4-2021

NewsLink: Empowering intuitive news search with knowledge graphs

Yueji YANG

Yuchen LI
Singapore Management University, yuchenli@smu.edu.sg

Anthony TUNG

Citation

1

2021 IEEE 37th International Conference on Data Engineering (ICDE) | 978-1-7281-9184-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICDE51399.2021.00081

NewsLink: Empowering Intuitive News Search with Knowledge Graphs

Yueji Yang
National University of Singapore
yueji@comp.nus.edu.sg

Yuchen Li
Singapore Management University
yuchenli@smu.edu.sg

Anthony K. H. Tung
National University of Singapore
atung@comp.nus.edu.sg

***Abstract*—News search tools help end users to identify relevant news stories. However, existing search approaches often carry out in a "black-box" process. There is little intuition that helps users understand how the results are related to the query. In this paper, we propose a novel news search framework, called NewsLink, to empower intuitive news search by using relationship paths discovered from open Knowledge Graphs (KGs). Specifically, NewsLink embeds both a query and news documents to subgraphs, called *subgraph embeddings*, in the KG. Their embeddings' overlap induces relationship paths between the involving entities. Two major advantages are obtained by incorporating subgraph embeddings into search. First, they enrich the search context, leading to robust results. Second, the relationship paths linking entities inter and intra news documents can help users better understand and digest the results for the given query. Through both human and automatic evaluations, we verify that NewsLink can help users understand the result-to-query relatedness, while its search quality is robust and outperforms many established search approaches, including Apache Lucene and a KG-powered query expansion approach, as well as popular deep learning models, Sentence-BERT (SBERT) and Doc2Vec.**

I. Introduction

News search is important for end users to navigate through massive contents generated by online media [1], [2]. For instance, journalists issue queries to discover relevant news articles for drafting newsworthy stories [3], [4], [5], [6] and intelligent agents unveil hidden relationships between involving entities [7], [8]. At present, most search approaches focus on improving search quality with many added features and complex models [9], [10], [11], [12], [13]. However, the search process has often been conducted in a black box and there lacks an intuitive way to help users understand and digest the search results. For example, the results returned by Google News Search are a list of ranked websites. There are no *intuitive* clues that explain the relatedness between the query and the results, other than the matching keywords. Therefore, in this paper, we propose a novel news search framework, NewsLink, that empowers intuitive news search by utilizing open Knowledge Graphs (KGs), e.g., Freebase [14] and Wikidata [15].

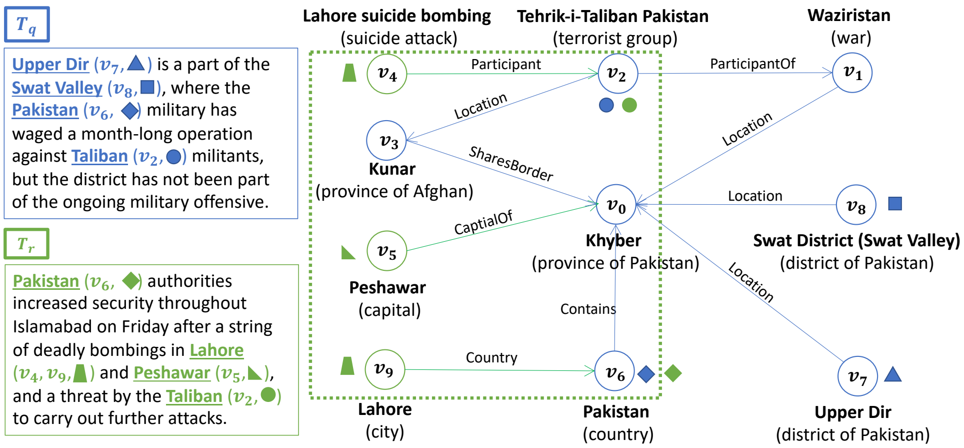
To achieve the above goal, NewsLink first embeds both the query and the news documents to subgraphs, called *subgraph embeddings*, in a KG. The overlap of their subgraph embeddings results in relationship paths that link together entities inter and intra the news documents. We show that this overlap information can be used to improve the *search robustness*. This is because the subgraph embeddings containing additional information from the KG can enrich the search *context* of the query. For instance, the enriched context mitigates the problem of *vocabulary mismatch* [16]. Moreover, the relationship paths between the query and the result are concrete evidences for their relatedness. Such evidences are intuitive and crucial for interpreting the black-box search process and enable easy user comprehension. We illustrate NewsLink more specifically by Example 1 as follows.

Example 1. *Figure 1 contains two pieces of news $\{T_q, T_r\}$. T_r is a search result for the query T_q by* NewsLink*.*

Towards Improving Robustness*. The typical search approaches are mainly based on keyword matching [17]. Thus, they can only capture the matched entities in the texts, e.g. v_2 and v_6 in Table I. However, for unmatched entities (see Table I), it is challenging to identify the relevance between them. This can lead to unsatisfying search results.* NewsLink *enriches the search context with the induced entities that are found from open KGs based on all entities in the text. The matching of these induced entities can thus reflect the relevance between those unmatched entities, e.g. Khyber v_0 and Kunar v_3 are induced entities for T_q and T_r (last column in Table I). The overlap of the induced entities adds up to the confidence of the relevance between T_q and T_r during search, leading to improved robustness.*

Towards Result Understanding*. In this example, the subgraph embeddings mainly reflect geographical relationships between places mentioned in the text, e.g. almost all the places are in or near Khyber (v_0), a province of Pakistan, that is missing in the original text. Such background information is crucial for users to get a big picture of these news stories. In addition, as we mentioned above,* NewsLink *uses the overlap of induced entities to improve search quality. This overlap also results in relationship paths that can link unmatched entities. These intuitive relationship paths help users understand and digest the information while browsing the results (see Table II for an example). This is a desired feature of explainability, which is missing from deep learning based search approaches, e.g.* Doc2Vec *[9] and SBERT [10]. Although they may capture the semantic similarities of unmatched entities by comparing them in high dimensional spaces, they provide no intuitive clues that help explain their relevance. In contrast,*

978-1-7281-9184-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDE51399.2021.00081



Upper Dir (v_7, ▲) is a part of the Swat Valley (v_8, ■), where the Pakistan (v_6, ◆) military has waged a month-long operation against Taliban (v_2, ●) militants, but the district has not been part of the ongoing military offensive.

Pakistan (v_6, ◆) authorities increased security throughout Islamabad on Friday after a string of deadly bombings in Lahore (v_4, v_9, ⏢) and Peshawar (v_5, ◣), and a threat by the Taliban (v_2, ●) to carry out further attacks.

Fig. 1: Subgraph embeddings by NewsLink. T_r is the result for T_q by NewsLink. The subgraph embedding of T_q is colored in blue, and that of T_r is shown in the green dotted box. Thus, the blue part in the dotted box is the overlap between T_q and T_r. Table I shows summaries of the news stories.

TABLE I: Summaries of news stories and different types of entities during search.

News id	News Summary	Matched entities (in text)	Unmatched entities (in text)	Example induced entities (in embedding)
T_q	Military conflicts between Pakistan and Taliban.	Taliban (v_2), Pakistan (v_6)	Upper Dir (v_7), Swat Valley (v_8)	Khyber (v_0), Kunar (v_3), Waziristan (v_1)
T_r	Bombing attack by Taliban in Pakistan.	Taliban (v_2), Pakistan (v_6)	Lahore (v_4, v_9), Peshawar (v_5)	Khyber (v_0), Kunar (v_3)

TABLE II: Examples of relationship paths linking entities between news texts.

Entity from T_q	Entity from T_r	Relationship Path
Upper Dir (v_7)	Peshawar (v_5)	$v_7 \rightarrow v_0 \leftarrow v_5$
Swat Valley (v_8)	Lahore (v_9)	$v_8 \rightarrow v_0 \leftarrow v_6 \leftarrow v_9$

NewsLink *overcomes this problem by providing users with explicit and intuitive relationship paths found in open KGs.*

However, it remains unclear how to extract a reasonable subgraph embedding from a KG for a news document. The processing can be divided into two steps. First, entities are recognized from the news document. Second, the subgraph embedding is found based on the recognized entities. The first step can be handled mainly by existing NLP pipelines [18], [19]. Whereas, the second step is more challenging, which we discuss below. We summarize two desired properties of a subgraph embedding: *coverage* for sufficiently enhancing search contexts and *compactness* for ensuring strong entity relationships. In other words, a good subgraph embedding should *cover* sufficient relevant information while still keeping the nodes within the embedding *compactly* connected. To extract a subgraph for the recognized entities, the existing models include Group Steiner Tree (GST) [20] and Lowest Common Ancestor (LCA) [21]. Nevertheless, these models are focused on *compactness* to find a compact subgraph representation, overlooking the *coverage* property which is essential for improving the search quality. Furthermore, the GST problem is NP-hard [22] and the LCA model only works for tree structured data. Therefore, we propose a novel and tractable model, called *Lowest Common Ancestor Graph* denoted as G^*, by relaxing the LCA model in two aspects. First, G^* can work in general KGs. Second, G^* allows multiple paths between an entity node and the common ancestor node, e.g. two paths from v_2 to v_0 in Figure 1. Hence, G^* can cover more information than a tree representation. As a result, our proposed model can lead to subgraph embeddings that cover sufficient information while still remaining compact to reflect strong entity relationships. To efficiently find a G^* from a large KG, we design a novel algorithm that works with three procedures. The first two procedures run in a loop, namely *path enumeration* and *candidate collection*. These two procedures end immediately after collecting enough candidate subgraph embeddings. The third procedure, *compactness sorting*, ranks the candidates and outputs the optimal subgraph embedding.

