QATAR UNIVERSITY

COLLEGE OF ENGINEERING

BACKGROUND LINKING OF NEWS ARTICLES

BY

MARWA MOHAMED ESSAM

A Dissertation Submitted to

the College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Doctorate of Philosophy in Computer Science.

December 2024

COMMITTEE PAGE

The members of the Committee approve the Dissertation of
Marwa Mohamed Essam defended on 01/06/2024.

Dr. Tamer Elsayed
Dissertation Supervisor

Prof. Iadh Ounis
Committee Member

Prof. Abdelaziz Bouras
Committee Member

Prof. Saeed Salem
Committee Member

Approved:

Khalid Kamal Naji, Dean, College of Engineering

ABSTRACT

Essam, Marwa, M., Doctorate : December: 2024, Doctorate of Philosophy in Computer Science.

Title: Background Linking of News Articles

Supervisor of Dissertation: Dr. Tamer Elsayed.

Nowadays, it is very rare to find a single news article that solely contains all the information about a certain subject or event. This dissertation focuses on addressing the news background linking problem, which aims to find news resources that can be linked to a query news article to help readers understand its background and context. We propose multiple approaches for addressing both the effectiveness and the efficiency aspects of this problem. We first conduct a qualitative analysis of some query articles and its background links to understand the notion of background relevance. Based on the insights drawn from this analysis, we propose two approaches for reranking a candidate set of background links that integrate the lexical and the semantic matching signals between the query article and its candidate links. We experimented with both Transformer-based models and Large Language Models, in a Zero-shot setting, to semantically link query articles to its background links. We conducted our experiments on the Washington Post news dataset, specifically released for news background linking, and we show that our approaches achieve a new state of the art performance for this problem. We moreover propose a query reduction approach that can speedup the retrieval of candidate links by up to 13.3x times through reducing the query article into representative search queries that are employed to retrieve the required background links in an ad-hoc setting. Our work further opens future research directions in addressing the news background linking problem through the analysis of the performance of our proposed approaches on different query articles, paving the way for enhancing the overall quality and depth of news reporting.

DEDICATION

To my mom and dad who always believed in me,

to my husband for his constant support,

to my daughters and sisters for being my emotional anchor throughout all the tough times

ACKNOWLEDGMENTS

This thesis acknowledgement is a tribute to all the people who made my academic journey worthwhile.

I am extremely grateful to my supervisor, Dr. Tamer Elsayed for his invaluable advice, continuous support, and patience during my PhD study. His immense knowledge and plentiful experience have guided me through all of my academic research time.

I express my sincere gratitude to my colleagues in the bigIR research group, specially Maram Hasanain, Reem Suwaileh and Fatima Haouari for sharing their expertise in the IR field, for offering continuous academic and emotional support, and for a cherished time we spent together in the group meetings.

I would also like to thank the Computer Science professors of Qatar University, whose valuable and useful courses paved the way to the the work in my dissertation.

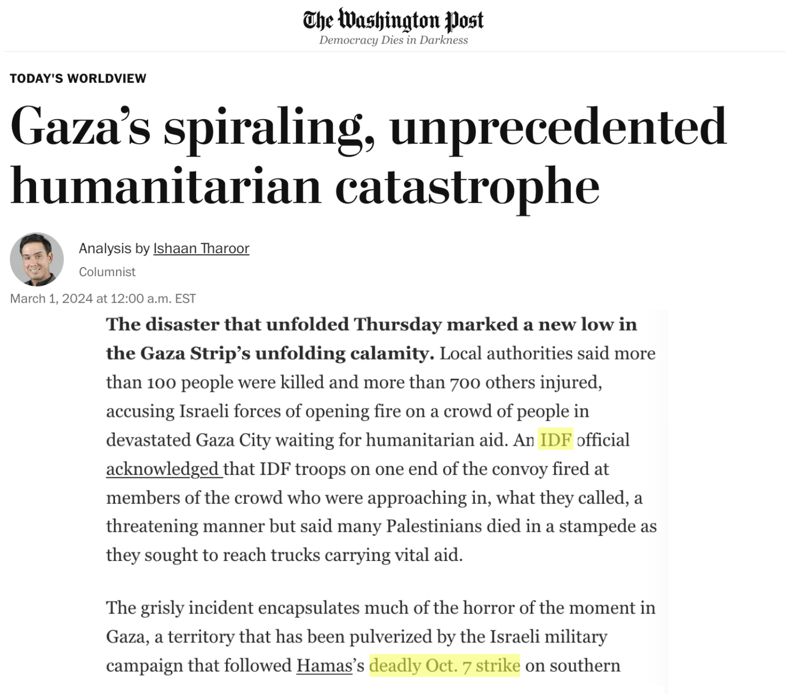
TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Many people nowadays acquire their knowledge about different subjects or events from online news articles. Yet, it is rare to find a single article that exclusively contains all the information about a certain subject or event. Most often, authors of different news articles assume that readers have some kind of background knowledge on the subject of the article they are reading. Therefore, in many cases, one randomly or accidentally reads an article (e.g., a newspaper article or a blog post) but does not fully understand its contents. Hence, he/she starts searching for other resources to help him/her to understand the background events/history that preceded this article and resulted in its publishing. The reader may also search for concepts or abbreviations that the article's author uses in his writing, and that the reader does not know about. However, with millions of information resources on the web, it becomes very difficult for a reader to select what to read next to understand the article at his hand. Moreover, a reader may end up with more documents that he/she does not also fully comprehend, and thus continues the searching process, while failing sometimes to acquire the needed knowledge.



The Washington Post
Democracy Dies in Darkness

TODAY'S WORLDVIEW

Gaza's spiraling, unprecedented humanitarian catastrophe

Analysis by Ishaan Tharoor
Columnist
March 1, 2024 at 12:00 a.m. EST

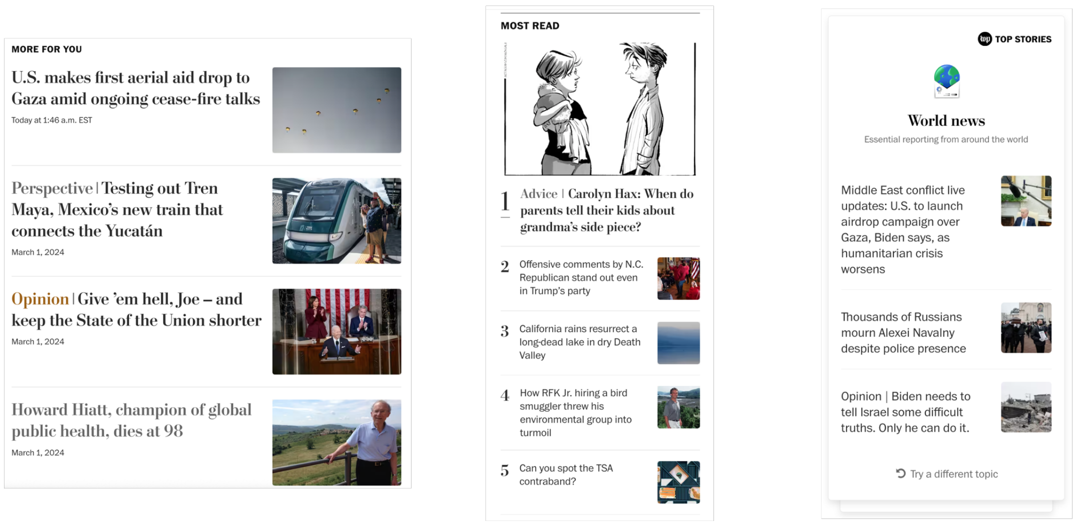
The disaster that unfolded Thursday marked a new low in the Gaza Strip's unfolding calamity. Local authorities said more than 100 people were killed and more than 700 others injured, accusing Israeli forces of opening fire on a crowd of people in devastated Gaza City waiting for humanitarian aid. An IDF official acknowledged that IDF troops on one end of the convoy fired at members of the crowd who were approaching in, what they called, a threatening manner but said many Palestinians died in a stampede as they sought to reach trucks carrying vital aid.

The grisly incident encapsulates much of the horror of the moment in Gaza, a territory that has been pulverized by the Israeli military campaign that followed Hamas's deadly Oct. 7 strike on southern

Figure 1.1: An Example of an article that a reader needs background information for.

Consider for example the news article in Figure 1.1. This article was recently published in the Washington Post newspaper, and it mainly talks about the current humanitarian catastrophe in Gaza and how its people are facing the risk of famine. The author of this article though, while referring to the 7th of October event that eventually led to this situation in Gaza (as shown highlighted in the figure), does not elaborate on it to help readers understand the full context of the situation. He also does not elaborate at all on the past conflict between Palestine and Israel that is the actual origin of all these current events. Note further that the author refers to the "IDF" without identifying this organization or telling any information about it. He has just put an assumption that readers of this specific article must have been following the recent events in Gaza and must have known this conflict and all its participants. This is not of course the case for all readers that might come across this story by chance with no background information about it. If interested in understanding this story, those specific readers may be eager to find resources that help explain the context of this story to them.

Recently, there have been attempts to explain news articles to readers. For example, the IndianExpress newspaper established an online section called "Explained" [1], where it attempts to explain articles that are published in other sections of the newspaper for readers who do not fully understand it. However, while this approach seems helpful to the readers, it may create duplicate content in the case where the concepts that are mentioned in the ambiguous article were already explained before in a previously published article. In such a case, instead of writing a new explanatory article, a link to the previous article will be sufficient.



MORE FOR YOU

U.S. makes first aerial aid drop to Gaza amid ongoing cease-fire talks
Today at 1:46 a.m. EST

Perspective | Testing out Tren Maya, Mexico's new train that connects the Yucatán
March 1, 2024

Opinion | Give 'em hell, Joe – and keep the State of the Union shorter
March 1, 2024

Howard Hiatt, champion of global public health, dies at 98
March 1, 2024

MOST READ

1 Advice | Carolyn Hax: When do parents tell their kids about grandma's side piece?

2 Offensive comments by N.C. Republican stand out even in Trump's party

3 California rains resurrect a long-dead lake in dry Death Valley

4 How RFK Jr. hiring a bird smuggler threw his environmental group into turmoil

5 Can you spot the TSA contraband?

TOP STORIES

World news
Essential reporting from around the world

Middle East conflict live updates: U.S. to launch airdrop campaign over Gaza, Biden says, as humanitarian crisis worsens

Thousands of Russians mourn Alexei Navalny despite police presence

Opinion | Biden needs to tell Israel some difficult truths. Only he can do it.

Try a different topic

Figure 1.2: Different types of links added to the news article shown in Figure 1.1.

Sometimes, authors or journalists manually add links to resources that contain the missing background information. See for example the author's underlined words in Figure 1.1, through which the author refers to previously published articles about the same topic of the article. This is not the case for all news articles and with all authors, though. In this context, it became vital to automate the background linking for readers, and to find techniques to link generally documents, and specifically news articles using the background knowledge they provide to one another. This way, when a reader reads one news article, he/she can directly follow background links to other resources/documents that help him/her understand the content of the news article at hand and gain more knowledge on its subject.

While news providers nowadays automatically add links to the published news articles, these links usually point to other articles written by the author, other articles that are mostly viewed on the website, or articles that are recommended to a reader based on his reading profile. In many cases, these links are not sufficient for the reader for acquiring the background knowledge. Figure 1.2 shows the links that were added to the article in Figure 1.1 while we were browsing it. As can be seen, almost all of these linked articles do not help at all in understanding more about the situation reported in the article, nor its context. It is important to note here further that adding background links manually, while writing his/her article, is a burden on the author.

To motivate the research on this specific problem, a background linking task, that aims to find background links for news articles, was recently introduced as a new challenge in the news track in the Text REtrieval Conference (TREC) in 2018, and as follow-up tasks in TREC 2019, 2020, 2021 [2]. In a collaboration with the Washington Post newspaper, TREC specifically released a dataset of news articles and blog posts for the news track, and asked participants to find background resources from within the collection to a set of input news articles.

Over the years of running this task within TREC, participants proposed many solutions to the news background linking problem, using both unsupervised and supervised solutions [3]–[5][1]. Among those solutions, the most stable and effective one adopts a lexical similarity based approach, where each input article is used as a search query to retrieve its background links from the given dataset, in an ad-hoc setting. This method was proposed by many teams, using different settings and with different retrieval models. Yet, regardless of its setup, it still, on average considering all released input articles, outperformed other solutions proposed, making it the state of the art (SOTA) for addressing the news background linking problem. The optimal reranking of the results obtained through this method shows, however, big room for improvement, confirming the need for an effective technique for reranking the background links.

The full-article retrieval approach is additionally not efficient as it involves issuing a “relatively” very long search query to the collection of possible background links. This puts a burden on the retrieval system, specially in situations where the newspaper is processing a large number of input articles, or where the lookup for background resources is performed online by the article’s author or its reader. Thus, other methods are needed to be investigated to reduce the lookup time needed to obtain the set of background links, to either represent it to the reader as is or to consider it for further reranking purposes.

In this dissertation, we address the news background linking problem, proposing approaches that improve both the effectiveness and the efficiency of the linking process. We started our work through the first qualitative analysis of news articles for background linking [6]. This analysis helped us to draw very informative insights on the notion of background relevance that paves the way when working on this problem. For addressing the effectiveness aspect of the problem, we additionally propose two new state of the art candidate background links reranking techniques that integrate both the lexical and the semantic matching signals between the query article and the background links for an effective linking process. We experimented with Transformer-based models [7] and Large Language models [8] in our proposed approaches to capture the semantics of news articles. For addressing the efficiency aspect of the background linking process, we further propose a technique for reducing the input article into a much smaller search query to efficiently and still effectively retrieve the candidate set of background links [9].

Following, we formally define the news background problem in Section 1.1, distinguish it from other similar problems in literature, and further give examples of some

[1]The overview paper for the news track in 2021 is not available on TREC website. Only participants were able to download the notebook paper

query articles and its background links to highlight the problem's challenging nature. We then detail the research challenges in Section 1.2, and list the addressed research questions in Section 1.3. Later, we give an overview of our proposed solutions in addressing this problem in Section 1.4, and finally we list our contributions to the research community in Section 1.6.

