.

As for BM25 , it uses the parameter k_1 to determine the saturation level of word frequencies, which is important for scoring passages containing only part of the subgraph with high frequency. LM on the other hand is linearly affected by word frequencies, a passage containing the word "born" but not the objects and subjects in the subgraph will have a higher score with LM than with BM25.

Finally, in Table 4, we display the top passages retrieved by our best-performing model, FT-PairWise, for several example knowledge subgraphs. As

can be seen from the table, the top ranked passages are all relevant to the knowledge subgraphs. They provide textual evidence that verify the facts in the knowledge subgraphs and in addition might provide additional contextual information. For instance, for the first knowledge subgraph in Table 4, one can verify that Albert Einstein graduated from ETH Zurich, but that also more than twenty Nobel laureates have either studied at ETH or were awarded the Nobel Prize for their work achieved at ETH. Similarly, for the last subgraph in Table 4, the top passage does not only confirm that Marie Curie and Henri Becquerel won the Nobel Prize in Physics but it also indicates that Curie later won another Nobel Prize in Chemistry, and that her daughter Irène Curie also won a Nobel prize in Chemistry. Note that this passage was retrieved from the Wikipedia article of Marie and Pierre Curie's grandson, Pierre Joliot. Similar interesting information can be deduced from the top passages retrieved for the other two knowledge subgraphs shown in Table 4 as well.

5 Conclusion

In this paper, we have proposed `FacTify`, a novel approach to retrieve textual passages for knowledge subgraphs consisting of one ore more facts. We proposed multiple IR models that combine exact matching through the Okapi BM25 model with semantic matching using word embeddings. We evaluated our approach using a benchmark consisting of 56 knowledge subgraphs extracted from YAGO and passages retrieved from Wikipedia, which were manually annotated for relevance through crowdsourcing. Our experimental results show that our approach outperforms the baselines and related work for retrieving relevant passages for knowledge graph facts. In future work, we plan to evaluate our approach on other knowledge graphs and text corpora. Furthermore, we plan to use our benchmark to train supervised deep-learning models that can rank passages based on their relevance to knowledge subgraphs consisting of one or more facts.

Acknowledgments. This research was partially funded by the Danish Council for Independent Research (DFF) under grant agreement no. DFF-8048-00051B and Aalborg University's Talent Management Programme.

References

1. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *The Semantic Web*, pages 722–735. Springer, 2007.
2. S. Bhatia, P. Dwivedi, and A. Kaur. Tell Me Why Is It So? Explaining Knowledge Graph Relationships by Finding Descriptive Support Passages. *ISWC*, 2018.
3. G.-I. Brokos, P. Malakasiotis, and I. Androutsopoulos. Using Centroids of Word Embeddings and Word Mover's Distance for Biomedical Document Retrieval in Question Answering. *ACL*, page 114, 2016.
4. S. Clinchant and F. Perronnin. Aggregating continuous word embeddings for information retrieval. In *ACL*, pages 100–109, 2013.

5. J. Daciuk, S. Mihov, B. W. Watson, and R. E. Watson. Incremental construction of minimal acyclic finite-state automata. *Computational linguistics*, 26(1):3–16, 2000.
6. S. Elbassuoni, K. Hose, S. Metzger, and R. Schenkel. ROXXI: Reviving witness dOcuments to eXplore eXtracted Information. *PVLDB*, 3(2):1589–1592, 2010.
7. M. Gad-Elrab, D. Stepanova, J. Urbani, and G. Weikum. ExFaKT: A Framework for Explaining Facts over Knowledge Graphs and Text. In *WSDM*, 2019.
8. L. Galke, A. Saleh, and A. Scherp. Word Embeddings for Practical Information Retrieval. *INFORMATIK*, 2017.
9. D. Gerber, D. Esteves, J. Lehmann, L. Bühmann, R. Usbeck, A.-C. N. Ngomo, and R. Speck. Defacto - temporal and multilingual deep fact validation. *Web Semantics*, 35:85–101, 2015.
10. D. Gerber and A.-C. N. Ngomo. Bootstrapping the linked data web. In *Workshop on Web Scale Knowledge Extraction*, 2011.
11. J. Guo, Y. Fan, Q. Ai, and W. B. Croft. Semantic matching by non-linear word transportation for information retrieval. In *CIKM*, pages 701–710, 2016.
12. K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM TOIS*, 20(4):422–446, 2002.
13. K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Inf. Process. Manage.*, 36(6):809–840, 2000.
14. T. Kenter and M. De Rijke. Short text similarity with word embeddings. In *CIKM*, pages 1411–1420, 2015.
15. M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. From word embeddings to document distances. In *ICML*, pages 957–966, 2015.
16. T. K. Landauer and S. T. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211, 1997.
17. Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196, 2014.
18. S. Metzger, S. Elbassuoni, K. Hose, and R. Schenkel. S3K: seeking statement-supporting top-K witnesses. In *CIKM*, pages 37–46, 2011.
19. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.
20. N. Nakashole, G. Weikum, and F. Suchanek. PATTY: a taxonomy of relational patterns with semantic types. In *EMNLP*, 2012.
21. J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.
22. S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *TREC*, volume Special Publication 500-225, pages 109–126. National Institute of Standards and Technology (NIST), 1994.
23. F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, pages 697–706, 2007.
24. I. Vulić and M.-F. Moens. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *SIGIR*, pages 363–372, 2015.
25. G. Zhou, T. He, J. Zhao, and P. Hu. Learning continuous word embedding with metadata for question retrieval in community question answering. In *ACL*, volume 1, pages 250–259, 2015.