Our **contributions** are summarized as follows.

- We propose and implement a novel news search framework, called NewsLink, which extracts subgraph embeddings (from a KG) to improve the search robustness. The subgraph embeddings can provide evidences for the relatedness between the query and the result to help users digest the information. To the best of our knowledge, this is the first work on news search with subgraph embeddings from open KGs.
- We propose a new subgraph embedding model, called *Lowest Common Ancestor Graph*, which aims to cover sufficient relevant information while still keeping the subgraph compactly connected. We also devise efficient

algorithms to find the optimal subgraph embeddings.

- We conduct experiments on real-world news datasets to showcase the superiority of NewsLink. The search quality of NewsLink can outperform a set of established methods, including Apache Lucene [23] and a KG-powered query expansion approach [24], as well as deep learning based approaches, like SBERT [10] and Doc2Vec [9]. Furthermore, through both case study and user study, we demonstrate the distinguishing feature of NewsLink in helping users understand the results.

The remaining part of this paper is organized as follows. We discuss the related work in Section II. Section III introduces the architecture of NewsLink. Then, Section IV, Section V and Section VI show the details of the components within NewsLink. Section VII reports the experiment results, followed by conclusions in Section VIII.

II. Related Work

In this section, we discuss several related research areas.

Query Expansion (QE). QE techniques improve search quality by expanding queries with relevant terms. There are two popular QE techniques: expansion using retrieved documents and expansion from external resources. For the former technique, the most famous method is Pseudo Relevance Feedback (PRF). It expands the original query with terms from the top ranked documents (w.r.t. the original query). There are many works fall into this category, to name a few [25], [26], [27], [28], [29]. NewsLink is more related to the second kind of QE techniques that relies on external information. The approaches of this category [30], [11], [31], [12], [13] extract various features or terms from a KG, such as entity labels, descriptions, types and so on. Nonetheless, as pointed out by Xiong et al. [24], most QE techniques are 'high risk / high reward' since they often negatively impact as many queries as they improve. The expanded terms may contain noise, especially when the terms are from noisy resources like open KGs. In particular, [24] uses the terms from the descriptions of the linked object in KG for QE combined with a PRF mechanism. We note that none of the above techniques consider extracting subgraph embeddings that reflect explicit relationships between entities. Therefore, they do not provide any intuitive clues that help users understand the results.

Subgraph Extraction (SE). NewsLink embeds a news document by extracting relevant subgraphs for its entities from the KG. Two widely-used subgraph extraction models that are relevant to our problem are Group Steiner Tree (GST) [20] and Lowest Common Ancestor (LCA) [21] models. They are used in many applications, e.g. keyword search on graph-structured data [32], [33], [22], [34] and tree-structured data [35], [36], [37].The GST problem [20] aims at finding a minimum-weight connected subtree which covers all the input labels. Nonetheless, the problem is NP-Hard and cannot be approximated by a constant ratio in polynomial time [22]. In comparison, although the LCA model is more computationally efficient, it is designed for tree-structured data rather than general KGs. Moreover, both GST and LCA models are aimed at returning a concise tree that succinctly answers a query, overlooking the *coverage* property as desired by NewsLink for improving search robustness.

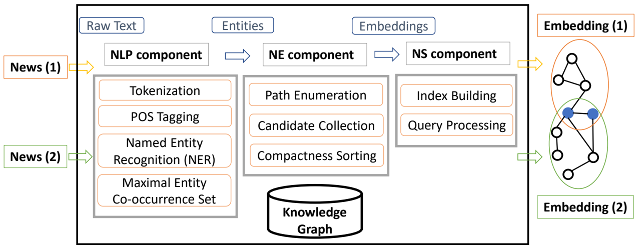


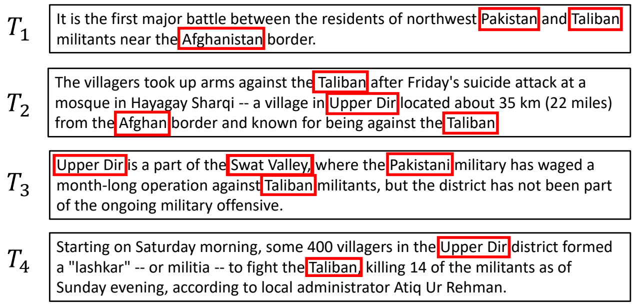
Fig. 2: The Architecture Overview of NewsLink.

Document Similarity Search (DSS). DSS systems are widely used by major search engines. As introduced in [38], DSS systems often take as input a piece of text and return top-k results according to some similarity metrics. Various models for calculating similarity have been proposed, such as Vector Space Model [17], Probabilistic Relevance Model [39] and Latent Dirichlet Allocation [40]. On top of these models, there are established DSS systems like ElasticSearch [41] and Apache Lucene library [23]. In addition to these traditional methods, Doc2Vec [9] is a deep learning model that maps a text segment to a high dimensional space. It uses the skip-gram model [42] to capture the word co-occurrences and achieves competitive search result as reported in [43], [9]. Moreover, SBERT [10] modifies the pre-trained BERT network [44] by using siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine similarity. However, these typical search approaches focus on search quality while overlooking the need for helping users understand the results. NewsLink overcomes this by extracting subgraph embeddings from a KG to provide intuitive clues for explaining the relatedness between queries and results.

News Search and Recommendation (NSR). Most news search methods are based on general text similarity search, i.e. DSS discussed above. Li et al. [45] propose methods to search news with images and texts jointly. Moreover, [46] studies temporally sensitive word embedding for news search. In addition, Abel et al. [47], [48] propose to use tweets for user modeling and recommendation. In comparison, NewsLink focuses on news search task where users' preferences are not known. In particular, none of the above work considers using subgraph embeddings from KGs to improve search robustness.

III. Architecture Overview

In this paper, we study the news search problem. Given a text query, e.g., news headings or paragraphs, we return top-k most similar and relevant news documents from a corpus. In the following, we introduce the architecture overview of NewsLink, as shown in Figure 2. There are three major components in NewsLink: the Natural Language Processing



T_1 It is the first major battle between the residents of northwest Pakistan and Taliban militants near the Afghanistan border.

T_2 The villagers took up arms against the Taliban after Friday's suicide attack at a mosque in Hayagay Sharqi -- a village in Upper Dir located about 35 km (22 miles) from the Afghan border and known for being against the Taliban

T_3 Upper Dir is a part of the Swat Valley, where the Pakistani military has waged a month-long operation against Taliban militants, but the district has not been part of the ongoing military offensive.

T_4 Starting on Saturday morning, some 400 villagers in the Upper Dir district formed a "lashkar" -- or militia -- to fight the Taliban, killing 14 of the militants as of Sunday evening, according to local administrator Atiq Ur Rehman.

Fig. 3: Four *news segments* from a news document with named entities recognized, shown in red boxes.

(NLP) component, the News Embedding (NE) component and the News Search (NS) component. In this paper, we focus on the NE and NS component since the NLP component is implemented mainly with existing tool chains. Nevertheless, the NLP component is crucial for ensuring result quality, we thus provide some details for reproducible requirements.

The NLP component (Section IV). The NLP component includes a series of standard text pre-processing, e.g. tokenization, POS tagging, and named entity recognition (NER). The input is a news document and the output comprises groups of entities identified from the input. More specifically, after NER, we divide the identified entities into different groups to preserve their co-occurring relationships in the text. The news text corresponds to one entity group is referred to as a *news segment*. For instance, entities occurring in one sentence can be collected into a group. Such a group of entities has a stronger relationship with each other, and it is likely there also exists a subgraph embedding that closely connects them in the KG. Therefore, we find one subgraph embedding for every co-occurring entity group. Nonetheless, the number of entity groups to be processed may be large especially when embedding a corpus. This can cause efficiency issues especially when processing a large news corpus. Given a news document, we formalize the entity group representation (i.e. *maximal entity co-occurrence set*) which helps to reduce the number of entity groups necessary to extract subgraph embeddings. The details are in Section IV. Figure 3 shows an example where four groups of entities (w.r.t. four *news segments*) are identified from a news document.