1.1. Research Problem: News Background Linking

As mentioned before, authors often assume that newsreaders have some kind of background knowledge of the subject/story of the news article. Readers who do not have this knowledge often search for it on the web to comprehend the article at hand. News background linking aims to automate this knowledge acquisition process for the readers, or to automate the background linking process for authors if they opted to add background links upon publishing their articles. Following, we formally define the problem, give examples of it, and distinguish it from other similar problems.

1.1.1. Problem Definition

We define the background linking problem as follows:

Definition 1. ***News Background Linking Problem***

Assume we have a query news article q that a reader does not fully understand, and a collection of text resources C, suggest a ranked list of resources $B \subset C$ that a reader of q can read to provide him/her with the context and background knowledge required to comprehend the content of the query article q (see Figure 1.3).

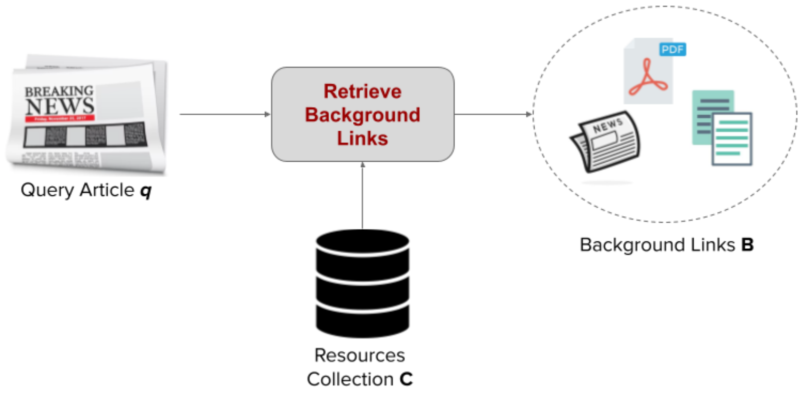


Figure 1.3: Retrieving a set of background links for a query news article

In the context of the above definition, we further define the following terms, as we will use it throughout the dissertation:

Definition 2. ***Query Article***

A query article q is an input news article that a reader does not fully comprehend, and requires more background knowledge to contextualize its content.

Definition 3. ***Background Link***

A background link of a query news article q is a document that contains some knowledge required to comprehend and conceptualize some of the content of q.

The background link can be a link to any useful background resource. However, since news background linking was mainly addressed in the literature within TREC using the Washington Post collection of news articles specifically released for the news background linking problem, in the rest of this dissertation we will assume that the background link to a query article is a link to another news article that contains the required background information.

1.1.2. Difference from other problems

Finding *related* articles in general has been studied in the literature for many years. News recommendation is the closest problem in definition to background linking, assuming that we search a collection of only news articles [10]. The difference though between the two problems is that in news recommendation, the proposed system uses the reader's profile including his/her browsing history, online interactions, or sometimes his/her location during the recommendation process. This usually helps the recommendation system to recommend news articles that are of interest to the reader [11]–[13]. However, in news background linking, the reader is assumed to be anonymous, with no possible way of finding any information related to him/her. Accordingly, the attention is mainly on the content of the query article.

There has been some further work recently on event detection within new articles, and also on finding the story lines of those events [14]–[18]. In most of that work, an event is assumed to definitely exist within the news article, and the goal of the system is to extract the components of this event (what happened, who, when, where, and how). This is not assumed in background linking, as some articles may discuss general subjects and may not at all refer to specific events. Additionally, in some cases, an article may be helpful to the reader even though it does not mention the specific event of the query article.

Another close problem is the clustering of related news articles within a news articles collection [19]–[21]. In this problem, news articles are clustered into a limited number of clusters such that each cluster has a high internal coherence (i.e., having articles addressing the same topic, such as sports, politics, etc.), and low external coherence with the other clusters. While clustering new articles might reduce the number of candidate articles for background linking, the selection of which articles to recommend for the reader from within the cluster is still challenging. Additionally, there might be articles from another cluster that make a very relevant background to an article within the cluster.

1.1.3. Examples

Following, we give examples that better clarify the news background linking problem, showing relevant and non-relevant background links and highlighting the difference between this problem and the other problems in literature as shown above.

Figure 1.4a shows an example of news background linking, where the query article discusses the progress in the trial of a truck driver who was accused of the death of

10 illegal Mexican immigrants in his truck. The relevant background articles give the reader more context and background information on this specific incident. One article for example discusses how the company that owns the truck, for which the driver works, is famous for falsifying records, and how the company denies any accusations to it regarding this incident. The second article does not mention this specific incident, however it discusses the patterns of Mexican immigration to the United States, and how the immigration pool should be handled.

The non-relevant article in Figure 1.4a also discusses a case of illegal immigrants from Mexico, but it does not at all contain information that help readers of the query article understand its content or even contextualize it. Instead, it just details the possible consequences that two soldiers will face after being convicted for illegally driving two Mexican men through an immigration checkpoint. It is important to note here that a typical news recommendation system may retrieve this specific article and link it to the query article as it might be of interest to a reader who reads about illegal immigration cases in the U.S for example or the world in general.

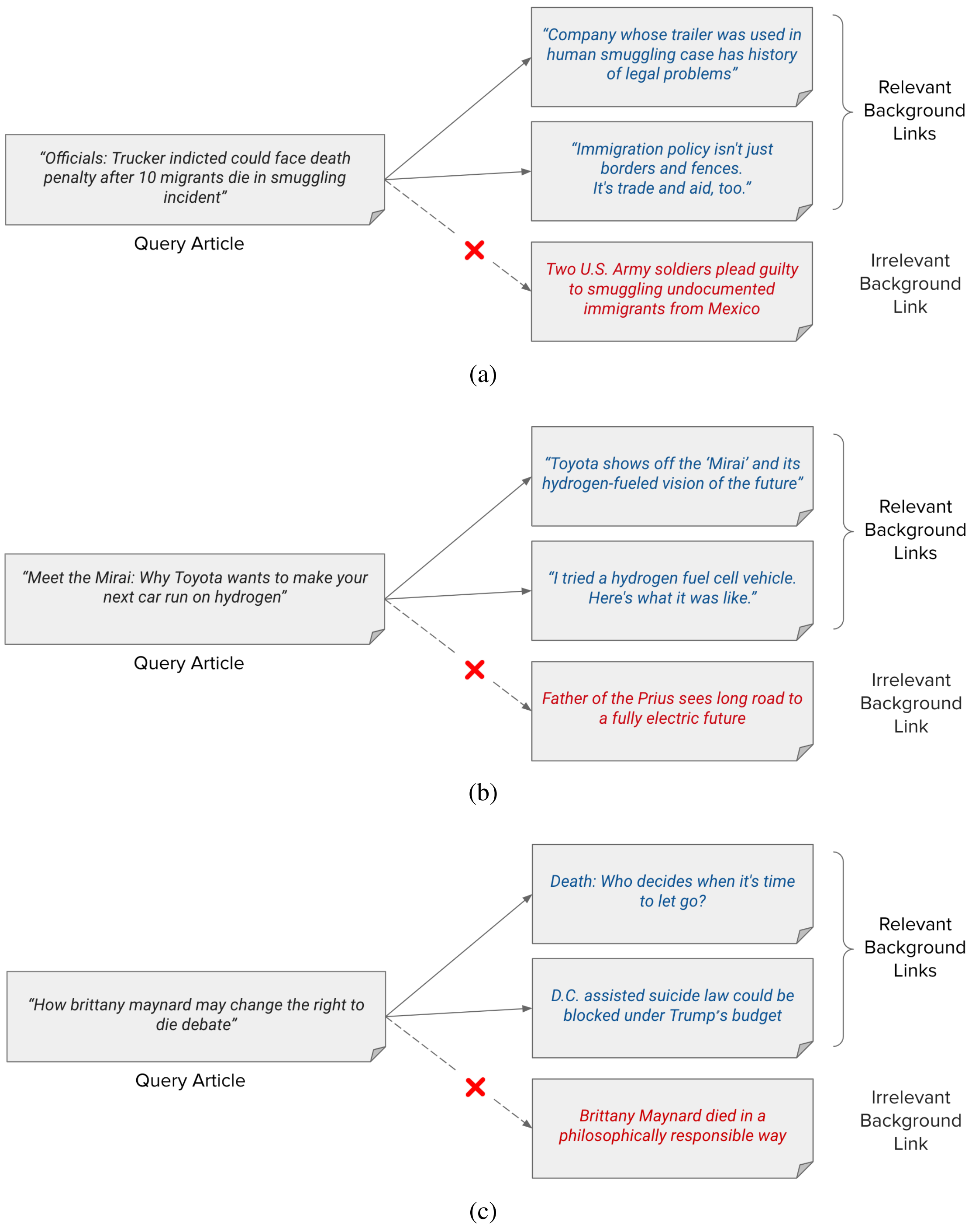

"Officials: Trucker indicted could face death penalty after 10 migrants die in smuggling incident"
Query Article
"Company whose trailer was used in human smuggling case has history of legal problems"
"Immigration policy isn't just borders and fences. It's trade and aid, too."
Relevant Background Links
Two U.S. Army soldiers plead guilty to smuggling undocumented immigrants from Mexico
Irrelevant Background Link
(a)
"Meet the Mirai: Why Toyota wants to make your next car run on hydrogen"
Query Article
"Toyota shows off the 'Mirai' and its hydrogen-fueled vision of the future"
"I tried a hydrogen fuel cell vehicle. Here's what it was like."
Relevant Background Links
Father of the Prius sees long road to a fully electric future
Irrelevant Background Link
(b)
"How brittany maynard may change the right to die debate"
Query Article
Death: Who decides when it's time to let go?
D.C. assisted suicide law could be blocked under Trump's budget
Relevant Background Links
Brittany Maynard died in a philosophically responsible way
Irrelevant Background Link
(c)

Figure 1.4: Examples of news background linking drawn from the TREC dataset for the news track

Another background linking example is shown in Figure 1.4b, where the query article is talking about energy-saving cars released by "Toyota" and how the company is placing a bet on hydrogen cars, focusing on the performance of its recently released car "Mirai". The first relevant background article is showing the history of releasing "Mirai" into the market, while the second discusses the experience of its author while driving another hydrogen-fueled car and the challenges that these cars face. The non-relevant background link, although briefly mentioning Toyota's earlier plans to produce both hydrogen-fueled cars and full-battery cars, does not give sufficient details on any. It focuses more on the company's chairman opinion that there is still a long road for full-battery vehicles to become a widespread replacement for gas. Clearly, this article is a non-relevant background link as it does not help the reader at all gain more information on the car "Mirai", nor on the history or the capabilities of hydrogen-fueled cars like "Miari". Again, we can notice that a news recommendation system may retrieve this specific non-relevant article for readers of the query article, as it contains information that might interest one who reads stories about the future of electric cars.

A third example is shown in Figure 1.4c. The query article discusses the effects of the death of a young lady "Maynard", who took pills to die after being diagnosed with cancer, on the "right to die" debate in the United States. The background articles give the reader more context on this debate, without even mentioning the specific case of "Maynard". The first article discusses the approval of the "right to die" law after case courts, focusing on a case of a lawsuit in Texas where a man is asking a hospital to release his brain-dead wife from a ventilator. The article further shows context around approving this law in other states like New Mexico. The second relevant article discusses an economical view on approving the right-to-die law in former president Trump's budget. The non-relevant background article, while actually mentioning the case of "Maynard", discusses "Maynard" death from a sentimental aspect. It merely talks about how people like "Maynard" who devoted their few remaining days in life to a cause are not self-centered. Hence, this article gives no information on the 'right to die' debate or even more information regarding 'Maynard' death, making it of no use for a reader who wants to understand more the contents of the query article. It is important to note here that a system that aims to find story lines may link the query article to this former non-relevant article as both mention the specific case of "Maynard's" death. It will further exclude the other relevant background links from this story line. This clearly shows why the news background linking problem can not be simply and directly addressed using techniques adopted for events detection in news articles.

1.2. Research Challenges

Following, we list the different research challenges when working on the news background linking problem.

Newsreader's Anonymity As mentioned before, a newsreader is assumed to be completely anonymous in the news linking problem, leaving solely the content of the query article as input to the news background linking process. This content is assumed to contain places of ambiguity or missing information that is supposed to be shown in the useful background article. Accordingly, the first challenge in this problem is to find the specific relation between the text content of the background article and the text content of the query article that controls that background article's usefulness (in other words, relevance) to the query.

Non-binary Relevance Scale The second challenge is that the relation between a query article and a background link is not binary (i.e. relevant or not relevant). While some background articles provide critical background information for the query article, some others just help readers understand the broader story of its context. Accordingly, an effective news background linking system must not only retrieve relevant links, but also ensure that the links are ranked according to its usefulness when presented to the reader.

Background Relevance Criteria Upon its introduction to the research community in 2018 [3], many researchers attempted to address the news background linking problem, proposing a number of solutions that adopted unsupervised, supervised techniques, or sometimes both [22]. The most effective method reported, to date, simply used, without deep analysis of the content, the query article as a search query in an ad-hoc setting to retrieve the background links [23]. This method depends on only the lexical similarity between the query article and its background links. Accordingly, it ranked low articles that are very useful, but for example presented the background information using different vocabulary. The optimal ranking of the links retrieved through the full-article search method, however, shows very high potential for performance improvement. This means that this method may be used as an initial step to retrieve candidate background links for a later reranking process that can effectively rerank the retrieved articles based on its actual background relevance to the query article. One must notice though that the word "relevance" in this problem is a tricky word as, most probably, all articles, retrieved through lexical similarity with the query article, have somehow context relation (in other words, relevance) with it. Accordingly, to develop an effective background links reranking system, the notion of "background relevance" must be clearly understood, in order to be able to specifically identify the criteria that distinguishes more useful background links from less useful ones.