The NE component (Section V). The NE component handles one group of co-occurring entities at a time for a *news segment*, which has been processed the NLP component. Given a group of the identified named entities, the NE component finds a *connected* subgraph embedding from the KG. Then, the embedding of a news document is derived as the union of the embeddings for all its corresponding *news segments*. The proposed model and algorithm of the NE component are detailed in Section V.

The NS component (Section VI). The NS component mainly has two functionalities, i.e. *index building* and *query processing* as shown in Figure 2. Based on the embeddings extracted for all news documents from a corpus, we build retrieval indexes, which are later used for query processing. One design feature of NEWSLINK is that it is compatible with typical search approaches in two ways. First, NEWSLINK using subgraph embeddings can be based on the typical term-weighting (e.g. TF-IDF) and scoring functions (e.g. cosine similarity) that are widely used in VSM [17]. Second, the search by NEWSLINK can also incorporate textual information from the news document in addition to the subgraph embeddings. This compatibility makes it easy to integrate NewsLink with existing search systems.

IV. THE NLP COMPONENT

In this section, we briefly describe how to concisely represent the identified groups of entities for each news document during the processing of the NLP component. As mentioned in Section III, every news document is segmented into small and semantic-consistent pieces, e.g., news segments. The entities co-occurring in one news segment are expected to have a close semantic relationship. However, the identified entity set of one news segment may be a subset of another. In this case, we only need to embed the news segment that gives the larger set of entities. In Definition 1, we formalize this idea with *maximal entity co-occurrence set*. Specifically, given entity sets (i.e. groups) identified for all news segments, we only preserve those that are not proper subsets of any other sets. If there are equivalent entity sets, then we only preserve one of them.

Definition 1 (Maximal Entity Co-occurrence Set)**.** *Given all identified entity sets* $U = \{L_1, \ldots, L_n\}$*, where* $L_i = \{l_1^i, \ldots, l_t^i\}$ *denotes* t *entities identified from a news segment* T_i*. The maximal entity co-occurrence set given* U *is defined as* $U_m \subseteq U$*, s.t.* $L_i \in U_m$ *iff* $L_i \not\subset L_j$ *for* $\forall L_j \in U$*.*

Example 2. *In Figure 3, four news segments* $\{T_1, T_2, T_3, T_4\}$ *are identified from a news document. Accordingly, the identified entities are* $L_1 = \{Pakistan, Taliban, Afghan\}$*,* $L_2 = \{Upper\ Dir, Afghan, Taliban\}$*,* $L_3 = \{Upper\ Dir, Swat\ Valley, Pakistan, Taliban\}$*,* $L_4 = \{Upper\ Dir, Taliban\}$*. Thus, all identified entity sets are included as* $U = \{L_1, L_2, L_3, L_4\}$*. Since* $L_4 \subset L_2$*, the maximal entity co-occcurrence set* $U_m = \{L_1, L_2, L_3\}$ *with* L_4 *ruled out. In the end, only three sets of entities are passed down to the NE component for subgraph embedding processing.*

During the entity recognition process, we consider all entity types except those representing numbers or quantities. The considered types include person, nationality, religious or political groups, facilities, organization, GPE (Geo-Political Entity), location, product, event, work of art, law and language. The matching from entity label to entity nodes in the KG follows an exact matching manner. The matching ratio is over 96% on average per news segment, which we report in Section VII.

V. THE NE COMPONENT

The goal of the NE component is to discover a subgraph embedding for each group of entity within the *maximal entity* *co-occurrence set* from the NLP component. Thus, we focus on elaborating the processing for one entity group. In this section, we propose a novel subgraph embedding model that mitigates the major limitations of existing approaches (discussed in Section II) from two aspects. First, the embedding model should cover sufficient relevant information. Second, the model should be able to work for general graphs. To this end, we generalize the LCA model to work in KGs. Specifically, we allow multiple paths to be preserved between an entity label and the root (i.e. the common ancestor), leading to rich coverage of relevant information. This can be seen as increasing the "width" of the subgraph embedding. In addition, to ensure entities are strongly connected, we define *compactness order* that helps us to determine the most *compact* subgraph embedding in terms of a small "depth". In the following, we first develop a model called *Common Ancestor Graph G*. The optimal G is then defined as *Lowest Common Ancestor Graph*, referred to as G^*. We will use G^* as the subgraph embedding model for each news segment. Given a document with multiple entity groups identified, we take the union of all G^* as the final document subgraph embedding. All frequently used notations are summarized in Table III.

A. The Proposed Subgraph Embedding Model

W.L.O.G., we assume the KG, denoted by $\mathcal{K}(\mathcal{V}, \mathcal{R})$, is connected, labeled and weighted. $\mathcal{V}$ and $\mathcal{R}$ denote the sets of entity nodes and relationship edges, respectively. Given an entity l, or equivalently entity label, it is mapped to a set of nodes $S(l)$ from $\mathcal{K}$ whose labels contain l through exact string matching.

Example 3. *Taking Figure 1 and T_q as an example,* $L = \{Upper\ Dir(l_1), Swat\ Valley(l_2), Pakistan(l_3), Taliban(l_4)$ *We have* $v_7 \in S(l_1), v_8 \in S(l_2), v_6 \in S(l_3),\ and\ v_2 \in S(l_4).\}$.

To enhance the graph connectivity, we add a reversed edge for each original relationship edge. As a result, $\mathcal{K}$ is bi-directed. Note that our definitions apply to general directed graphs, and we only consider simple (directed) paths with no cycles during search.

Distance. The distance from node u to node v is defined as the shortest path weight, denoted as $\mathcal{D}(u, v)$. The set of shortest paths with weight w is defined as $P(u \to v, w)$. Since the graph is bi-directed, the distance is symmetric between two nodes, i.e. $\mathcal{D}(u, v) = \mathcal{D}(v, u)$. For example, in Figure 1, the distance between v_2 and v_0 is $\mathcal{D}(v_2, v_0) = 2$ and $P(v_2 \to v_0, 2) = \{v_2 \to v_3 \to v_0, v_2 \to v_1 \to v_0\}$. For illustrative purpose, we assume the edge weights are all 1.

Pseudo Ancestor. A node u is a pseudo ancestor of v, denoted as $u \lhd v$,if and only if there is a directed path from v to u. Note that $\forall u \in \mathcal{V}, u \lhd u$. For example, in Figure 1, v_0 is the pseudo ancestor of any other node. Note that pseudo ancestor is different from the typical definition of ancestor in trees or acyclic directed graphs. In our definition, two different nodes can be pseudo ancestors of each other. Unless mentioned, we use "ancestor" to refer to "pseudo ancestor" for ease of presentation in this paper.

TABLE III: Notations.

Notations	Meanings
$\mathcal{K}(\mathcal{V}, \mathcal{R})$	The knowledge graph.
$l, S(l)$	Entity label and the set of nodes containing that label.
$\mathcal{D}(u/l, v)$	Distance from node u or label l to v.
$P(u/l \to v, w)$	All shortest paths from node u or label l to v with weight w.
$G_r(L), G_r, G$	A common ancestor graph rooted at r for labels in L.
$d(G)$	The depth of G.
G^*	The lowest (optimal) common ancestor graph, i.e., the optimal G.

Definition 2 (Entity-Node Distance)**.** *We define the distance from an entity with label l to a node v as* $\mathcal{D}(l, v) = \min\limits_{u \in S(l)} \mathcal{D}(u, v)$.

For instance, in Example 3, $\mathcal{D}(l_4, v_0) = 2$. In addition, we use $P(l \to v, \mathcal{D}(l, v))$ from Equation 1, or simply $P(l \to v, \mathcal{D})$, to denote the set of all shortest paths from l to v. The shortest paths lead to close connections that best reflect the strong relationships between l and v.

$$P(l \to v, \mathcal{D}(l, v)) = \bigcup_{u \in S(l)} P(u \to v, \mathcal{D}(l, v)) \qquad (1)$$

Example 4. *Continuing with Example 3, and taking l_4 and v_0 as an example (see Figure 1), $P(l_4 \to v_0, \mathcal{D})$ will only be the set $P(v_2 \to v_0, 2)$ as there is no other node containing Taliban with distance 2 to v_0.*

Definition 3 (Common Ancestor Graph)**.** *Given labels $L = \{l_1, \ldots, l_m\}$, a node r, called root, is a common ancestor of L iff $\forall i \in \{1, \ldots, m\}$ s.t. $\exists v_i \in S(l_i), r \lhd v_i$. The Common Ancestor Graph rooted at r is defined as $G_r(L) = \bigcup\limits_{i=1}^{m} P(l_i \to r, \mathcal{D})$ with depth $d(G_r) = \max\limits_{l_i \in L} \mathcal{D}(l_i, r)$.*

In Definition 3, given a set of entity labels, we define a common ancestor graph G rooted at r (i.e. their common ancestor) to cover all shortest paths from every label l_i to r. Multiple shortest paths enrich the resulting subgraphs for increasing the "width" of the embedding. For instance, in Figure 1, there are two common ancestor graphs rooted at v_0 corresponding to T_q and T_r respectively. Furthermore, it is clear that there are two paths from l_4 (Taliban) to the ancestor v_0 (Khyber). This helps improve the coverage of the information to be extracted and used for search purpose. Moreover, $d(G_{v_0}) = 2$ w.r.t. T_q. Note that $\mathcal{K}$ is bi-directed but a common ancestor graph consists of uni-directed paths ending at the common ancestor.