Limited Context Length of Pre-trained Language Models To get deeper insights of the context within the news articles, supervised learning models may be adopted. Since the news background linking problem is relatively new though with limited data for training and testing new supervised models from scratch, semantic-based techniques that rely on transfer learning may be considered [22], [24], [25]. News background linking is faced by another challenge here though, which is the context length restrictions of the input to many pre-trained models. To avoid truncating news articles during the reranking process, one needs to find a model with an input context length that can fit a pair of relatively long news articles (the query article and one candidate) in the case for point-wise reranking of the candidate background links, not to mention the need to fit more articles in the case for pair-wise or list-wise reranking of candidate links. Accordingly, there is a challenge her on how to adopt pre-trained models, given their context limits, for news background linking.

Crafting News Linking Prompts for Large Language Models Very recently, research has been conducted on harnessing the reasoning capabilities of Large Language Models (LLMs) in many information retrieval tasks, and the results are very promising [26]. To overcome the context limitation of earlier pre-trained models and to reveal the context discussed within news articles and link these articles based on this context, LLMs may further be adopted. Up to our knowledge, the news background linking problem was never addressed using LLMs before. The LLMs performance depend heavily on the prompts given to the models, leaving the challenge in how to adequately craft prompts to the LLMs that allow it to effectively analyze and link the news articles.

Inefficient Retrieval of Candidate Background Links Finally, the full-article retrieval method clearly involves processing relatively long search queries (the query articles itself), making it inefficient for retrieving the background links either to present it as is to the reader, or to use it in an initial stage of a candidates reranking system. This is especially valid in scenarios where the background linking process is performed online, searching a big collection like the Web. Many terms that appear within the articles are just noise and does not actually represent its mere content, thus not needed to be included in the search queries. This shows a need for an effective background linking system that can produce background links as with this full-article retrieval approach but in a more efficient way.

1.3. Research Questions

To address the above research challenges in news background linking, we formulated the research questions given below. The dissertation chapter that shows how we address each question is further given.

- **RQ1** : What criteria in both the query and its candidate background articles affects the background retrieval relevance? **(Chapter 3)**
- **RQ2** : Can we adopt Encoder models for a semantic based reranking of the links retrieved through the full-article search approach? **(Chapter 4)**
- **RQ3** : Can we harness the potential of the recently developed Large Language Models (LLMs) to effectively rerank the links retrieved through the full-article search approach? **(Chapter 5)**
- **RQ4** : How can we increase the efficiency of the ad-hoc based retrieval of news background links? **(Chapter 6)**

1.4. Overview of Proposed Solutions

In this dissertation, we improve the news background linking process by proposing techniques that automate the lookup for background links of a given news article. The work introduced here builds upon the ideas that we initially proposed for addressing the news background linking problem [27]. The proposed techniques address the news background linking problem from two aspects: effectiveness and efficiency. To propose these techniques, we first conducted a qualitative analysis of a sample of query articles and its background links in an attempt to understand the notion of background relevance [6]. Through our analysis, we drew multiple useful observations that paved the way for our proposed news background linking solutions. These observations may further guide the future research in this problem.

Addressing the effectiveness aspect of news background linking, we investigated how to effectively rerank the candidate set of background links obtained using the full-article ad-hoc based retrieval approach. We experimented with two approaches for this purpose: The first [7] uses a transformer-based model to encode both the query article and its candidate links in the semantic space, then rerank the candidate links based on an aggregation of both its lexical similarity and semantic similarity with the query article. In the second proposed approach [8], we adopt Large Language Models (LLMs) to first analyze the query article, reveal its discussed subtopics, then extract the relevant information related to those subtopics from each candidate link . Finally, we ask the LLM to rerank the candidate links in a zero-shot list-wise reranking setting. We show in this work that both of our proposed reranking approaches exhibited a statistically significant improvement over the full-article search method, marking new SOTA for the news background linking problem.

To increase the efficiency of the background linking process, we propose to enhance the lexical retrieval phase of the news linking process by reducing the size of the search query issued to the retrieval system. Mainly, we propose to analyze the query article, reveal its most influential keywords, then use these keywords instead of the full article, in a simple ad-hoc setting, to retrieve the set of background links. We experimented with multiple keyword extraction techniques for this purpose [9]. Through our proposed search query reduction approach that adopts an unsupervised statistical keyword extraction technique, we show that we can still effectively retrieve a set of relevant background links, compared to the full-article search approach, but with considerably limited number of search terms.

1.5. Research Findings

Below, we list our major research findings that pave the way for enhancing news reporting through news background linking.

- Through our qualitative analysis for the notion of background relevance, we found that query articles that are driven by the occurrence of some specific event in time (e.g., a crime or a natural disaster) are, on average, *harder* to handle than other articles that explore more general issues.
- We further found through our qualitative relevance analysis that highly relevant background articles need to detail information on one/more subtopics mentioned in the query article, with additionally introducing its own new subtopics or aspects related to the query article's main topic.
- We found that matching background links to query articles based on only lexical similarity is a relevance factor for news background linking, but is not sufficient for an optimal background linking process.
- We found that the best dense representation of news articles for the news background linking problem can be obtained though a sentence-level hierarchical aggregation approach that adopts a sentence-based encoder.
- We found that an effective reranking of the candidate background links for a query article cannot be achieved solely by semantic matching; We show that an aggregation of the semantic and lexical matching scores can be used to better rerank the candidate set.

- Through our experiments with some Black-box LLMs, we found that these LLMs can not be directly asked to effectively rank a candidate set of background links for a query article. Yet, an analysis of the query article and its background links through these LLMs allowed us to harness the capabilities of the LLMs for an effective background linking process.
- The most effective and the most stable in performance LLM we experimented with in news background linking is the black-box GPT4-Turbo model.
- We found that the query article needs to be considered as a whole for news background linking, and its lead paragraphs are not sufficient to substitute it during the background links retrieval process.
- We found that a simple statistical-based keyword extraction method can be adopted to reduce the full-article search query into much fewer search terms, leading to an efficient retrieval of the candidate background links, while maintaining the effectiveness of the full-article search approach.

Throughout the rest of this dissertation, we elaborate on each of the above findings while addressing our research questions.

1.6. Contributions

The contribution of this dissertation is multi-fold as shown below:

- We provide an extensive review of the literature of content only based news articles contextualization and linking.
- We are the *First* (to our knowledge) to conduct a qualitative analysis of relevance for the news background linking problem. Based on our analysis, we draw informative insights on the notion of background relevance that can guide the future research on the problem.
- We propose a zero-shot reranking technique for background links that adopts transformer-based models to semantically match the query articles to its candidate links. The proposed technique outperforms the full-article retrieval approach, on average, over all the query articles released for news background linking, establishing a new state of the art for the news background linking problem.
- We are the first, up to our knowledge, to address the news background linking problem using Large Language Models (LLMs).
- We propose another reranking technique for background links that adopts a Large Language Model, post curating its prompts on a training set of the query articles, to analyze each query article and its background links and rerank those links effectively according to this analysis. This approach further significantly outperforms the full-article retrieval approach on the test set of query articles.
- We show that we can quite efficiently reduce the query response time needed for the retrieval of the set of candidate background links using simple unsupervised statistical keyword extraction techniques, with no statistically significant difference in effectiveness than with the full-article retrieval approach.

- We make our data and code for running all our experiments publicly available for the future research on the news background linking problem.

The rest of this dissertation is organized as follows: In Chapter 2, we provide a review of the literature on contextualizing news articles. Since we are interested in our work on reranking the candidate set of background links retrieved through the full-article search approach, we review further the recent work done for long documents reranking. In Chapter 3, we present our qualitative analysis for the notion of background relevance. In Chapter 4, we propose our first approach for an effective news background linking that adopts transformer-based models for the semantic representation of news articles. Chapter 5 details our work in harnessing the capabilities of LLMs for news background linking. In Chapter 6, we propose our work to address the efficiency aspect of news background linking. Finally, in Chapter 7, we conclude this dissertation and show possible future work.

CHAPTER 2: LITERATURE REVIEW

In the last decade, online news reporting services have played a vital role in changing the way people share and receive news. Nowadays, many people read news online, whether on the website of published newspapers or on digital-only news platforms. As the news is increasingly being transmitted in the digital format and with news sites post new information on an hourly basis, online news articles have been an excellent, rich and reliable sources of information for many researchers in different fields. Consequently, the research in news articles analysis have been ongoing heavily for many purposes such as sentiment and bias analysis [28], [29], fake news detection [30], financial market prediction [31], and event detection [32], among others.

While the research on news articles content analysis is vast, we focus in this chapter on the research work that is mainly related to news articles conceptualization and to news articles linking. Precisely, in Section 2.1, we survey the work done for understanding the contents of news articles, with a focus on the work proposed to specifically solve the news background linking problem within TREC. We additionally introduce the work proposed earlier work for content-based news recommendation in Section 2.2.

As mentioned in Chapter 1, in our study, we are interested in reranking the set of background links retrieved by using the query article as a search query. Accordingly, in Section 2.3, we further review the work done, in general, for direct long documents reranking.

2.1. News Articles Contextualization

Often, an author of any news article assumes that readers have certain background knowledge on the article's topic or story. This is of course not the case for many readers. Therefore, it is essential to enrich articles with links to other sources of knowledge to provide readers with the desired missing information. In this section, we review the work precisely done for news articles contextualization. We first start by reviewing the work done in TREC (considering other news articles than the query as sources of knowledge). We then review the other work that was done earlier and considered other sources of knowledge (such as Wikipedia pages).

2.1.1. News articles as sources of knowledge

News background linking is a problem that was mainly introduced in TREC starting from 2018 till 2021 [3]–[5][1]. Within TREC, a collection of news articles from the Washington Post newspaper was released, and participants were asked to find background links from within this news articles collection to a set of input query articles. Retrieved articles from participants were pooled, then judged based on how much they help readers of query articles in understanding its content and context. In this section, we first introduce unsupervised methods for news background linking, that mostly adopted an ad-hoc based retrieval approach to find the background links. Then, we introduce the work that adopted supervised models for news background linking.

Unsupervised News Background Linking The news background linking problem can simply be addressed following an ad-hoc search approach, in which an input query

[1]The overview paper for the news track 2021 was not published to the research community. Only participants papers can be found on the website: https://trec.nist.gov.

article is used as a search query and an inverted index of a collection of resources is looked up for candidate links that match the search query. Since the best match from the collection is the query news article itself, an exact match check is performed and the query article is excluded from the results. Following this approach, Yang and Lin [33] introduced several methods that were all implemented in Anserini [34]. One method selected the 1000 article's terms having the highest *TF-IDF* values from the article, and used them as a search query. Another method used the first 1000 terms that appeared in the article as the search query. Lopez-Ubeda, Diaz-Galiano, Valdivia, *et al.* [35] constructed different queries using the article's title, the full body text, and both the title and the body, to generate initial results. The candidate articles were then selected based on a k-means clustering to diversify the results for the newsreader.

A number of teams focused on extracting the most representative keywords from the query article to be used as their search query. Bimantara, Blau, Engelhardt, *et al.* [23] proposed to use key phrases extracted using $TextRank$ [36] (a basic graph-based text analysis algorithm) as a search query. They also investigated the use of a list of named entities extracted also from the query article as the search query. Lua and Fang [37] suggested splitting an article into paragraphs and extracting query terms from each paragraph to form multiple search queries using a probabilistic model. The authors later suggested, to assign different weights to a list of spotted DBpedia entities extracted from the input articles [38] , and use these entities as their search query [39]. In 2019, we participated in the 2019 track, using graph-based approaches to generate the search queries [40]. Precisely, we experimented with the graph-based keyword extraction mechanisms k-Core [41] and k-Truss [42] to determine the most influential terms from the query article, and used up to 20% of the extracted terms from the query article as a search query. Missaoui, MacFarlane, Makri, *et al.* [43] initiated a search first using the named entities extracted from the query article's title. They then used the top 10 retrieved articles in expansion to update the search query. Following the same approach but with different initial queries, Ding, Lian, Zhou, *et al.* [44] used the title and body of the input article with weights equal to 0.3 and 0.7, respectively, to retrieve their first list of candidate links. They then adopted Rocchio and RM3 for expansion.

Wagenpfeil, Mc Kevitt, and Hemmje [45] converted each document in the Washington Post dataset to a node graph called Multimedia Feature Graph [46], and applied similarity measures between the query article graph and other graphs to obtain the ranked list of background links. Engelmann and Schaer [47] worked on reranking the top 200 articles retrieved using the full-article retrieval approach. They specifically extracted the named entities from both the query and candidate documents, and developed a formula, based on their analysis of the queries released in 2020, that computes the similarities between those entities given their relation in the text.

Another technique was proposed [48] that attempts to selectively expand the query articles during the lexical matching process based on whether this article is reporting time-sensitive information. To retrieve the candidate background articles though, the authors used, as a search query, the description of what readers want from a background article (i.e., the information need). This description was only available for the TREC 2021 set of query articles that the authors experimented on, and was given by the track organizers in TREC that year to show participants examples of what a reader would want when seeking background articles. However, in a real life scenario, this description is not intuitively known ahead, as news providers give readers only the news articles without a description of what is missing in it (required background knowledge). Therefore, it is the goal of a background linking system to discover what is missing and use it to retrieve the required background links.

Most of the unsupervised techniques shown above, in one way or another, attempted to capture a sort of lexical similarity or distance between the query article and its background links. While important, some useful background links may be contextually helpful, yet written with different vocabulary, different writing style, or using different entities, than the query article. Hence, we hypothesize that ensuring that the semantics of both articles are captured and evaluated is further essential to ascertain that all useful background articles are retrieved during the background linking process.

Supervised Models for News Background Linking Regarding the use of supervised models for news background linking, teams followed two lines of work. The first line used the learned models as is, without further training or fine-tuning, in order to represent the query and background articles in the semantic space, and compared the pair's semantic representation hoping to reveal its actual contextual relatedness. In the second line of work, the participating teams attempted to learn models to either represent the articles in the semantic space or to estimate the usefulness of a background article to a query article.

Following the first line of work, Khloponin and Kosseim [49] semantically represented all articles in the news collection as vectors using Doc2Vec distributed representation [50], then they ranked all articles in the collection using its cosine similarity with each query vector. Day, Worley, and Allison [51] followed the same approach, but computed embeddings for the documents using Sentence-BERT [22]. In their work however, an initial set of background links were first retrieved using the highest in *TF-IDF* terms as a search query, then for each article in this set, an embedding is generated by pooling the embedding vectors of the leading three paragraphs generated by Sentence-BERT. Khloponin and Kosseim [52] explored other document representations models, with different proximity measures. Among the models they compared, they found that GPT2 and XLNet embeddings generally lead to higher performance in the semantic representation of articles for this task, yet not to the extent that outperforms the lexical matching for news background linking.

Deshmukh and Sethi [53] and Sethi and Deshmukh [54] suggested first to obtain a candidate list of background links by using a search query constructed from terms extracted from within the query using a TF-IDF updated formula. RAKE keyword extraction algorithm [55] was then used on each retrieved candidate to produce a set of keywords which were further concatenated and fed to a Sentence-Bert model to obtain an embedding for the candidate. Candidates were finally sorted given their keywords semantic similarity to the query article. Cabrera-Diego, Boros, and Doucet [56] proposed to initially obtain an embedding for each article in the Washington Post dataset by averaging all the tokens embeddings generated by a pretrained model from the Sentence-BERT family, after feeding the query article to the model sentence by sentence. The generated embeddings were then indexed in a dense index using the Open Distro for ElasticSearch[2]. Finally, to get the background links for a specific query article, the authors issue three queries to the indexing platform, the first two are lexical based ones using keywords extracted from the article's title and the article's body and

[2]https://opendistro.github.io

the third is a query that asks the index to retrieve the documents using K-NN and cosine similarity of the dense vectors.

Following the second line of work, Rahul Gautam [57] trained first a word2vec model on all articles in the news collection. They then represented each article as the sum of the vectors of its most frequent non-stop 25 words. Cosine similarity between the embeddings of the query article and the candidate background articles was then used for obtaining the background links. Foley, Montoly, and Pena [58] trained linear learning-to-rank retrieval models on the 2018 set of queries, and adopted the Coordinate Ascent model [59] to select the best set of features for the model. Those features included textual similarity features, temporal features(ex: difference in publication dates), quality features (ex: candidate document length) and Clickbait features (ex: clickbait-probability of candidate title). Koster and Foley [60] trained a Random Forest model on the 2018, 2019 and 2020 sets, with the same set of features. This model though failed to produce more effective results. Qu and Wang [61] used logistic regression to train a multi-class classifier to rerank candidate background articles. Candidates were classified from 0 to 4 according to how much knowledge they provide to the query article. The top 80 frequent words in the query article (using *TF-IDF* scores) were used as a search query to retrieve an initial set of 1,000 candidate background articles.

Adopting BERT [62] for background linking [62], Ak, Köksal, Fayoumi, *et al.* [63] suggested further two supervised methods: The first used BERT-based extractive summarization methods to generate a summary of 180 words from the query news article, then used this summary as a search query to retrieve the background links. The second method fine-tuned the next sentence prediction task in BERT for background linking. Since BERT input is limited to 512 tokens, both the query article and the candidate background article were cropped to 256 tokens during the fine-tuning process.

In an attempt to analyze the different topics discussed in the news articles and how much the common topics between articles affect relevance in this problem, Ajnadkar, Jaiswal, Gourav Sharma, *et al.* [64] used a trained LDA model to analyze the news articles in the Washington Post dataset and obtain its topic distributions, then find the most similar articles to a given query article based on the maximum cosine similarity between the topics representation vectors.

The work reviewed above attempted to capture, somehow, the semantics of the query and the candidate background links for a better linking process. Yet, the techniques proposed treated the query article as one entity. Most often though, a news article discusses several subtopics related to its main topic within its paragraphs, and a useful background article may extend the reader's knowledge on just one or more of these subtopics, not necessarily all of them. Accordingly, we hypothesize that an effective background linking technique needs to evaluate the information provided in the candidate background article's against the different components/parts of the query article, not against the query article as a whole. In fact, in the 2021 news background linking track, TREC released a set of subtopics manually extracted from each query article, and asked participants to find background articles for each subtopic. We participated in this track with a background linking system that splits the candidate background links into paragraphs, uses a passage reranking model to rerank the paragraphs given the subtopic as a query, then assign the background article the score of its maximum scoring paragraph [65]. In a real life scenario though, the subtopics of a query article are not intuitively known ahead. Thus, we need to figure out how to effectively split the article into subtopics.

Finally, considering all the techniques/ methods used to retrieve the background links from news articles, we note that the most stable and effective method presented by the participants within TREC, considering most years of the track, was a brute-force method that used the full input query article as a search query in an ad-hoc setting to retrieve the background links. This approach was introduced by different teams using different platforms and with different preprocessing setups. For example, some participant teams opted to filter out articles that were published after the query article and other teams did not. Also, some teams filtered non-relevant articles ahead of the indexing process, and others included it in the index and filtered it out or penalized it after the retrieval process. Yet, regardless of the setup, this simple retrieval method always showed a stable high performance compared to all other proposed approaches for news background linking. Aside from the work of some participants that showed slight improvement for some of the queries released specifically in 2021 over their implementation of the baseline method [47], this method still, on average considering all the query articles released in the news background linking track, outperformed most of the other approaches proposed, making it the SOTA for the news background linking problem.

It is important to highlight here, however, that the full-article ad hoc-based retrieval approach considers only the lexical similarity between the query article and the background links as a relevance signal. Further, it considers the query article as a whole during the matching process. As mentioned before, this approach in retrieving the background links might miss background articles that are written with a different vocabulary or that expand the newsreader on only one or few aspects mentioned in the query article. Therefore, assessing the background links using other relevance signals, other than the lexical similarity with the whole query article, is essential to ensure an effective background linking process. The optimal reranking of the candidate links retrieved by the full-article retrieval approach though shows big room for improvement. This leaves a challenge in building an effective news background links reranking system that can effectively rerank the background links for a query article while considering multiple relevance signals, which is our focus in this work.

2.1.2. Other sources of knowledge

Aside from the work that was done for the background linking task in TREC, there has been some research into contextualizing the news articles using external sources (other than other articles from the same collection). Ravenscroft, Clare, and Liakata [66] built a system that links a news article that reports or discusses a specific scientific research paper into this mentioned paper. The content of each news article was processed to extract possible names of authors and scientific organizations. Pairs of author names and organizations were then used as a search query, utilizing Microsoft Academic Knowledge, Scopus and Springer APIs, to retrieve an initial set of candidate scientific articles. Candidate articles were then scored using a mechanism that is based on the Levenshtein Ratio between the entity mentions in the news article and the authors names and affiliation in the scientific work. Finally, candidates were ranked according to their scores.

Recognizing named entities that are mentioned generally in any text and specifically in news articles, then linking these entities to entities in knowledge sources, has been an active research area in text contextualization. In fact, there is a research problem called

Wikification, in which, entities are supposed to be spotted in a text and then linked back to a Wikipedia page. Various systems have been introduced for named entity recognition, such as Stanford NER [67] and TagMe [68]. More recently, a number of systems were also proposed that performs both entity cognition and entity linking. Examples of these systems are Spotlight [38], Conerel [69] and Recognyze [70].

In [71], Rudnik, Ehrhart, Ferret, *et al.* produced mappings between news articles and events in Wikidata. To check if an article mentions a Wikidata event, the publishing date of the article was checked against the date of the event. Additionally, the article was checked to see if it mentions the event location in its lead paragraphs and title. Finally, the lead paragraphs of the article were checked for high similarity score with the Wikidata event type and title. For annotating the events within the articles, the authors used a predefined news events vocabulary. There have been additionally some work recently on detecting events within new articles that discuss incidents or stories, and cluster these articles according to the story it covers [14], [15]. This is to provide readers with links to other articles that report the same story.

To enrich news articles with data visualizations that are available on the web (Ex: maps, line graphs, bar charts) Lin, Ford, Adar, *et al.* [72] proposed a system that uses a learning to rank classifier to link news articles to visualization images that exist on Wikipedia. The candidate images were extracted from the webpages of named entities that were mentioned in the articles. To train the classifier, features such as the semantic relatedness score between the article's title and content and the image caption or its contained page were used.

Linking the news article's content to entities in knowledge sources or to visualizations will allow readers to gain more knowledge on the article's content, and accordingly help the readers contextualize it. Nonetheless, as some articles may not have named entities, additional techniques for contextualization are required. Moreover, the Wikipedia pages or the linked knowledge sources may not contain the up-to-date information that the reader seeks. The work reviewed above, however, in general can be complementary to the research done in news background linking using news articles as background links.

2.2. Content-based News Recommendation

The goal of news recommendation systems in general is to make reading suggestions to readers [73]. Recommendation systems are mainly implemented in two ways: collaborative or content-based filtering. In collaborative filtering [13], [74], [75], a reader is known to the news provider, and his past behavior along with similar behaviors made by other readers, are utilized to select what to offer to the reader next. In content-based filtering [76], the content and properties of the documents that the reader checked in the past are analyzed to recommend documents with similar or related content. Since our goal is to link articles solely based on their content, we opted to review here the work done for content-based recommendation of articles.

Lv, Moon, Kolari, *et al.* [76] developed a unified relatedness function that computes the probability of recommending one article to the reader of another article. The function was developed using a learning to rank model that took as input: the lexical similarity between the pairs of articles (measured using cosine similarity, BM25, language models with Dirichlet prior smoothing, and language models with Jelinek-Mercer smoothing), and the articles' connection clarity was computed using the KL-divergence between the topical distributions of both articles.

In[77], Desarkar and Shinde acquired a set of articles to recommend to a reader by adopting an optimization algorithm that optimizes some aspects between the query and candidate news articles. These aspects include the articles' topical similarity, its common entities mentions, its publishing date and popularity.

There has been also some research work on news recommendation that utilizes, in addition to the news content, some features about the reader that can be obtained, even if the reader is anonymous to the news source. For example, Chen, Lukasiewicz, Meng, *et al.* [78] utilized the current location of the reader as well as the location at which the news happened to perform the recommendation. Another example is the work done by Souza Pereira Moreira, Ferreira, and Cunha in [79], which regarded the articles that the anonymous reader is checking in a browsing session as his preference, and recommended the news of interest to him accordingly.

News background linking can be viewed as a special type of news recommendation, where the goal of the recommendation process is to recommend articles that the reader can read to increase his knowledge on the subject of the article at his/her hand. The above techniques though considered only the general relatedness between a couple of articles for recommendation. As shown from the background linking examples in Chapter 1, the articles that may be recommended to a reader because they are related to the query article may not contain the background information that the reader seeks. Hence, the criteria that judges the background relevance between articles needs to be further investigated and considered when recommending articles to readers.

2.2.1. Event Extraction and Information Threading

There has been some work on extracting information about events from large unstructured collections of news articles and link articles that discuss the same event together, forming a story line or an information thread to the newsreaders [14]. Assuming that a query article for news background linking is actually reporting an event, then articles that contain information about this event might be useful background links.

To find articles that discuss the same event, some research focused on extracting trigger event words or phrases from the news articles and use these keywords to relate articles that discuss the same event together [16], [18]. Other work focused on answering more specific questions like Who did What, When and Where from the articles, then use these questions answers along with the temporal relations between the articles to link it together [80]. Further, some research focused on building supervised models that classify the sentences within articles based on the event entities it mentions, then use these sentences to illustrate the evolution of events over time [15].

It is important to note here though that the information acquired through information threading might not be the only background information needed for a query article that discusses an event. For instance, a useful background information may be obtained from reading about similar events as the one mentioned in the query article. In addition, a useful background article may not at all mention or discuss the event reported in the query (as shown in the examples in Chapter 1. Furthermore, a query article may discuss general subjects and may not at all mention a specific event. Therefore, this line of research, while it may benefit newsreaders, can't be solely and effectively used to address the news background linking problem.

2.3. Long Documents Reranking

Since news articles are considered relatively *long* documents and our goal is to rerank the candidate set of news articles obtained through the full-article search approach, we opted to review here the work done for direct long documents reranking.

Direct document reranking is usually used in information retrieval systems to enhance the ordering of a set of documents initially retrieved for a specific search query. It mainly involves reordering these candidate documents given some preference criteria. While earlier work for direct document reranking adopted lexical based methods such as content-based analysis of documents [81] or document clustering [82] , the vast majority of the recent work that we reviewed for long documents reranking adopted Pre-trained Language Models (PLMs) to represent the query and the documents in the semantic space, and rerank the documents given its semantic similarity with the query.

PLMs have proven to be effective for a number of tasks [62], [83]. When dealing with long documents though, researchers are faced with the context length restriction imposed by these models. To overcome this limitation, many approaches have been proposed recently. A number of researchers broke the candidate documents into chunks of texts (passages), and aggregated relevance signals with the query from those passages. Dai and Callan [84] split each document using a sliding window of 150 words passages with a stride of 75 words, then used BERT [62] to predict a score for each passage given the query. Finally, they calculated the document scores as the score of its first passage, the score of its best passage or the sum of scores of its passages. Yilmaz, Wang, Yang, *et al.* [85] split the document into sentences, used BERT to infer the relevance of each sentence to the search query, then applied a weighted sum of the top three scored sentences to compute the document scores.

Jiang, Xiong, Lee, *et al.* [86] explored building a transformer model that adopts sparse attention in order to be able to process long text documents for different tasks, including documents ranking. In addition to global query tokens, additional local attention tokens were added at the beginning of each sentence in the candidate document in order to capture the document's relevance to the query. Very recently, Gao and Callan [87] suggested to first independently encode the query and the document using two Transformer encoders, then use another interaction module/Transformer that applies self attention on the query tokens, and a cross attention between the query tokens and all the document tokens. The author further suggested splitting the document into chunks, obtain an independent representation for each chunk, then concatenate those representations as a document representation before feeding it to the interaction module. In their experiments, the authors used a chunk size of 512 tokens, with maximum 6 chunks per document. Li, Yates, MacAvaney, *et al.* [88] proposed to aggregate the passages representations into a document representation before calculating the document score. Post using BERT to obtain a representation for each passage, the representations of passages are concatenated in order and fed to a transformer layer to obtain an entire document representation. This representation passes through a feed forward network to compute the document score. In their work, documents were split into maximum 16 passages using a sliding window of 225 tokens with a stride of 200 tokens.