In fact, any node in $\mathcal{K}$ can locate a G for a given L when we assume $\mathcal{K}$ is connected and bi-directed. To select the best common ancestor graph, we need to identify the optimal G, that *compactly* connects the input entities by restricting them in a small "depth". Within a small "depth", all the entity nodes are connected more closely and strongly. To this end, we define the *compactness order* (Definition 4) to help us identify the optimal G. Such an optimal common ancestor graph is defined as the *Lowest Common Ancestor Graph* in Definition 5. In

Algorithm 1: G^* Search Algorithm

```
Input: The set of entity labels $L = \{l_1, \dots, l_m\}$ identified from a
       news segment, the adjacency lists of $\mathcal{K}$.
Output: The lowest common ancestor graph $G^*$.
1  foreach $l_i \in L$ do
2      Initialize a distance min-priority queue $F_i$ measured by $\mathcal{D}(l_i, \cdot)$;
3      foreach $v \in S(l_i)$ do
4          $\mathcal{D}(l_i, v) \leftarrow 0$;
5          Push $(v, \mathcal{D}(l_i, v))$ into $F_i$;
6  $Candidate \leftarrow \emptyset$;          // Container of candidates.
7  $min\_depth = \infty$;          // Depth of collected $G'$s.
8  while Not Timeout do
9      $v_f, \mathcal{D}(l_i, v_f) \leftarrow$PathEnumeration($F$); // $F$ denotes
         all $F_i$.
10     CandidateCollection($v_f, Candidate, min\_depth$);
       /* Termination test by condition $C_1$ & $C_2$
          (Section V-B).                                  */
11     $\mathcal{D}'_{min} \leftarrow \min_{(v,\mathcal{D})=F_i.top(), l_i \in L} \mathcal{D}$;
12     if $Candidate \neq \emptyset$ and $min\_depth < \mathcal{D}'_{min}$ then
13         break;
14 $G^* \leftarrow$ sort $Candiate$ by Compactness Order (Definition 4);
15 return $G^*$;
```

Algorithm 2: The *PathEnumeration* procedure

```
Procedure PathEnumeration($F$)
    $l_i, v_f = \arg\min_{(v_f,\mathcal{D})=F_i.top(), l_i \in L} \mathcal{D}$;   // By Equation 2.
    $F_i.pop()$;   // $v_f$ marked as processed w.r.t. $l_i$
      and $F_i$.
    foreach $v_n \in Neighbors(v_f)$ do
        if $v_n \in F_i$ then
            adjust $\mathcal{D}(l_i, v_n)$ to the smaller value with and without
              passing $v_f$;
        else
            push $(v_n, \mathcal{D}(l_i, v_n))$ to $F_i$;
    return $v_f, \mathcal{D}(l_i, v_f)$;
```

Definition 4, for a $G_r(L)$ rooted at r, let $\mathcal{D}_{(1)}, \dots, \mathcal{D}_{(m)}$ denote the distance values from each $l_i \in L$ to r *sorted in a descending order.*

Definition 4 (Compactness Order). *Given a set of entity labels $L = \{l_1, \dots, l_m\}$, for $G_r(L)$ and $G_{r'}(L)$ with different roots, their compactness order is as follows.*

1) *$G_r = G_{r'}$ iff $\mathcal{D}_{(i)} = \mathcal{D}'_{(i)}$ for $1 \leq i \leq m$,*
2) *$G_r < G_{r'}$ iff $\exists k, 1 \leq k \leq m, \mathcal{D}_{(i)} = \mathcal{D}'_{(i)}$ for $i < k, \mathcal{D}_{(k)} < \mathcal{D}'_{(k)}$.*

To illustrate Definition 4, in Figure 1 (Example 3), the distance is ordered as $\{\mathcal{D}_{(1)} = \mathcal{D}(l_4, v_0) = 2, \mathcal{D}_{(2)} = \mathcal{D}(l_1, v_0) = 1, \mathcal{D}_{(3)} = \mathcal{D}(l_2, v_0) = 1, \mathcal{D}_{(4)} = \mathcal{D}(l_3, v_0) = 1\}$. Suppose there is another G_u rooted at u with distance ordered as $\{\mathcal{D}(l_1, u) = 2, \mathcal{D}(l_4, u) = 2, \mathcal{D}(l_3, u) = 1, \mathcal{D}(l_2, u) = 1\}$. G_{v_0} is considered more compact than G_u in terms of "depth" because the second (largest) distance of G_{v_0} is smaller.

Definition 5 (Lowest Common Ancestor Graph). *Given $L = \{l_1, \dots, l_m\}$, the Lowest Common Ancestor Graph, referred to as $G^*_r(L)$, is defined as the common ancestor graph $G_r(L)$ s.t. $\nexists G_{r'}, G_{r'}(L) < G_r(L)$.*

Next, we explain how our formulations contribute to the *compactness* and *coverage* properties as desired by subgraph embeddings. The benefit of the *compactness order* is that it enhances the structural compactness of the derived *lowest common ancestor graph* G^*, which will be used as the subgraph embedding model. This is reflected in Lemma 1 and 2. The *compactness* property of G^* forces entity nodes to closely connect with each other, capturing strong relationships.

Lemma 1. *Given L, G^* has the smallest depth $d(G^*)$ over all possible G's.*

Proof of Lemma 1. Given a set of entity labels L, suppose there are two common ancestor graph G_r and $G_{r'}$. According to the compactness order, $d(G_r) < d(G_{r'}) \Rightarrow G_r < G_{r'}$ must holds. This can easily be verified based on Definition 4. Suppose the lowest common ancestor graph is G^*_r, if there is another common ancestor graph G_r s.t. $d(G_r) < d(G^*_r)$. Then, $G_r < G^*_r$, which means G^*_r is not the lowest common ancestor graph. By contradiction, we can derive Lemma 1. □

Lemma 2. *Given L, the distance between any two nodes in G^* is upper bounded by $2 \cdot d(G^*)$.*

Proof of Lemma 2. Suppose G^*_r is the lowest common ancestor graph for entity labels L. For any two nodes v_1 and v_2, there is always a path between them through the root r. Note that we model the graph as bi-directed. Hence, the distance of a path is symmetric to its reversed version, i.e. $\mathcal{D}(v_1, r) = \mathcal{D}(r, v_2)$. Therefore, by following the path via r, we know $\mathcal{D}(v_1, v_2) \leq \mathcal{D}(v_1, r) + \mathcal{D}(r, v_2) \leq 2 \cdot d(G^*_r)$, where $d(G^*_r)$ is the largest leaf-root distance. □

In addition to the *compactness* property discussed above, the *coverage* property of a G^* lies in the fact that it preserves all shortest paths w.r.t. every entity label, i.e. $P(l \rightarrow r, \mathcal{D})$. In summary, the lowest common ancestor graph identifies a compactly connected subgraph while covering sufficient information. This information is further used to enhance search context, leading to improved search quality.

B. The Lowest Common Ancestor Graph Search Algorithm

In this section, we describe the algorithm that discovers the G^* for a group of entities with labels $L = \{l_1, \dots, l_m\}$. After obtaining source nodes $S(l_i)$ with mentions for each label, the search proceeds in three major procedures (Algorithm 1): (1) the *PathEnumeration* procedure (Algorithm 2); (2) the *CandidateCollection* procedure (Algorithm 3); and (3) *compactness sorting* (line 14 in Algorithm 1). The first two procedures execute in a loop: every time a path ending at a frontier v_f is enumerated, we check if v_f locates a candidate common ancestor graph G. After collecting enough candidates, the first two procedures terminate and the search enters the third procedure to find out the G^* from all candidates according to the *compactness order*.