All the above models experimented with datasets that contained relatively short queries such as questions, documents titles, or sometimes document descriptions. Accordingly, it focused on how to bypass the context length limit restriction of transformer-based models for only the relatively long candidate documents. In news background linking, however, the query article is also a long piece of text that needs to be represented to the reranking model. Accordingly, there is a need for models that are able to perform long to long document matching.

Recently, there has been some work on building supervised models that specifically aim to overcome the limits in models that apply short to long documents matching. Yang, Zhang, Li, *et al.* [89] proposed SMITH, a dual encoder matching model that takes a pair of long documents as input and creates an independent representation for each document in the documents pair with a goal of increasing the pair similarity score. To create a document representation using encoders with limited input length such as BERT, a document in SMITH is split into blocks of pre-defined size that contain multiple sentences. The model was trained with a masked block task, and was tested on matching Wikipedia pages of similar outgoing links. Adopting similar work, Pang, Lan, and Cheng [90] used PageRank during the document representation learning process to filter out noisy sentences and words from documents, to increase the efficiency of the matching process. The proposed model was tested on a Chinese news corpus, where pairs of news articles were considered positive samples if they share the same news story. Jha, Rakesh, Chandrashekar, *et al.* [91] introduced CoLDE, a contrasting learning model that is trained using a pair of documents with the goal of maximizing the similarity between sections of the same document and minimizing the similarity between sections of different documents. The model hence breaks a document into two sections during the encoding process, then each section is further split into 512 tokens chunks. Post chunks representation through BERT encoders, an LSTM is used to generate section representations for each document. CoLDE was tested on datasets that include pairs of scientific articles commonly cited, Wikipedia articles that share outgoing links, and patent documents that have similar topics.

All of the above models that were proposed for long-to-long document matching were tested for binary relevance, and for tasks other than news background linking. Whether they will perform as effective for news background linking that require multi-class classification is still an open research for investigation.

2.4. Conclusion

The above literature review has emphasized the need for a more effective news background linking system that considers other relevance factors in addition to the lexical similarity factor used by full-article retrieval approach. We showed that a number of systems attempted to adopt supervised models for news background linking, yet they were not reported to be more effective than the simple full-article retrieval approach. From reviewing the proposed systems for specifically news background linking, one can further notice that the notion of background relevance is not yet quite understood nor captured by any of the proposed systems for news background linking, as most of these systems didn't identify the actual criteria that judges the background relevance while building their systems. This leaves finding these criteria and harnessing it for building effective news background linking systems, a research gap. Our review here additionally showed that there is still a challenge in building an effective reranking model for candidate background links retrieved using the full-article retrieval approach, as both the query article and the candidate background links are relatively long documents that need to be represented to the reranking model. Following, we show our work in understanding the notion of news background relevance, and how we adopted this in our proposed background linking approaches.

CHAPTER 3: QUALITATIVE ANALYSIS OF BACKGROUND RELEVANCE

News background linking is a relatively new problem in the research field. As shown in the previous chapter, most of the systems proposed to address this problem adopt techniques that doesn't actually depend on understanding the exact notion of relevance in this specific problem, and why a specific background document might be more useful to the reader of a query article than another document. We argue that there are some characteristics that are inherent to the nature of background relevance and that need special attention not properly captured by the existing proposed techniques to address this problem. In this chapter, we show our qualitative analysis on a sample of query articles and their corresponding judged articles released by TREC. The objective of the study is to better understand the notion of background relevance. More precisely, we aim to determine the criteria that distinguish background articles from non-relevant ones. Based on the identification of that criteria, more effective and efficient background linking approaches can be developed.

There are many criteria that distinguish one news article from another. Among these criteria is the article's main topic, its discussed subtopics, whether it is a short news report or a feature article discussing a specific topic in details, whether or not it is *event-triggered*, and whether it reports hard news or soft news. Event-triggered articles are the ones that are mainly triggered by the occurrence of an *event* in a specific time (e.g., a terrorist attack incident). Non event-triggered articles, however, discuss more general topics (e.g., the cost and benefits of college education). Hard news often covers subjects such as politics, crimes or economics. Soft news, however, generally aim to entertain or advise the readers. It discusses subjects such as art, sports and lifestyle.

This chapter aims to address the research question:

RQ1: *What criteria in both the query and its candidate background articles affects the background retrieval relevance?*

To address this questions, we specifically aim to answer the following sub questions:

- **RQ1.1**: Do the background and query articles have a *common* main topic?
- **RQ1.2**: What is the relation between a query article and its background article in terms of the topics discussed in each?
- **RQ1.3**: Should we handle the query articles that are driven by the occurrence of events *differently*?
- **RQ1.4**: To what extent, does the lexical similarity between the query and background links affect the relevance in this problem?

Through the rest of this chapter, we address each of the above questions in detail. While, our qualitative analysis of news background relevance here is based on our own *subjective* view of the articles, we believe that the insights drawn through our discussion of the results can help develop more effective background retrieval models. In fact, this study opened the door to one of our proposed techniques for an effective news background linking process as shown in the following chapters of this dissertation. Following, we give details on the Washington Post Dataset specifically released for the news background linking problem in Section 3.1, with details on the selected annotated samples. Then we explain our annotation process in Section 3.2. Finally, we present and discuss our results in Section 3.3.

Year / Relevance Level	4	3	2	1	Total Relevant	Non-Relevant
2018	106	164	584	1189	2043	6465
2019	273	431	660	1677	3041	12614
2020	50	132	631	1603	2416	15348
2021	261	266	582	1448	2557	10351
Total	690	993	2457	5917	10057	44778

Table 3.1: The number of judged articles in the Washington Post Dataset, in each relevance class, given each year of running the background linking task

3.1. Dataset

The only available dataset for news background linking is the **Washington Post dataset** released by TREC for this task. The latest version of the dataset contains 728,626 entries published from January 2012 through December 2020. Four sets of 50, 57, 49 and 51 query articles were released in each year of running the task at TREC from 2018 to 2021 respectively. Each article in the dataset is formatted as a JSON object, which is broken into a title and multiple content paragraphs with HTML text. Each article is associated with metadata such as the title, author URL, and publishing date. The dataset contains news articles with the type ”Opinions”, ”Letters to the Editor”, or ”The Post’s View”. These articles are considered non-relevant in the background linking task.

Each pooled article from participants to the background linking task was given a background relevance level given the scale below, taken from TREC description of the news background linking task [3]:

- **0**: The linked document provides little or no useful background information.
- **1**: The linked document provides some useful background or contextual information that would help the user understand the broader story context of the query article.
- **2**: The linked document provides significantly useful background
- **3**: The linked document provides essential useful background.
- **4**: The linked document MUST appear in the sidebar otherwise critical context is missing.

Table 3.1 shows the statistics of the background links within the dataset, in each relevance level, and in each year of running the background linking task within TREC. Note that the relevance levels representations within the Washington Post dataset is highly skewed. For example, the highly relevant background articles (level 4) are very few compared to the other lower but also relevant articles. Additionally, the total number of relevant articles is almost quarter the number of non-relevant ones. This shows the challenge in building a news background linking system that can effectively retrieve the useful background links.

Evaluation Measure : As mentioned before, the goal of news background linking is to present readers with links to the *most* useful resources that provide the contextual information on the content of the query article. Compared to other cases where a user in a search engine is presented with a relatively big list of retrieved documents to select from, readers of news articles are usually presented with very few links at the tail of the news article's page. Hence, nDCG with depth 5 is used as the primary evaluation measure for effectiveness in this problem as it allows each retrieved link to have a graded relevance as shown above. The gain value for each link/article is calculated as 2^r, where r indicates the relevance level given the scale shown above.

Annotated Samples In our qualitative analysis for background relevance, we randomly selected 25 query articles from the 2018 query articles of the Washington Post dataset. For each chosen query q, we sorted the background articles in descending order of the relevance level, excluding the articles of zero relevance level (i.e., non-relevant), and then chose the top 5 of them to annotate (if any). To ensure that our sample covers all relevance levels, we additionally added one randomly-sampled judged article from each relevance level that was not represented in the top 5 (if any). For non-relevant articles, we sampled two judged articles that mostly appear as potentially-relevant from their titles. Overall, we annotated **227 articles**, **25 query articles and 202 judged articles** (an average of 8 per query) distributed as follows: 51 judged articles of relevance 4, 35 of relevance 3, 33 of relevance 2, 33 of relevance 1, and 50 of 0 relevance.

3.2. Annotation Process

For each query article, we identified the following:

- **Main topic**: This is defined as the main subject that the article's author seeks to introduce to the reader. To identify the main topic, we assumed that authors adopted the widely-known inverted pyramid structure [92] when they wrote their articles, where the essential and most attention-grabbing elements are introduced first. Accordingly, we considered the title of the article as well as the first one to two paragraphs as the sources for identifying the main topic.

- Whether the title is indicative enough of the main topic.

- **Subtopics**: We define subtopics as aspects that are related to the main topic and that expand its discussion from a particular view. For example, if the main topic is about "an accident of a black bear attacking a camper", subtopics can be "similar black bears attacks", or "food searching habits of black bears". Subtopics are usually mentioned in later paragraphs in the article. In our annotations, we identified up to 4 subtopics for each judged article.

- Whether the article is *event-driven* or not: Event-driven articles are the ones that are mainly triggered by the occurrence of an *event* in a specific time (e.g., a terrorist attack incident). Non event-driven articles, however, discuss more general topics (e.g., the cost and benefits of college education).

- **Total Judged-Relevance:** We computed the total judged-relevance score as follows: $R(q) = \sum_{a \in J(q)} r(a, q)$ where $r(a, q)$ indicates the judged relevance level

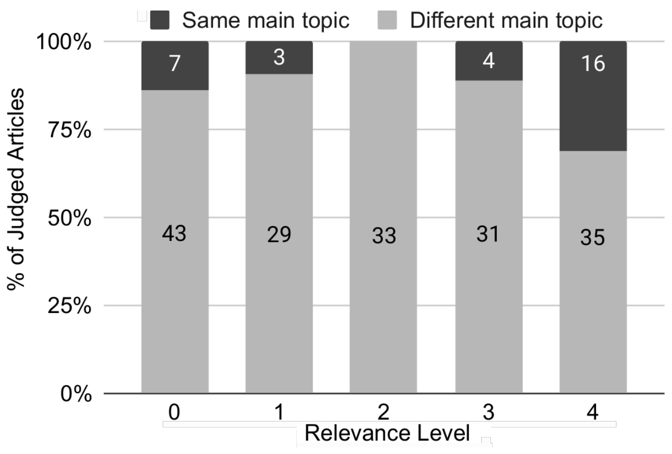


Figure 3.1: Percentage of the judged articles having the same or different main topic as the query articles.

of a with respect to q. It gives a rough indication of how easy (difficult) for a background retrieval system to answer the query, based on the assumption that the more (less) background articles with high relevance levels for a query article are, the easier (harder) for the system to find relevant articles for it.

For each judged article, we identified the main topic and subtopics, as described above, in addition to the following:

- Whether it has the same main topic as the query article.
- **Discussion of Query Topics**: Here, we identified the extent to which the judged article discusses the topics (including both the main topic and subtopics) that are of the query article. For this, we assigned a score from 0 to 4, where 0 indicates no subtopics are discussed, 1 indicates it just mentions one subtopic with no further details, 2 indicates it mentions at least two subtopics with no further details, 3 indicates it discusses one subtopic with more details, and 4 means it discusses at least two subtopics with more details.
- **Addition of New Subtopics**: Here we identified the extent to which new subtopics mentioned in the judged article are useful to comprehend/contextualize the query article. For this, we assigned a score from 0 (indicating none of the newly-added sub-topics are useful) to 3 (indicating the newly-added sub-topics are very useful).
- In case the query article is event-driven, whether the judged article mentions that event or not.

3.3. Results and Discussion

We discuss in this section, the results of our analysis addressing each of our former identified research question, and providing insights regarding the notion of background relevance in the context of each question.

To address $RQ1.1$, Figure 3.1 shows the percentage of judged articles that have the same (or different) main topic as the query article at each relevance level. We notice

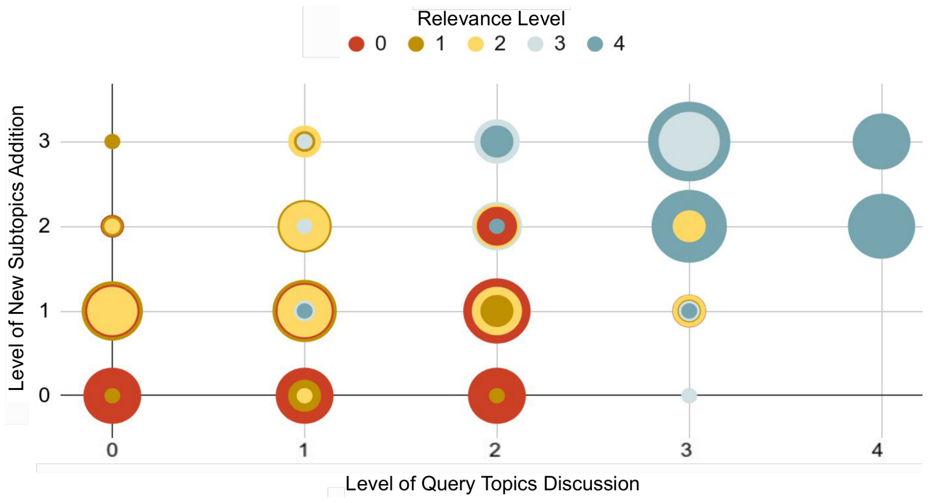


Figure 3.2: Relation between query topics discussion, new subtopics addition, and relevance of judged articles.

that the vast majority of the background articles, regardless of the relevance level, do not have the same main topic as the query article. Moreover, some articles have the same main topic but judged non-relevant. Furthermore, we found that, in 25% of the annotated query articles, the title is not at all indicative of the main topic. For example, the query article titled *"The solution to Climate Change that has Nothing to do with Cars or Coal"* focuses mainly on the effect of deforestation on the environment. We further noticed that, for event-driven articles, 41.7% of the titles do not even refer to the event. Consequently, methods that utilize the title or the first paragraph to form a search query or to construct training samples (e.g., [44]) are subject to lower performance compared to utilizing the full content.