In Algorithm 1, given entity labels $L = \{l_1, \dots, l_m\}$, we initialize a frontier queue F_i for every l_i, respectively. Every F_i is a min-priority queue based on the distance, i.e. the path weight, between l_i and a node v_i. *In the first procedure* (line 9 in Algorithm 1), we enumerate a new path

Algorithm 3: The *CandidateCollection* procedure

```
1 Procedure
  CandidateCollection($v_f$, $Candidate$, $min\_depth$)
2     $d \leftarrow -1$;
      /* If any label has not reach $v_f$, their
         distance is $\infty$.                              */
3     foreach $l_i \in L$ do
4         if $d < \mathcal{D}(l_i, v_f)$ then
5             $d \leftarrow \mathcal{D}(l_i, v_f)$;
6     if $d \neq \infty$ then
7         Insert $G$ rooted at $v_f$ into $Candidate$;
8         $min\_depth = d$;
```

that results in the overall smallest distance value across all labels. The selection of v and l_i is decided by Equation 2 at line 2 in Algorithm 2. This path enumeration process leads to monotonically increasing of distance values (see Lemma 3), which is essential for the algorithm correctness in Theorem 1.

$$l_i, v^* = \underset{v \in S(l_i), l_i \in L}{\arg\min}\ \mathcal{D}(l_i, v) \tag{2}$$

Lemma 3. *(Monotonicity of $\mathcal{D}(l_i, v_f)$) Given entity name labels L, $\mathcal{D}(l_i, v_f)$ is non-decreasing for all $l_i \in L$ w.r.t. the order of paths enumerated (line 9 in Algorithm 1).*

Proof of Lemma 3. Lemma 3 is derived from two steps as the expansion is based on Equation 2. First, for each label l_i, the enumerated path ending at a node v has an increasing distance value to l_i compared to previously enumerated nodes from l_i. Second, according to Equation 2, the path to be enumerated is the smallest over all labels $\forall l_i \in L$. Therefore, the path will have an overall monotonically increasing distance value with respect to the path enumeration ordering. □

During expansion, each $S(l_i)$ also propagates its entity label l_i to the frontier v. Then, *in the second procedure*, we check the frontier node v resulted from *path enumeration* whether it receives all labels in L so far (line 3-4 in Algorithm 3). If so, the frontier is identified as locating a candidate subgraph. The first and second procedures execute iteratively until the termination condition is satisfied. The termination condition guarantees that the G^* is already collected within the candidates and there is no need for further expansion. There are two conditions, namely C_1 and C_2, that must be satisfied simultaneously for termination. They are listed below.

C_1: At least one common ancestor graph has been found.

C_2: The distance value of the next frontier is larger than $d(G_r)$, where G_r is the first identified *common ancestor graph*.

These two conditions are checked at lines 11-13 in Algorithm 1. Line 11 calculates the distance value (i.e. weight) of the path to be expanded next. At line 12, $Candidate \neq \emptyset$ corresponds to C_1 and $min\_depth < \mathcal{D}$ checks C_2.

Lastly, *in the third procedure*, we apply the *compactness order* to sort all candidates, and finally generate the G^* for embedding the news segment. We omit the details as it is straightforward according to Definition 4. Theorem 1 shows the correctness.

Theorem 1. *Algorithm 1 correctly returns the lowest common ancestor graph.*

In Figure 4, we show a complete example subgraph embedding found for entity groups identified in Figure 3. The overlapped parts suggest entities appearing more frequent in the context. We will use such information during news search, which is detailed in the next section.

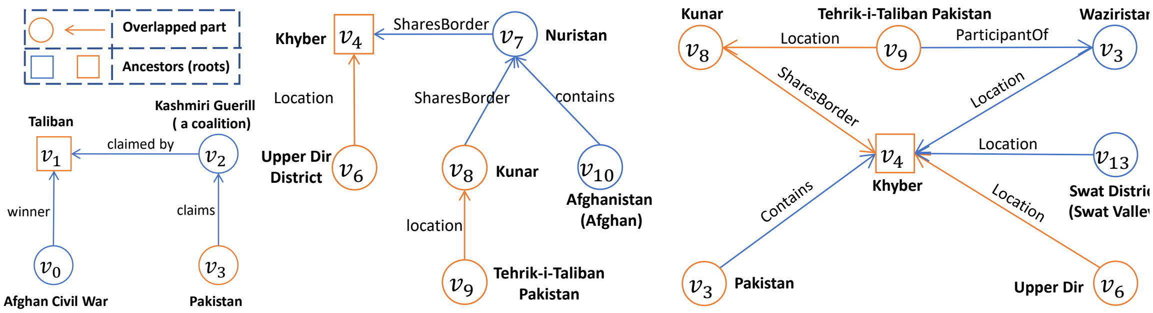
VI. The NS component

In Section V, we have shown how to find the G^* for an individual news segment. Given a news document $\overline{T}$, we find all embeddings $\overline{G^*} = \{G^*_1, \dots, G^*_k\}$ for all entity groups in its *maximal entity co-occurrence set* (see Figure 4 for an example). Then, in the NS component, we use these subgraph embeddings to improve the search quality. We handle the query and the news document in the same way. In the following, we introduce the NS component in terms of *index building* and *query processing*.

We make the NS component compatible and flexible in two ways. *First* (**scoring compatibility**), we incorporate TF-IDF term-weighting strategies to represent a news embedding $\overline{G^*}$ as a vector with weights for nodes. We refer to this representation as Bag-Of-Node (BON) model, analogous to Bag-Of-Word (BOW) model whose words (i.e. terms) are replaced by nodes in BON. For instance, in Figure 4, according to BON model, orange nodes will have a higher frequency within the news embedding. Combining their occurrences across the embeddings of the whole corpus, we can similarly apply a TF-IDF style weighting strategy to those nodes. Due to the scoring compatibility, the news search based on subgraph embeddings can benefit from many existing approaches and tools that are well-developed for BOW model. For *index building*, we separately build two inverted indexes for the news embeddings and the news documents of the corpus in the same way. Then, the *query processing* will be based on these indexes as in [17]. *Second* (**text compatibility**), based on BON model, the result scores from the embeddings and the document texts can be easily combined to improve the search robustness. This is because the relevance between embeddings of news documents can be measured the same as that between their texts with BOW model. Specifically, given $\overline{T}_q$ as a query and $\overline{T}_c$ as a candidate news document taken from the inverted indexes, the relevance is computed by Equation 3. Their embeddings are denoted by $\overline{G^*}_q$ and $\overline{G^*}_c$, respectively.

$$\mathcal{F}(\overline{T}_q, \overline{T}_c) = (1-\beta)\mathcal{F}_{BOW}(\overline{T}_q, \overline{T}_c) + \beta\mathcal{F}_{BON}(\overline{G^*}_q, \overline{G^*}_c) \tag{3}$$

where $\beta \in [0, 1]$, $\mathcal{F}_{BOW}(\overline{T}_q, \overline{T}_c)$ and $\mathcal{F}_{BON}(\overline{G^*}_q, \overline{G^*}_c)$ denote the similarity scoring based on BOW and BON models, respectively. With larger β, the search process relies more on the relationships among entities. In contrast, smaller β makes the search rely more on textual similarities between news documents. During *query processing*, we employ existing top-k ranking algorithms [49], [38] to retrieve the top-k news documents ranked by Equation 3.



(a) $G^*(L_1)$ (b) $G^*(L_2)$ (c) $G^*(L_3)$

Fig. 4: The subgraph embeddings output by the NE component w.r.t. Figure 3. The input is the maximal entity co-occurrence set $U_m = \{L_1, L_2, L_3\}$ extracted from news segments in Figure 3 by the NLP component. In particular, L_1 =$\{$*Pakistan, Taliban, Afghan*$\}$, L_2 =$\{$*Upper Dir, Afghan, Taliban*$\}$, and L_3 =$\{$*Upper Dir, Swat Valley, Pakistan, Taliban*$\}$. Overlapped nodes and edges of the three embeddings are in orange. Nodes in square shapes denote the *lowest common ancestors*, i.e. roots.

VII. Experiment Studies

In this section, we focus on the following questions.

- Can NewsLink produce competitive search results given a partial query with possible missing context? (Table IV and Section VII-C)
- How do end users think about NewsLink? (Figure 5 in Section VII-D.)
- What subgraph embeddings can NewsLink find, and how the embeddings help users to understand and digest the results? (Figure 6 in Section VII-E)
- How effective is the proposed subgraph embedding model of the NE component with varying parameter β of Equation 3, compared with the typical tree-based subgraph extraction model? (Table VII and Section VII-F)

In addition, we also validate that our algorithm works more efficiently than that of the tree-based model for implementing the NE component. The complementary efficiency results, including the running time breakdown for processing queries, are reported in Section VII-G.

A. Settings

1) Knowledge Graph: We use Wikidata KG [15] to embed the news documents. The Wikidata dump is publicly available, with over 30 million nodes and 135 million relationship edges.