Motivated by our observation that having the same main topic as the query article is not the key to background relevance, we investigated the relation between the other topics discussed in both the query and judged articles to address $RQ1.2$. Figure 3.2 illustrates the relation between the discussion level of query topics, level of new subtopics addition, and relevance level, where bigger bubbles indicate more judged articles of the corresponding levels.[1] It is clearly noticed that highly-relevant background articles (colored in cyan) simultaneously discuss, to a great extent, the topics of the query article *and* add new subtopics that are very useful in comprehending it. In fact, the correlation between the relevance level and levels of query topics discussion and new subtopics addition are 0.46 and 0.64 respectively; however, the figure indicates that each separately is not enough for highly-relevant background articles.

We next investigate some characteristics pertaining to the event-driven queries in the context of *RQ1.3*. Figure 3.3 depicts Total Judged-Relevance scores for event-driven queries compared to non-event-driven ones. The score for each query is also broken down into two sub-scores over the corresponding judged articles published before and after the query article. Comparing the average scores of the two types of queries indicates that event-driven queries are generally *harder* to handle than others. Furthermore, the figure shows that most of the background articles are published before the query article

[1]We added small noise in few cases to ensure no bubble of any level is hidden.

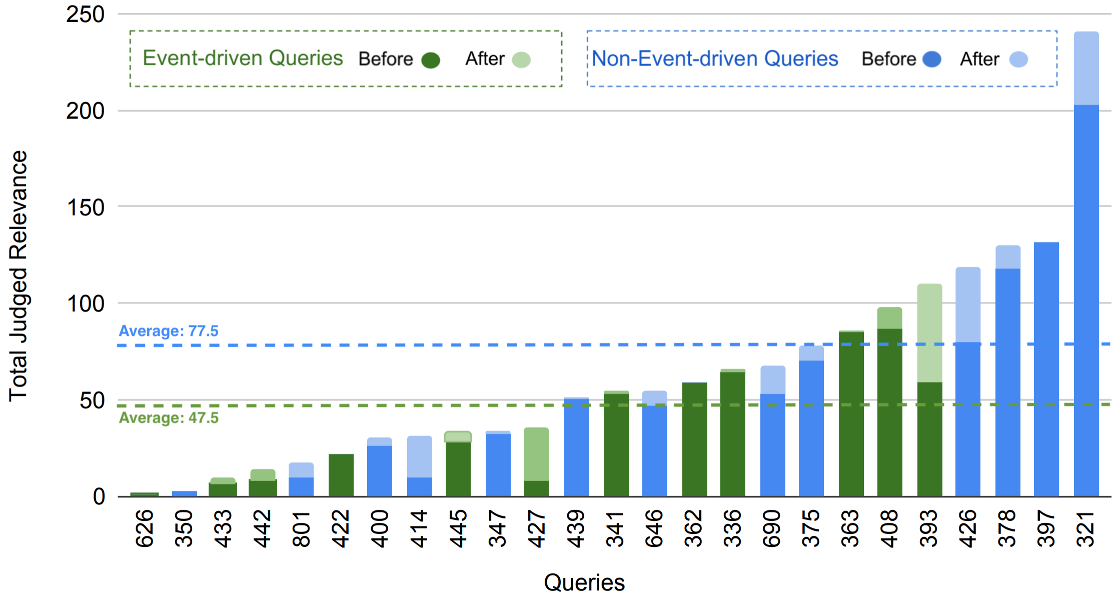


Figure 3.3: Total judged-relevance scores of query articles.

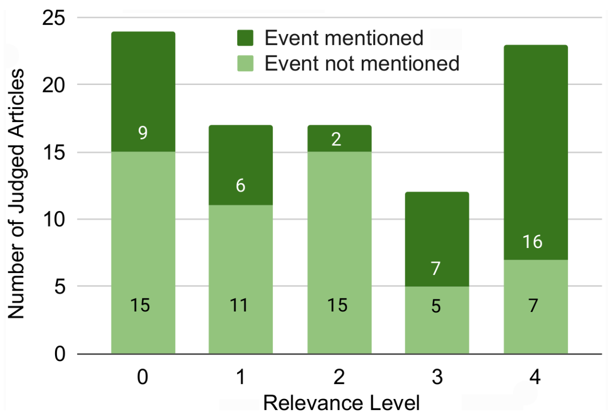


Figure 3.4: Distribution of the judged articles of event-driven queries over relevance levels.

for both types; however, for event-driven query articles, some background articles were unexpectedly published after the query article itself. Upon investigation, we found that some articles are based on an event that is still to happen in the future. For example, query 427 titled "*Here's What Happened to People Who Tried to Watch a Solar Eclipse Without Special Glasses*" talks about an eclipse that actually occurred a month later. Hence, all articles that discuss subtopics related to the eclipse, and were published after this article but before the actual eclipse, were judged relevant.

Focusing on event-driven queries, Figure 3.4 explores the relation between the relevance level of their judged articles and whether they mention the event or not. Two major observations are revealed. First, as expected, the majority of the highly-relevant background articles (3-4 relevance levels) do indeed mention the event. However, a considerable number of them do not. We found that those articles often discuss the context at which the event appears and in many cases, they add knowledge on the future

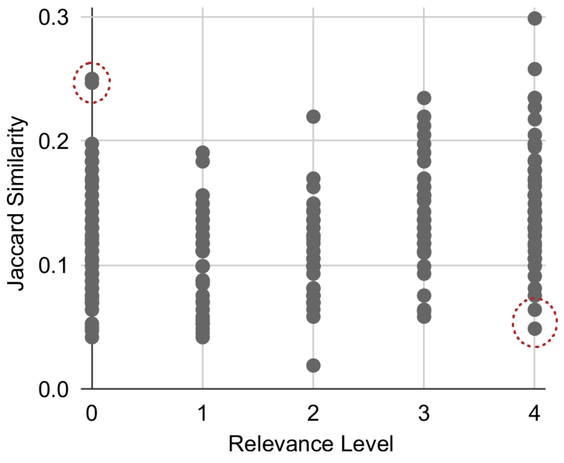


Figure 3.5: Jaccard similarities between query articles and judged articles at different relevance levels.

consequences of the event. For example, the article titled "*A bear mauled a cyclist in the Alaskan woods*" was found to be relevant to a query article that details an incident of a bear attacking a camper in Colorado. Although this article does not mention the exact event of the query article, it talks about deadly attacks of bears, which is clearly a context of the query article. Second, 9 articles mentioned the event, but they were judged non-relevant. We found that they neither discuss the topics of the query article in detail nor add new useful subtopics, indicating that just mentioning the event is not sufficient for background relevance.

As mentioned in Chapter 2, the most effective method for news background linking follows a simple ad-hoc based retrieval approach that is based mainly on measuring the lexical similarity between the query article and the background links. The optimal reranking of the candidate links obtained through this method however shows big chances for improving the retrieval effectiveness. We can conclude from this that some background articles do not exhibit the highest lexical similarity with the query article, yet they are relevant and vise versa. It is intriguing though, to investigate the lexical similarity between the query articles and their judged articles in our annotated samples to see to what extent does the lexical similarity affect relevance in news background linking ($RQ1.4$). For that, we have computed Jaccard similarity (as an example measure of lexical overlap) between the query article and the judged articles using the 100 most-frequent terms in each (after stop words removal, lower-casing, and stemming). Figure 3.5 is a scatter plot of the computed similarities over our annotated set of judged articles distributed over different relevance levels.

The figure indicates several interesting observations.[2] First, higher levels of relevant articles have, on average, higher similarities with their corresponding query articles. However, many of them do not exhibit high similarity with the queries. Two examples are the ones at the right-bottom corner of the plot. In fact, the correlation between similarity and relevance levels of the background articles (i.e., excluding non-relevant ones) is 0.43, which is positive but not very strong, indicating that lexical similarity is

[2]We found similar patterns when Jaccard similarity is computed over the full text of the articles instead of just the 100 most-frequent terms.

a relevance factor but not sufficient. Second, we notice that the similarity scores are generally low, ranging from 0.01 to 0.3, suggesting that effective background linking methods should go beyond traditional models that are based on only the lexical-overlap between the query article and its background links. Finally, some non-relevant articles have relatively-high similarities. A couple of examples are the ones in the top-left corner of the plot, reassuring the earlier suggestion.

3.4. Conclusion

To sum up, in this chapter, we showed how we analyzed a sample of the query articles and their background links in an attempt to understand the notion of background relevance, and build more effective news background linking systems. Through our analysis, we believe that an effective news background links might be built by distinguishing "event-driven" query articles from other query articles and treat each differently during the linking process. We hypothesize that to find the relevant background links for an "event-driven" article, one needs first to identify the discussed event, then focus on finding other articles that explain context information around the occurrence of specifically this event (such as more information on event participants, other earlier or similar events that might led to it, why this event occurred, . . . etc.). We also observed through our analysis that an effective news background linking system may consider the identification of the specific topics discussed within the query article and aim to find background links that expand the knowledge on these specific subtopics with more knowledge or other related new subtopics.

In the next chapter of this dissertation, we will detail our first proposed news background linking approach. The proposed approach does not directly depend on the insights drawn from our qualitative analysis of relevance shown in this chapter. Yet, it benefits from the intuitive observation, confirmed as shown above by our analysis, that lexical similarity is not the only relevance factor for news background linking, and that some highly relevant background links might have very low lexical similarity with the query article. Hence, it integrates the lexical and the semantic matching signals between the query article and its candidate links for an effective background linking process. Later, in Chapter 5, we will show how we actually used other insights drawn from this chapter to build our second proposed news background linking approach.

CHAPTER 4: ZERO-SHOT RERANKING WITH ENCODER MODELS

In this chapter, we present our work on how to increase the effectiveness of the full-article search method through the reranking of its retrieved background links. As shown from our qualitative analysis for the notion of background relevance in the previous chapter, some news articles exhibit very high relevance with the query article even though they are written with different vocabulary. Accordingly, we must investigate other relevance signals than the lexical similarity. We hypothesize that for some query articles that do not report specific news events but rather discuss general subjects, deviating from the query article's exact vocabulary is actually essential in order to find new background links that are not just duplicating the knowledge in the query article. In other words, we need to go beyond the surface-level information that exists in the query article in order to find what actually benefits the reader.

Although some research studies adopted semantic matching in news background linking as shown in Chapter 2, none showed significant improvement over the simple lexical matching between the query article and the candidate link. Accordingly, in our work, we propose to integrate both the lexical and semantic relevance signals for more effective news background linking. As training data is relatively scarce for the task, we opt to adopt a *zero-shot* setup that leverages the transfer learning capabilities of existing pre-trained Transformer-based encoders for effectively representing spans of text in the semantic space. We propose to split the query article into multiple passages (potentially capturing its subtopics), encode each passage and match it with the candidate background link, and finally aggregate the matching scores for better semantic ranking.

To work in the semantic space, many pre-trained Transformer-based models were recently proposed [22], [24], [25], [93]; each of which has a restricted length of the input sequence and was trained using different text types. While some were trained given sentences as input, for example, others were trained on longer sequences, such as paragraphs, Wikipedia articles, or full news articles. Those models were shown to be effective in different language understanding tasks, such as text classification and document retrieval [94], page linking [95], sentence similarity and sentence clustering [93], [96]. However, to our knowledge, they were not effectively adopted for the specific task of news background linking before. Accordingly, we aim to study the effectiveness of those models in semantic matching between the query article and the candidate links for better news background linking.

More specifically, in this chapter, we address the research question:

RQ2: *Can we adopt Encoder models for a semantic based reranking of the links retrieved through the full-article search approach?*

To address this question, we aim to investigate the answers to the following sub questions:

- **RQ2.1**: Given the input sequence length restrictions of existing pre-trained encoder models, what is the best model to use, without the need for further pre-training or fine-tuning, to represent news articles for semantic-based news background linking?

- **RQ2.2**: Can we integrate both the semantic and lexical relevance signals for an effective reranking of the candidate background links?

- **RQ2.3**: Do the query articles vary in their need for the lexical and semantic relevance signals for reranking the candidate background links?

To address the above questions, we propose a *zero-shot* approach that does not need any labeled data. We rely on the recent pre-trained dense encoder models to semantically represent the news articles. For a given query article, we then apply semantic matching to rerank a candidate set of background links obtained initially by sparse retrieval. Our results show that the best dense representation of news articles for our problem is a sentence-level hierarchical aggregation using a sentence-based encoder. Regardless of the representation model though, we found that an effective reranking of the candidate links cannot be achieved solely by semantic matching; we show that a normalized sum of the semantic and lexical matching scores can be used to better rerank the candidate set. This simple zero-shot integration of relevance signals exhibited a statistically significant improvement over the state of the art (SOTA) approach for the news background linking problem. Our analysis of the different query articles further revealed that the degree by which a semantic relevance signal is used should be query-based. If this degree is predicted accurately, further significant performance improvement can be achieved.

In the rest of this chapter, we present first our preliminary work in supervised news background linking in Section 4.1. We then show the details of our proposed framework for the zero-shot encoder-based news background linking approach in Section 4.2, including the pre-trained models we adopted to transform the news articles to the semantic space, along with a description of how we obtain a semantic representation for a news article given the input length restriction of each model. Later, in Section 4.3, we show our experimental setup and results, as well as a detailed discussion of the answers to our research questions.

4.1. Preliminary Work on Supervised Background Linking

As shown in Chapter 3, the lexical similarity between the query article and its background links is not the only relevance signal in news background linking. That is being noted, we attempted to build a supervised model that can predict background relevance for the news background linking process. Precisely, we investigated with building a multi-class classification model that can take the query article and one candidate background link as input, and outputs the relevance level from within the relevance scale given in the Washington Post dataset (i.e., output a value from 0 to 4, where 0 means that the background link is not useful to the query article and 4 means that this background link must be presented to the reader of the query article, otherwise, critical information will be missing).