2) News datasets: We collect two news datasets that are publicly available, *CNN* [1] with 92,580 news documents and *Kaggle* [2] with 90,130 news documents. The two datasets contain many news across many types such as sports, politics and entertainment. We crafted the evaluation task and queries as described in Section VII-B. For the collected news corpus, we filtered out all documents that we can not find a subgraph embedding. As a consequence, *CNN* contains 89,197 (96.3% of total) documents and *Kaggle* contains 82,182 (91.2% of total) documents.

3) Competitors: The competitors are described below.

- **NewsLink** (β): Our proposed approach for searching news documents. β is the weight for news embedding during search, introduced in Equation 3.
- **Doc2Vec** [9]: A deep learning model that transfers a text segment or a document to its vector representation. We use Gensim library[3] to train the model with 500 dimensions.
- **SBERT** [10]: A sentence embedding approach based on BERT. We use the published *pretrained* model, called bert-large-nli-mean-tokens (1024 dimension). We use the open-source code[4] provided by the authors of [10].
- **LDA** [40]: A topic model that represents a document as a mixture distributions of latent topics. We use PLDA [50] to train the model in parallel with 500 dimensions.
- **QEPRF** [24]: It uses terms from the description of linked entity nodes for query expansion with a Pseudo Relevance Feedback mechanism. We use its unsupervised version that outperforms the state-of-the-art as reported in [24].
- **Lucene** [23]: It is a VSM-based search library. The version we use is 7.7.0. We use BM25 [39] for term weighting with default settings of the library.

We randomly split each news dataset into training (80%), validation (10%) and testing (10%) data. Training data is used for training Doc2Vec and LDA models to be used as competitors. The trained models are used to infer vector representations of all documents. Validation data is used for tuning Doc2Vec and LDA models. The evaluation is done on testing data. The average entity matching ratio (the number of matched entities over that of identified entities) per test query is shown in Table V.

4) Platform and Implementation: The NLP component of NewsLink is implemented in Python 3.6. We choose to use spaCy [51] library with its pre-trained model. The NE component is implemented in C/C++ 4.8. It runs as a back-end server. We use every sentence as a *news segment* as it

[1] https://cs.nyu.edu/ kcho/DMQA/

[2] https://www.kaggle.com/snapcrack/all-the-news

[3] https://radimrehurek.com/gensim/models/doc2vec.html

[4] https://github.com/UKPLab/sentence-transformers

TABLE IV: Effectiveness of search results against popular approaches. The displayed scores correspond to largest-entity-density-query/randomly-selected-query.

	CNN					*Kaggle*				
	SIM@k			HIT@k		SIM@k			HIT@k	
	top-5	top-10	top-20	top-1	top-5	top-5	top-10	top-20	top-1	top-5
DOC2VEC	.839/.833	.839/.833	.840/.835	.333/.230	.545/.337	.919/.649	.914/.667	.911/.684	.439/.087	.495/.126
SBERT	.944/.941	.943/.940	.941/.938	.127/.103	.204/.172	.937/.932	.937/.933	.937/.933	.181/.149	.247/.208
LDA	.945/.942	.945/.942	.944/.941	.055/.046	.135/.109	.935/.933	.936/.935	.937/.936	.057/.045	.123/.099
QEPRF	.963/.952	.956/.946	.953/.941	.807/.793	.915/.914	.960/.940	.955/.934	.951/.931	.829/.822	.891/.894
Lucene	.964/.953	.958/.947	.954/.941	.807/.806	.917/.926	.963/.942	.958/.936	.954/.932	.831/.838	.895/.917
NEWSLINK (0.2)	**.966/.964**	**.959/.957**	**.954/.952**	**.876/.862**	**.972/.967**	**.965/.963**	**.959/.957**	**.955/.953**	**.910/.892**	**.966/.953**

TABLE V: Average entity matching ratio.

Test Query Set	Entity Matching Ratio
CNN	97.54%
Kaggle	96.49%

guarantees the semantic consistence of occurring entities. The NS component is implemented using Apache Lucene [23] of version 7.7.0. The scoring is based on BM25 [39] with default settings provided by Lucene. Note that there are other alternative tools for implementing NEWSLINK. For instance, the NLP component can also be based on Standford CoreNLP [18] and the NS component can be built on ElasticSearch [41]. We use a single machine with CentOS 7.0 and Intel(R) Xeon(R) Platinum 8170 CPU @ 2.1GHz (1 TB RAM).

B. Evaluation Task and Metric

Evaluation Task. We design a *Partial Query Similarity Search* task that mimics a real application scenario: given a partial news text such as a news heading, a paragraph or a tweet, to retrieve similar news documents. Specifically, we select a **query sentence** q from a **test document** Q. The query sentence has the largest entity density within Q so that it captures the most relevant information and context in Q. The entity density equals the number of entities over that of terms within the sentence. Then, we use q to query the entire news corpus to see whether an approach can retrieve top-k **results** $\{R_1, ..., R_k\}$ that are similar to Q. By using q instead of an entire test document, we hide the context information so that we can better investigate the robustness of an approach in dealing with queries missing context. To make the comparison fair, all the evaluations are also conducted on randomly selected query sentences from the test documents.

Note that different competitors use different search methods and process the same test queries. When evaluating the *similarity* of results, the complete *test document* Q and *results* R_i are converted into FastText [52] embedding vectors, which is a popular generic embedding approach. The evaluation score is calculated by the cosine similarity of the FastText vectors of Q and R_i.

Evaluation Metric. We adopt two metrics as follows, i.e. **SIM@k** and **HIT@k**. First, **SIM@k** shows the ability of an approach in retrieving similar documents with limited information in a query. We calculate the average *Cosine Similarity score* between Q and the top-k results $\{R_1, ..., R_k\}$. The results are further averaged over all test cases as in Equation 4.

$$SIM@k = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{k}\sum_{j=1}^{k} cosine\_sim(Q^{(i)}, R_j^{(i)}) \qquad (4)$$

where N is the number of test cases and $Q^{(i)}$ denotes the *query document* of the *query sentence* $q^{(i)}$. Accordingly, $R_j^{(i)}$ denotes the top-j result for $q^{(i)}$. Note that the cosine similarity is measured in the FastText embedding space. It is common to measure text similarity by their embedding vectors [52], [9], [53], [10], [42].

Second, **HIT@k** shows the percentage of test queries q's whose query documents Q's are returned within the top-k results, i.e. Q falls in the top-k results, given a tested competitor approach.

C. Overall Search Quality

From the results shown in Table IV, it can be seen that, with subgraph embeddings, NEWSLINK achieves very competitive results even though the queries only contain partial texts from the test document. Especially for HIT@k metric, NEWSLINK can recover most original query documents within a small top-k value compared to other methods including the keyword-based approach, Lucene, and the query expansion approach, QEPRF. This indicates the robustness of NEWSLINK in handling partial queries with possible missing context. In addition, there is a drop from using largest-entity-density query sentences to randomly chosen ones. This suggests that the former helps retrieve more relevant results since it captures more entities from the news.

The results of different β values are shown in Section VII-F. From Table IV, the overall best performance in terms of both SIM@k and HIT@k is consistently achieved by NEWSLINK (0.2). This is an inspiring result since it means that incorporating external information from the KG can improve text similarity search results. This finding is in line with many deep learning approaches on machine reading comprehension tasks [54], [55], where using KG as external training sources leads to better results.

As for embedding approaches including DOC2VEC, SBERT and LDA, their HIT@k is worse than BOW model based approaches. This is because BOW-based approaches find common words while paying less attention to the word semantics

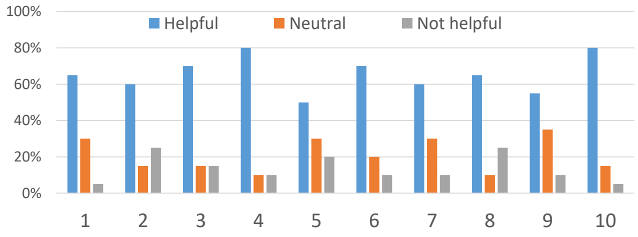


Fig. 5: Result of user study.

as those embedding approaches. This benefits them in recovering the query documents.

D. User study

We conduct a user study as shown in Figure 5. In order to better evaluate the effectiveness of the subgraph embeddings, we use the top retrieved results via subgraph embeddings only, i.e., $\beta = 1$. For each evaluation, participants are presented with a pair of news stories along with their subgraph embeddings. One news piece is the top result by using the other one as the query. We obtain ten different pairs of news pieces including the topics, such as military, politic and sport. We ask the participant a question: do you think the subgraph embedding information is helpful for understanding the news stories and their relatedness? Every participant chooses one of the three options: helpful, neutral, not helpful. We found 20 participants who are either undergraduate and graduate students from different disciplines. They are unaware of our approach. The case study (Section VII-E) is an example of the questions presented to participants during the user study.

Overall, more than half participants think that the subgraph embeddings are helpful for them to understand the results. We also collected some feedback from the participants for their choices. We are particularly interested in feedback from those choosing neutral or not helpful. We summarize three points that may help us improve NewsLink in the future. First, if participants already know the connections between the two news pieces, then the additional information does not help much. Second, if the additional information already appears in the news, then they are not helpful. Third, if the subgraph embeddings contain too much information, it overwhelms users and is thus not helpful. The first two points inspire us to explore relevant information that does not overlap too much with the original text. The last point inspires us to present only necessary path relationships and make the visualized parts of subgraph embeddings more concise for users to digest.

E. Case Study

In this section, we conduct a case study to reveal more insights about the distinguishing advantage of NewsLink for news search. The result R is retrieved with only subgraph embeddings used given the query Q, i.e., by NewsLink (1).

In Figure 6, both the query Q and the result R are about USA presidential election and the related politicians, including Hillary Clinton, Trump, Sanders. R is retrieved while it has few textual similarities with Q (only keyword *Sanders* is matched). Nonetheless, by NewsLink, we find that the two news statements are closely related to the US presidential election (v_0), which is never mentioned in the text of Q and R. In particular, the relationship paths through v_0 provide evidences that further *explain* the relatedness between Q and R. This feature of NewsLink is very attractive and useful as most search methods do not provide such intuitive explanations. NewsLink can identify these connections via subgraph embeddings from the KG. This enhances the capability of the typical keyword-based search approach [17]. In addition, in Table VI, we show some example relationship paths (from the KG) between entities with an intuitive description. It is obvious that the path information makes it easier for users to understand and digest the search results.

F. Varying parameters and the NE component

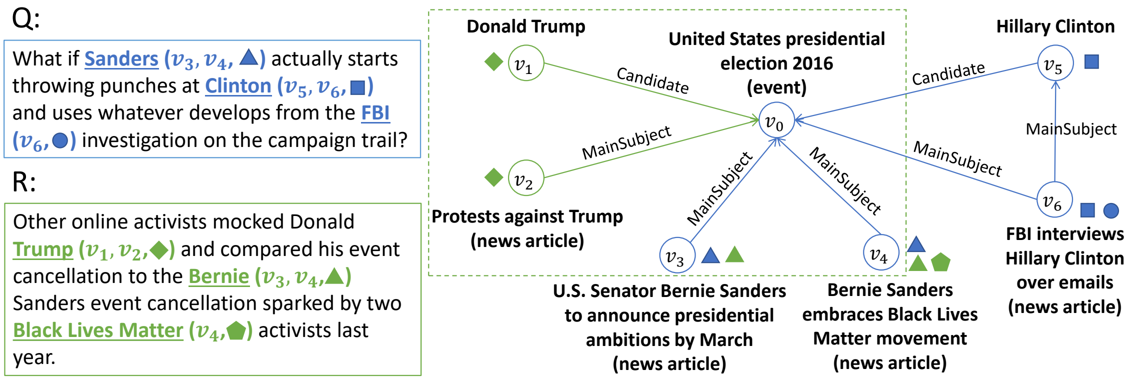
We replace the proposed subgraph embedding model by a tree-based one [33], referred to as *TreeEmb (β)*. It approximates Group Steiner Tree model. The resulting trees from TreeEmb are used as news embeddings. We aim to validate the model design in Section V. The *evaluation method* is the same as that in Section VII-B.

From the results shown in Table VII, we draw two major conclusions. First, the general idea of using extracted subgraphs to support news search task is effective. Even though we change the subgraph embedding algorithm inside the NE component, the search results from TreeEmb still outperform many competitors in Table IV. Second, our proposed *Lowest Common Ancestor Graph* model is better than the tree-based model. In addition, we note that even relying solely on subgraphs from the KG with $\beta = 1$, NewsLink can still produce competitive performance.

In comparison, NewsLink (1) from Table VII without textual information has a relatively lower score than $\beta < 1$. This is because NewsLink can retrieve news documents that may be considered less relevant by the FastText evaluation score. Since NewsLink is entity and relationship oriented, those seemingly less similar results are actually relevant based on relationship connections from the KG, which has been verified by the user study. This reflects that NewsLink can retrieve potentially similar and relevant results which may be missed by typical search approaches.

G. Time cost for corpus and query processing

In Figure 7, we show the average time cost for embedding all news documents in the corpus. The NE component costs the most time as compared to the other two components. On average, there are around 8 to 10 *news segments* per news document. Our proposed algorithm is signficantly faster because we can early terminate from the graph traversal and the subsequent compactness sorting is not a time-consuming step. In addition, we note that for processing corpus data, we can easily parallelize the process to speed up the processing. Given adequate computational resources, the NE component should not be a severe bottleneck of NewsLink. We also


Q:

What if Sanders (v_3, v_4, ▲) actually starts throwing punches at Clinton (v_5, v_6, ■) and uses whatever develops from the FBI (v_6, ●) investigation on the campaign trail?

R:

Other online activists mocked Donald Trump (v_1, v_2, ◆) and compared his event cancellation to the Bernie (v_3, v_4, ▲) Sanders event cancellation sparked by two Black Lives Matter (v_4, ⬟) activists last year.

Donald Trump
United States presidential election 2016 (event)
Hillary Clinton
v_1
v_0
v_5
Candidate
Candidate
MainSubject
MainSubject
MainSubject
MainSubject
MainSubject
v_2
v_6
Protests against Trump (news article)
v_3
v_4
FBI interviews Hillary Clinton over emails (news article)
U.S. Senator Bernie Sanders to announce presidential ambitions by March (news article)
Bernie Sanders embraces Black Lives Matter movement (news article)

Fig. 6: Case Study. Subgraph embedding of Q is rendered blue and that of R is in the green dotted box.

TABLE VI: Examples of relatedness evidence for Case Study 1 (Figure 6).

Identified Entity Pair	Relationship Path	Description
Clinton (v_5), Trump (v_1)	$v_5 \leftarrow v_0 \rightarrow v_1$ $v_6 \rightarrow v_5 \rightarrow v_0 \leftarrow v_1$	They are both candidates of the US presidential election 2016. FBI interviewing Email affects Clinton negatively and gains advantages for Trump in the election.
Clinton (v_5), FBI (v_6)	$v_5 \leftarrow v_6$	Negative Events that involve Clinton and FBI.
Sanders (v_3, v_4), Clinton (v_5)	$v_3 \rightarrow v_0 \leftarrow v_5$ $v_4 \rightarrow v_0 \leftarrow v_5$	Sanders announces presidential ambitions for the presidential election that Clinton also runs for. Sanders embraces the Black Lives Matter movement that is inline with his presidential ambitions.

TABLE VII: Effectiveness of search results compared with TreeEmb and different values of β (Equation 3).The displayed scores correspond to largest-entity-density-query/randomly-selected-query. $\beta = 0$ reduces to Lucene approach.

	CNN					*Kaggle*				
	SIM@k			HIT@k		SIM@k			HIT@k	
	top-5	top-10	top-20	top-1	top-5	top-5	top-10	top-20	top-1	top-5
NewsLink (1)	.951/.947	.944/.941	.939/.937	.661/.568	.863/.737	.951/.948	.945/.943	.941/.939	.669/.568	.816/.710
TreeEmb (1)	.871/.928	.815/.923	.750/.919	.701/.445	.887/.614	.891/.925	.853/.921	.813/.919	.684/.379	.826/.504
NewsLink (0.8)	.956/.956	.948/.949	.944/.945	.770/.741	.928/.920	.955/.955	.949/.950	.945/.946	.801/.767	.906/.893
TreeEmb (0.8)	.941/.942	.937/.935	.931/.929	.501/.788	.712/.936	.949/.938	.944/.932	.941/.929	.473/.787	.633/.900
NewsLink (0.5)	.963/**.962**	.957/.955	.952/.951	.853/.837	.967/.962	.962/.961	.956/.955	.953/.951	.889/.870	.958/.946
TreeEmb (0.5)	.952/.946	.947/.939	.941/.933	.782/.849	.922/.955	.960/.942	.954/.936	.950/.932	.777/.869	.884/.936
NewsLink (0.2)	**.966**/.947	**.959/.957**	**.954/.952**	**.876/.862**	**.972/.967**	**.965/.963**	**.959/.957**	**.955/.953**	**.910/.892**	**.966/.953**
TreeEmb (0.2)	.954/.947	.948/.940	.942/.933	.827/.850	.928/.952	.962/.943	.955/.936	.951/.932	.846/.875	.917/.936
$\beta = 0$	.964/.953	.958/.947	.954/.941	.807/.806	.917/.926	.963/.942	.958/.936	.954/.932	.831/.838	.895/.917

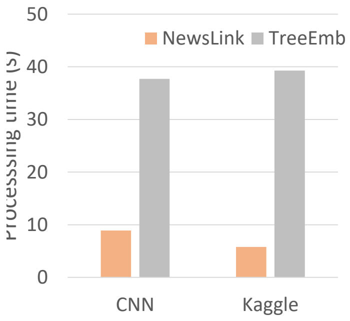


Fig. 7: Average embedding time per news document.

report the average query processing time for each component in Table VIII. The subgraph embedding process for the NE component costs the most time.