To build the proposed supervised model, we followed two architectures. In the first, we initially constructed a semantic representation for each of the full contents of the query article and the candidate background link. We then concatenated these representations, using different concatenation options, to form an input to the learning model that was trained later to predict the required relevance level. To obtain the representation of a news article, we experimented with a state of the art Transformer-based model (BigBird [24]), that takes relatively long input context (4096 tokens) as input, and was shown to be effective in other document classification and document matching tasks [94]. We further experimented with a Transformer-based model that takes sentences as input (Sentence-Bert [22]), and aggregated the news article's sentences representation to obtain an overall representation for the news article. To build the classification model, we experienced with supervised machine learning models, such as SVM, , logistic regression and decision trees, among others.

In our second attempt to build the news linking classification model, we experimented with fine-tuning Bigbird for news background linking. We split the input to this model into two parts with a separation token. The first part of the input takes the query article tokens and the second takes the candidate article tokens, post truncation if required. We then used the latent representation of the global CLS token at the beginning of the tokens stream to be input to a classification head that outputs the predicted background relevance level.

We trained and tested different classification models adopting our suggested architectures given the query articles and its judged background links in the Washington Post dataset. For each query, we sampled all judged relevant background links as positive samples. We further randomly selected an equal number of non-relevant background links to be negative samples. We divided the data samples among the training, development and testing sets, so that each set has a number of unique query articles.

Unfortunately, all of our attempts to build such a supervised classification model for news background linking were not successful. More precisely, the models failed, during the testing process, on average, to predict the relevance classes with an acceptable accuracy that leads to an effectiveness that is higher to the one achieved by the full-article retrieval method. We believe that this is due to the scarce and unbalanced data samples in the different relevant classes, as shown before in Table 3.1. In addition to the scarce data, there is also the challenging definition of the relevance scale for background linking, as shown in Chapter 3, that results in small gaps between consecutive relevance levels, making it difficult for a classification model to obtain a clear-cut for the boundaries between the different relevance classes.

Considering the challenges we experimented in training a supervised model for news background linking from scratch, we opted to investigate the supervised models in a zero-shot way, adopting it only to represent the news articles in the semantic space, thus harnessing its already seen potential in capturing the semantics of text, then used semantic similarity measures to determine the candidate's background link relevance to the query article. Following, we will show our proposed framework for this zero-shot news background linking approach in details.

4.2. Proposed Framework

In this section, we present our proposed framework that addresses the problem of background linking of news articles through the integration of both the lexical and semantic relevance signals between an input query article and its candidate background links. We start by describing the general structure of this proposed framework, then we delve into details on how this framework actually works.

Figure 4.1 shows the general structure of our proposed framework. It takes an input query article q for which we want to retrieve useful background links, and eventually provide a ranked list B of k background articles. The framework has three stages: (1) lexical retrieval, (2) semantic matching, and (3) integration of lexical and semantic signals.

At the first stage, denoted by *Lexical Retrieval*, we reduce the space of potential background links by retrieving, from the news articles collection, a candidate list B of k news articles having the highest *lexical* similarity (denoted as $L(b, q)$) of a candidate

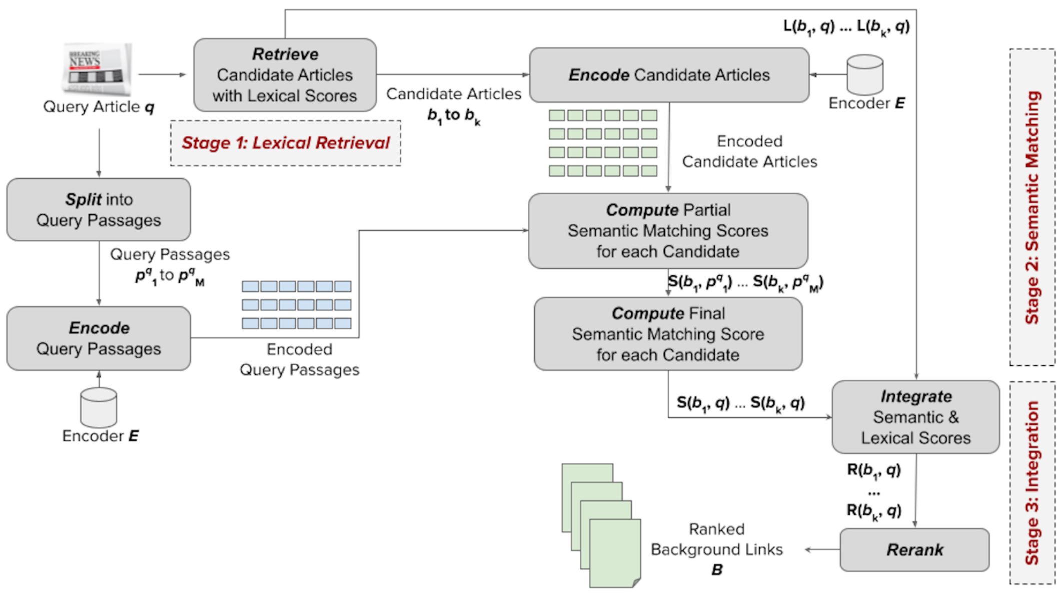

Query Article q
Retrieve Candidate Articles with Lexical Scores
Stage 1: Lexical Retrieval
Candidate Articles b1 to bk
Encode Candidate Articles
Encoder E
L(b1, q) ... L(bk, q)
Encoded Candidate Articles
Split into Query Passages
Query Passages p^q_1 to p^q_M
Compute Partial Semantic Matching Scores for each Candidate
S(b1, p^q_1) ... S(bk, p^q_M)
Encode Query Passages
Encoded Query Passages
Compute Final Semantic Matching Score for each Candidate
Encoder E
S(b1, q) ... S(bk, q)
Integrate Semantic & Lexical Scores
R(b1, q) ... R(bk, q)
Ranked Background Links B
Rerank
Stage 2: Semantic Matching
Stage 3: Integration

Figure 4.1: Our proposed framework for reranking background links using encoder models.

article b to the query article q. This is done using a basic sparse retrieval model (e.g., BM25). At the second stage, denoted by *Semantic Matching*, we *semantically* match the query article q against each candidate b to obtain semantic matching scores. To achieve that, we first split q into m query passages $p^q_{1..m}$. Using a Transformer-based encoder model E, we then encode each passage p_i and each candidate article b into embedding vectors. Next, we compute semantic matching scores between each candidate article and each passage, denoted as $S(b, p^q_i)$. The passage matching scores for each candidate are then aggregated into a semantic matching score $S(b, q)$. Finally, at the *Integration* stage, the lexical and semantic matching scores are combined into a final score $R(b, q)$ that is used to rerank the candidate background articles. Note that the framework proposes a zero-shot reranking that does not need any labelled data; it counts on the pre-training of the encoder to represent the candidate links in the semantic space and integrates the lexical (sparse) and semantic (dense) retrieval scores in an unsupervised manner. We discuss the details of all stages of the framework in this section.

4.2.1. Stage 1: Lexical Retrieval

This stage is quite simple. Its aim is to obtain a *candidate* list of news articles that most probably contains relevant and useful background links. For this, the text of each news article in the given articles collection is initially preprocessed (lowercased and stop words are removed), then it is indexed into an inverted index. For a given query article, the title of the article is concatenated with its text to compose a search query, which is preprocessed before being issued. Any robust sparse retrieval model (such as BM25 [97]) can be used to retrieve the candidate background links. The retrieval process often includes pruning articles that are known ahead to be non-relevant (such as the ”Opinion” and ”Letters to the editor” articles in the Washington Post Collection).

4.2.2. Stage 2: Semantic Matching

In the semantic matching stage, we first split the query article into shorter passages, potentially capturing its subtopics. Then we encode each passage and each candidate article into the semantic space. Finally, we match each candidate against each query passage, and aggregate the passage scores per candidate to obtain a candidate score. We detail all of these steps in this section.

Splitting the Query Article into Passages As discussed earlier, the goal of this stage is to match each candidate background article semantically against the query article. Since the query article has multiple subtopics and the candidate article might have the background knowledge or provide the context for one or more of these subtopics, we propose to split the query article into a number of shorter passages, called query passages, aiming for each passage to represent a subtopic. Then we semantically match each query passage with the candidate article. We hypothesize that a subtopic is addressed in a few consecutive paragraphs. Hence, we split the article into individual paragraphs, and use a sliding window of two consecutive paragraphs (with a step size of one paragraph) to construct query passages that each, hypothetically, represents a subtopic. A query passage is the concatenation of the enclosed paragraphs within the sliding window. We consider the title of the article as the first paragraph.

Contextual Representation of News Text To represent both the query passages and the candidate background articles in the semantic space, we leverage Transformer-based encoder models that were pre-trained for several Natural Language Processing (NLP) tasks. We experimented with a number of recent models based on the following criteria:

- Models that were pre-trained for text contextualization on datasets that included news articles (e.g., Sentence-BERT and BigBird).
- Models that were fined-tuned on pre-trained language models to learn representation of text that included links to other text sources (such as hyperlinks between documents in Wikipedia), counting on the closeness of this task to news linking (e.g., LinkBERT).
- Models that were pre-trained to learn representation of text that included named entities, since news articles often include references to persons, organizations, or locations (e.g., EASE and ERNIE-2.0).
- Models that were shown to be effective for tasks that depend on acquiring a good low-dimensional representation for long documents, such as document matching and document classification (e.g., LongFormer).
- Models that showed effective performance when evaluated on out of domain tasks (e.g., PromCSE).
- Models that have available open-source pre-trained encoders.

Given the above selection criteria, we further categorized the models into three categories based on the length of their training samples. We experiment with models that were trained on sentence-level input, limited-size input (512 tokens), and long-text input (up to 4096 tokens), as shown in Table 4.1.

Our goal is to test the ability of those models in capturing the core context presented within each candidate news article to allow for effective news background linking. In other words, a good model should capture the valuable information reported in an article and hence provide a representation that enables effective semantic matching between the query article and its relevant background links. We use those encoder models in a *zero-shot* setting; no labeled data for news background linking was used to further pre-train or fine-tune any model. This allowed us to test the ability of each model to derive a representation for news articles that can be used for effective new background linking, without specifically being trained for this task.

Model	Encoder Source	Input
SBERT	sentence-Transformers/all-mpnet-base-v2	Sentence
EASE	sosuke/ease-roberta-base	Sentence
PromCSE	sup-PromCSE-RoBERTa-large	Sentence
LinkBERT	michiyasunaga /LinkBERT-base	Paragraph
Ernie-2.0	nghuyong/ernie-2.0-base-en	Paragraph
BigBird	google/bigbird-roberta-base	Long Text
LongFormer	allenai/longformer-base-4096	Long Text

Table 4.1: Pre-trained encoder models used in our experiments along with the type of input they were pre-trained on.

Following, we discuss briefly how each model we selected was pre-trained, and how we use it to obtain a dense representation for both the query passages as well as the candidate articles.

- **Models trained on short-text input (Sentences)**
  - **Sentence-BERT SBERT**) [22] was proposed as a extension of BERT [62] to derive semantically meaningful sentence embeddings. Recently, Sentence Transformers have been proposed as a family of **SBERT** models as encoders for short text (sentences). Each of those models was trained on a different dataset from a different domain. The model we experiment with in this study, all-mpnet-base-v2, was fine-tuned from the pre-trained microsoft/mpnet-base model on a dataset of 1 billion sentence pairs with self-supervised contrastive learning objective that aimed to identify pairs of contextually similar sentences. As suggested by the framework, we obtain the sentence embeddings using this model through the average pooling of the sentence token embeddings.
  - **EASE** [93] similarly aims to capture semantics of sentences, while specifically focusing on identifying the information of related entities in sentences. Wikipedia sentences with hyperlinks treated as related entities were used for training this model. Similar to SBERT, we obtain the sentence embeddings through the average pooling of the sentence token embeddings.

- **PromCSE** [96] aims to overcome the drop in performance faced when evaluated on out-of-domain tasks. To do that, it prepends soft prompts (limited sequence of continuous vectors) at each layer of the pre-trained language model (BERT or ROBERTA) during the fine-tuning process, while freezing all the other model parameters. Since it showed SOTA performance when evaluated in new domains, we opted to experiment with it for news background linking. As recommended,[1] we obtain a sentence embedding from the [CLS] token hidden state from the last layer of the model.

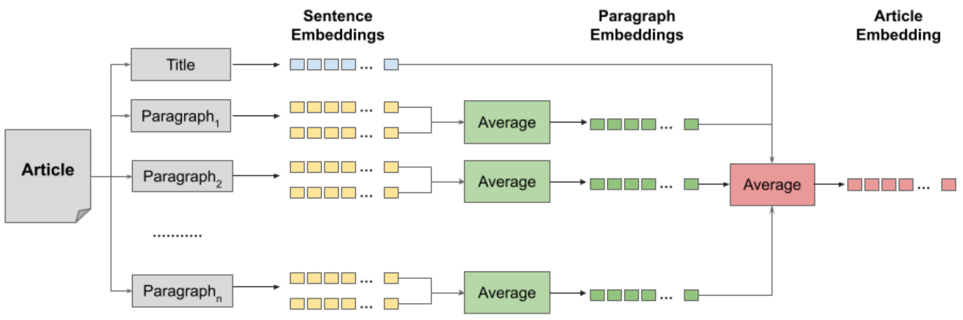


Figure 4.2: Sentence-based hierarchical dense representation of news articles using Transformer-based encoders

Using those sentence-based encoders, we get a single embedding vector for a candidate article using a hierarchical approach illustrated in Figure 4.2. We first split each paragraph in the article into sentences,[2] then obtain an encoding vector for the paragraph by averaging the embedding vectors of its sentences. Finally, we average all vectors of the paragraphs and title as well to obtain an overall representation for the article. We adopt the same approach to get the vector representation of each passage of the query article.

- **Models trained on limited-size input (maximum 512 tokens)**
  - **ERNIE-2.0** [98] is a model trained for natural language understanding tasks through continual multi-task learning. The training tasks relied on weak-supervised signals that were obtained from data such as named entities, adjacent sentences, and discourse relations. It was shown to be effective on many language understanding tasks.[3]
  - **LinkBERT** [95] is a BERT-based model trained on Wikipedia by feeding pairs of documents with hyperlink or citation information to the same model to obtain a contextual representation of text that incorporates more knowledge about the links within. While the content of news articles are somehow different, we hypothesize that LinkBERT might be able to similarly capture the semantic links between the query article and its background articles.