TABLE VIII: The query processing time breakdown for each component of NewsLink per test query.

	NLP time (s)	NE time (s)	NS time (s)
CNN	0.02	1.00	0.02
Kaggle	0.17	1.07	0.03

VIII. Conclusion

In this paper, we propose NewsLink to empower intuitive news search by utilizing open KGs. Through extensive experiments, we have demonstrated the efficiency and effectiveness of NewsLink. Our work pushes forward a step in retrieving similar and relevant news stories in the sense that the relevance is not just measured by the textual similarity. It can also be measured with relationship connections among entities from a real-world KG. In addition, the extracted subgraph embeddings can help users understand and digest the search results. Lastly, it is worth mentioning that the design of the NS component of NEWSLINK makes it easy to use and integrate with most existing search systems, such as ElasticSearch and Lucene.

ACKNOWLEDGEMENT

This work is supported in part by the International Cooperation Projects of Fujian in China (No. 2018|0016) and the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (Award No.: MOE2019-T2-2-065).

REFERENCES

[1] Y. Li, Z. Bao, G. Li, and K. Tan, "Real time personalized search on social networks," in *ICDE' 15*, 2015.

[2] Y. Li, D. Zhang, Z. Lan, and K. Tan, "Context-aware advertisement recommendation for high-speed social news feeding," in *ICDE' 16*, 2016.

[3] Z. Papacharissi and M. de Fatima Oliveira, "Affective news and networked publics: The rhythms of news storytelling on# egypt," *Journal of communication*, vol. 62, no. 2, 2012.

[4] Q. Fan, Y. Li, D. Zhang, and K.-L. Tan, "Discovering newsworthy themes from sequenced data: A step towards computational journalism," *TKDE*, vol. 29, no. 7, pp. 1398–1411, 2017.

[5] C. Martín Dancausa, D. Corney, A. Goker, and A. Macfarlane, "Mining newsworthy topics from social media," *Studies in Computational Intelligence*, vol. 602, 2013.

[6] H. Ceylan, I. Arapakis, P. Donmez, and M. Lalmas, "Automatically embedding newsworthy links to articles," in *CIKM '12*, 2012.

[7] M. S. Hossain, P. Butler, A. P. Boedihardjo, and N. Ramakrishnan, "Storytelling in entity networks to support intelligence analysts," in *KDD '12*, 2012.

[8] L. Fang, A. D. Sarma, C. Yu, and P. Bohannon, "Rex: Explaining relationships between entity pairs," in *VLDB '11*, 2011.

[9] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *ICML'14*, 2014.

[10] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *EMNLP '19*, 2019.

[11] J. Dalton, L. Dietz, and J. Allan, "Entity query feature expansion using knowledge base links," ser. SIGIR '14, 2014.

[12] H. Raviv, O. Kurland, and D. Carmel, "Document retrieval using entity-based language models," in *SIGIR '16*, 2016.

[13] Y. Xu, G. J. Jones, and B. Wang, "Query dependent pseudo-relevance feedback based on wikipedia," in *SIGIR '09*, 2009.

[14] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *SIGMOD '08*, 2008.

[15] F. Erxleben, M. Günther, M. Krötzsch, J. Mendez, and D. Vrandeăić, "Introducing wikidata to the linked data web," in *ISWC '14*, 2014.

[16] D. Metzler, S. Dumais, and C. Meek, "Similarity measures for short segments of text," in *ECIR'07*, 2007.

[17] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, no. 11, Nov. 1975.

[18] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *ACL System Demonstrations*, 2014.

[19] E. Loper and S. Bird, "Nltk: The natural language toolkit," in *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*, 2002.

[20] G. Reich and P. Widmayer, "Beyond steiner"s problem: A vlsi oriented generalization," Tech. Rep., 1989.

[21] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "On finding lowest common ancestors in trees," in *STOC '73*, 1973.

[22] R.-H. Li, L. Qin, J. X. Yu, and R. Mao, "Efficient and progressive group steiner tree search," in *SIGMOD'16*, 2016.

[23] M. McCandless, E. Hatcher, and O. Gospodnetic, *Lucene in Action, Second Edition: Covers Apache Lucene 3.0*. Manning Publications Co., 2010.

[24] C. Xiong and J. Callan, "Query expansion with freebase," in *ICTIR '15*. Association for Computing Machinery, 2015.

[25] T. Tao and C. Zhai, "Regularized estimation of mixture models for robust pseudo-relevance feedback," in *SIGIR '06*, 2006.

[26] K. Collins-Thompson and J. Callan, "Estimation and use of uncertainty in pseudo-relevance feedback," in *SIGIR '07*, 2007.

[27] K. S. Lee, W. B. Croft, and J. Allan, "A cluster-based resampling method for pseudo-relevance feedback," in *SIGIR '08*, 2008.

[28] D. Metzler and W. B. Croft, "Latent concept expansion using markov random fields," in *SIGIR '07*, 2007.

[29] C. Zhai and J. Lafferty, "Model-based feedback in the language modeling approach to information retrieval," in *CIKM '01*, 2001.

[30] J. Bhogal, A. Macfarlane, and P. Smith, "A review of ontology based query expansion," *Information Processing & Management*, 2007.

[31] X. Liu, F. Chen, H. Fang, and M. Wang, "Exploiting entity relationship for query expansion in enterprise search," *Information Retrieval*, 2014.

[32] B. Aditya, G. Bhalotia, S. Chakrabarti, A. Hulgeri, C. Nakhe, P. Parag, and S. Sudarshan, "Banks: Browsing and keyword searching in relational databases," in *VLDB'02*, 2002.

[33] V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar, "Bidirectional expansion for keyword search on graph databases," in *VLDB'05*, 2005.

[34] Y. Yang, D. Agrawal, H. V. Jagadish, A. K. H. Tung, and S. Wu, "An efficient parallel keyword search engine on knowledge graphs," in *ICDE '19*, 2019, pp. 338–349.

[35] Y. Xu and Y. Papakonstantinou, "Efficient lca based keyword search in xml data," in *EDBT '08*, 2008.

[36] Y. Li, C. Yu, and H. V. Jagadish, "Schema-free xquery," in *VLDB '04*, 2004.

[37] C. Sun, C.-Y. Chan, and A. K. Goenka, "Multiway slca-based keyword search in xml data," in *WWW '07*, 2007.

[38] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., 1986.

[39] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Found. Trends Inf. Retr.*, vol. 3, no. 4, Apr. 2009.

[40] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, Mar. 2003.

[41] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide*, 1st ed. O'Reilly Media, Inc., 2015.

[42] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NeurIPS' 13*, 2013.

[43] J. H. Lau and T. Baldwin, "An empirical evaluation of doc2vec with practical insights into document embedding generation," in *Proceedings of the 1st Workshop on Representation Learning for NLP*, 2016.

[44] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *ACL '19*, 2019.

[45] Z. Li, J. Liu, X. Zhu, and H. Lu, "Multi-modal multi-correlation person-centric news retrieval," in *CIKM '10*, 2010.

[46] T. Yoon, S. Myaeng, H. Woo, S. Lee, and S. Kim, "On temporally sensitive word embeddings for news information retrieval," in *ECIR '18*, 2018.

[47] F. Abel, Q. Gao, G.-J. Houben, and K. Tao, "Analyzing user modeling on twitter for personalized news recommendations," in *User Modeling, Adaption and Personalization*, 2011.

[48] ——, "Twitter-based user modeling for news recommendations," in *IJCAI '13*, 2013.

[49] N. Bruno, "The threshold algorithm: From middleware systems to the relational engine." IEEE Computer Society, January 2007.

[50] Z. Liu, Y. Zhang, E. Y. Chang, and M. Sun, "Plda+: Parallel latent dirichlet allocation with data placement and pipeline processing," *ACM Transactions on Intelligent Systems and Technology, special issue on Large Scale Machine Learning*, 2011.

[51] M. Honnibal and I. Montani, "spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing," 2017, to appear.

[52] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Apr. 2017.

[53] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP' 14*, 2014.

[54] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading Wikipedia to answer open-domain questions," in *ACL '17*, 2017.

[55] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," in *EMNLP '18*, 2018.