[1]https://github.com/YJiangcm/PromCSE

[2]We used the python *segtok* package for that (https://github.com/fnl/segtok.git).

[3]Within The General Language Understanding Evaluation (GLUE) benchmark `https://gluebenchmark.com/`.

We adopt the same hierarchical approach above to obtain a single embedding vector for each candidate article and the query article, except that we directly get the embedding vectors for paragraphs using the above models without the need to split them into sentences.

- **Models trained on long-text input**
  - **BigBird** [24] is a Transformer-based model that extends the traditional Transformer-based models to allow longer sequences of input to be processed through the concept of sparse attention. It also applies global attention on tokens such as the CLS token, as well as random attention. This model is shown to be effective in both document classification and document matching tasks [94], showing its ability to provide a good representation of long documents.
  - **LongFormer** [25] is a similar and competitive model to Bigbird that uses a combination of sliding window attention and global attention to avoid the quadratic processing time needed for the full-attention mechanism.

  Since the above models allow for very long sequences of text input (up to 4096 tokens), we obtain the vector representations of the candidate articles and also the query passages directly using those models.[4]

Computing Semantic Scores After obtaining the vector representation of the query passages and candidate articles, we compute the semantic scores in a straight forward way. Let $E(p_i^q)$ denotes the encoding of the query passage p_i^q and $E(b_j)$ denotes the encoding of the candidate background link b_j. A semantic score for a candidate news article b_j can then be computed through either the *average* or *maximum* aggregation functions as follows:

$$S(b_j, q) = \frac{\sum_{i=1}^{m} sim(E(b_j), E(p_i^q))}{m} \tag{4.1}$$

$$S(b_j, q) = \max_{i=1}^{m} sim(E(b_j), E(p_i^q)) \tag{4.2}$$

where sim is the cosine similarity function over the two input embedding vectors.

4.2.3. Stage 3: Integration of Lexical and Semantic Signals

We next integrate both the semantic and lexical matching signals to finally rerank the candidate articles for an effective news background linking. We experimented with two approaches: score aggregation and rank aggregation, surveyed from the literature [99].

Score Aggregation We experimented with the following score aggregation functions to compute the final score $R(b_j, q)$ of a candidate news article b_j:

- **SumOfScores**: $R(b_j, q) = L(b_j, q) + S(b_j, q)$

[4]Articles that exceeded 4096 tokens in length were truncated.

- **MaxScore**: $R(b_j, q) = max(L(b_j, q), S(b_j, q))$
- **MinScore**: $R(b_j, q) = min(L(b_j, q), S(b_j, q))$
- **ProductOfScores**: $R(b_j, q) = L(b_j, q) \cdot S(b_j, q)$
- **MixedMinMaxScore**:
  $R(b_j, q) = max(L(b_j, q), S(b_j, q)) - \frac{min(L(b_j,q),S(b_j,q))^2}{max(L(b_j,q),S(b_j,q))+min(L(b_j,q),S(b_j,q))}$

Before applying the aggregation function, we normalize both the lexical and semantic scores by applying sum-to-unity normalization. This normalization turns the scores into probabilities, such that the sum of scores of candidate articles in a specific ranking for a query is 1.

Rank Aggregation We also experimented with two popular rank aggregation techniques [99]. Let us assume that lexical or semantic scoring assigns a decreasing rank for each candidate news article b_j such as the most relevant article is ranked 1. Let $L_{rank}(b_j, q)$ and $S_{rank}(b_j, q)$ be the rank given to the candidate article b_j after lexical and semantic matching respectively. We applied rank aggregation using either of the following two methods:

- **Borda Rank**:
  $R(b_j, q) = (1 - \frac{L_{rank}(b_j,q)-1}{100}) + (1 - \frac{S_{rank}(b_j,q)-1}{100})$
- **Dowdall Rank**:
  $R(b_j, q) = \frac{1}{L_{rank}(b_j,q)} + \frac{1}{S_{rank}(b_j,q)_{b_j}}$

4.3. Experimental Evaluation

In this section, we show our experimental setup and our experimental results addressing each of our aforementioned research questions.

4.3.1. Experimental Setup

Our experimental setup covers the pre-processing and indexing phase of the Washington Post dataset, the baseline method, and how we implemented the proposed approach.

Preprocessing and Indexing : We used JSOUP library[5] to extract the raw text from the HTML text of the news articles' JSON objects. Afterwards, we lower-cased the text and removed stop words. Finally, the pre-processed text was indexed, along with the article's meta-data, using Lucene v8.[6]. We did not perform stemming as, per preliminary experiments with the full-article baseline, the performance was degraded compared to non-stemming.

[5]https://jsoup.org/
[6]http://lucene.apache.org/

Baseline : As mentioned before, since the news background linking problem was mainly and extensively addressed within the scope of the TREC conference, we elected to choose the baseline as the most (to date) effective method considering, on average, all queries released within the news background linking track [3]–[5]. This method uses the concatenated text of the query article's title and its body content (after stop words removal) as a search query to retrieve the background links. Throughout the years of running the background linking task in TREC, this method was implemented by different teams and using different setups (e.g., different retrieval models, different indexing systems, preprocessing options and post-retrieval filtering options). Therefore, for comparison with our proposed approach, we implement our own version of this method, and make our implementation code publicly available for future comparison purposes[7]. To obtain the background links using this method, we used the default Lucene scoring function, which is an implementation of the BM25 retrieval model, to lexically score the articles in all of our experiments.[8] Upon retrieval of the candidate articles, we excluded articles that are "Opinions", "Letters to the Editor", or "The Post's View" as they were declared by TREC to be non-relevant. While it was not mentioned in the task definition by TREC that background links should have been published before the query, we opted to filter out articles that are published after the query from the candidate links. We believe that in a real-life scenario where an author is writing an article, and wants to add background links to, then all what he will find is resources published before his written article. Note that not all teams that participated in TREC with a version of this baseline method opted to apply this filter. Accordingly, we can't directly report evaluation results from their work or from TREC.

Proposed Approach Implementation We used the baseline method explained above to retrieve an initial set of 100 candidate background links for each query article from the constructed index of news articles. For each query passage and each candidate article, we obtained the semantic representations using the pre-trained encoder models we selected, without fine-tuning. We then computed the semantic similarity scores required for the reranking process. We finally reranked the retrieved set of articles as shown earlier.

4.3.2. Semantic Representation of Articles for News Background Linking

To address **RQ2.1**, we reranked the candidate background links for each query using only the semantic matching scores achieved by the different models as we presented in Section 4.2.2. This shows the effectiveness of each of those representation models for our task. Table 4.2 presents the results for this experiment, from which we can draw multiple observations.

First, comparing the performance of the different types of models, the sentence-level models are clearly outperforming the limited-size-based and long-text-based models. More specifically, **SBERT** exhibits the best performance among all models. This can be due to our fine-grain construction of the hierarchical article representation that captured its semantics. It even achieved better performance than *EASE* and *PromCSE*, which

[7]https://github.com/Marwa-Essam81/ZeroShotRerankingNBL

[8]We experimented with other retrieval models implemented within Lucene, however, the default one achieved the best results for the baseline, therefore, we adopted it for our experiments.

Model	Model Input	Max Aggregation	Average Aggregation
SBERT	Sentence	**0.4145**	**0.4360**
EASE	Sentence	0.3376	0.3862
PromCSE	Sentence	0.3905	0.4032
ERNIE-2.0	Limited-size	0.2878	0.2921
LinkBERT	Limited-size	0.3057	0.3236
BigBird	Long-text	0.2059	0.1856
LongFormer	Long-text	0.1473	0.1462
Baseline			**0.4856**

Table 4.2: nDCG@5 performance of semantic matching using different semantic representation models.

were proposed more recent, and showed high effectiveness on several standard datasets for Semantic Textual Similarity (STS) Tasks [93], [96].

Second, surprisingly, the performance of the long-text-based models was the worst, although they were shown to be effective on a number of tasks that included long text matching [94].

We further notice that average aggregation is generally better than the maximum aggregation in this task. This is somewhat expected for our task, as the average aggregation combines more signals of context background across the query article than the maximum aggregation.

Finally, we notice that our best model (SBERT) is outperformed by the baseline. This is also expected, as it only captures the semantic signal with no emphasis on the lexical similarity between the query and the background articles.

4.3.3. Integration of Relevance Signals

Having noticed that the semantic matching signal is not sufficient for an effective background linking, and to address **RQ2.2**, we experiment with the different score and rank aggregating methods that integrate both the lexical and semantic signals. For that purpose, we focus only on the SBERT semantic model, as it exhibited the best performance among the encoder models. Table 4.3 presents the results of this experiment.

The results show that score aggregation methods are generally better than rank aggregation methods. More importantly, three of them, namely **SumOfScores**, **ProductOfScores**, and **MixedMinMaxScore** exhibit *statistically significant* improvement over the baseline, achieving a new SOTA for this problem. The difference between the three aggregation methods is not statistically significant though, favoring the SumOfScores method for simplicity and efficiency purposes.

4.3.4. Potential of Query-based Integration of Relevance Signals

While the above experiments demonstrated that the integration of the lexical and semantic relevance signals exhibited significant improvement over the SOTA baseline, it was applied to *all* queries in the same way. This is simple, but probably not optimal;

Aggregation Method	nDCG@5
SumOfScores	**0.5016***
MaxScore	0.4854
MinScore	0.4736
ProductOfScores	**0.5003***
MixedMinMaxScore	**0.5021***
Borda Rank	0.4831
Dowdall Rank	0.4775
Baseline	**0.4856**

Table 4.3: nDCG@5 performance of the different lexical-semantic aggregation methods. Results marked with * show a statistical significant difference over the baseline using pairwise t-test with 95% significance level.

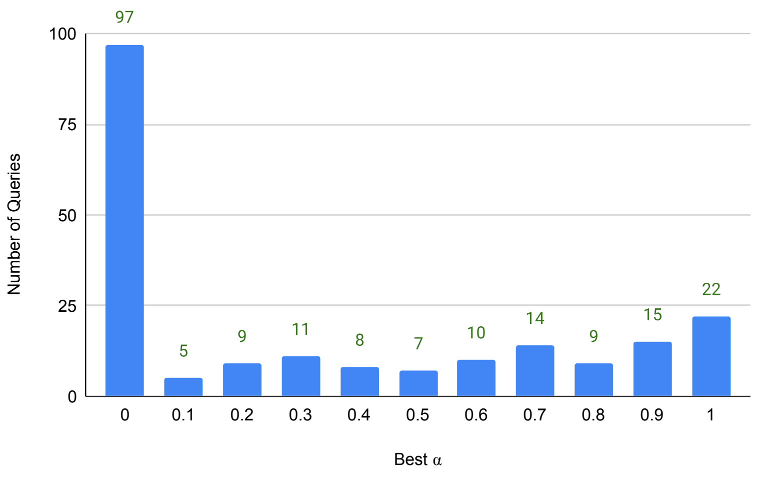


Figure 4.3: Distribution of queries over the values α that achieve the best nDCG@5 scores per query

we hypothesize that some queries might benefit *only* from the lexical signal, some might benefit *only* from the semantic signal, and others might benefit more from *mixing* both with different balances. To test this hypothesis, and addressing **RQ2.3**, we conducted an experiment where we computed the final score of a candidate article by a linear interpolation (weighted average) of the lexical and semantic scores, i.e., $R(b_j, q) = (1 - \alpha) \cdot L(b_j, q) + \alpha \cdot S(b_j, q)$, where $\alpha \in [0..1]$ is the weighting factor. We changed the value of α, per query, from 0 (no semantic signal) to 1 (no lexical signal) with an increment of 0.1. Figure 4.3 illustrates how many queries achieved its highest nDCG@5 score at the different values of α. We can draw several interesting observations from the figure. First, query articles widely vary in their need for lexical and semantic relevance signals, which indicates that our simple integration is clearly suboptimal. Second, 97 queries (about 47% of the query set) achieved their best nDCG@5 score using a pure lexical relevance signal. We can also notice from the figure that 70 queries (about 33.5% of the query set) needed more semantic than lexical signals, i.e., had their best nDCG@5 scores with $\alpha \geq 0.6$.

Given our earlier insights on the notion of background relevance, we expected that query articles that needed a pure lexical signal for the matching process may have reported specific *incidents* or *events*, for which finding background articles that share most of the used vocabulary increased the chance of acquiring more knowledge on the exact reported event/incident. Investigating some of these queries, we indeed found, for example, the query article titled "*Metro, Local Responders Would Jointly Test Radios Under New Plan*". This article is giving feedback on an incident in the "L'Enfant Plaza Station" in the United States, where a train had become disabled, and smoke filled the tunnel causing deaths and injuries. The article illustrates how there will be a test of the equipment that was not functioning properly during the incident. Clearly, the reader of the query article would want to know more details about this specific incident that was not detailed in the query article. One of the highly-relevant background articles for this query is the article titled "*Montgomery Firefighters Find Radio-signal Blind Spots Near 2 Metro Stations*". That article mentions the exact same incident, with more complementary details on why the radio signals were not functioning. It accordingly has high frequencies for words mainly used in the query article, such as "Metro," "Plaza," "firefighters," "radio," "communication," and "smoke".

On the other side, an example of the queries that needed a pure semantic signal during the matching process is the query article titled "*The Solution To Climate Change That Has Nothing To Do With Cars Or Coal*". That article specifically talks about the tropical forests near the Amazon river in Brazil and how cutting trees there had a negative effect on the climate change recently. For this article, articles that mainly talk about the dangers to the forests, the recent climate change, or either that are not necessarily around the Amazon river would increase the reader's knowledge on the subject of the main article. For instance, we found that the article titled "*The Forests of the World are in Serious Trouble, Scientists Report*" is highly relevant although it uses different vocabulary than the one used in the query, with big gaps in the frequencies of even common words. That article discusses mainly the major threats to forests of the world one by one. It hence uses words like "temperate," "boreal," and "planted," which are not at all mentioned in the query. Words like "Amazon," "Brazil," and "Climate" which mainly shape the discussion in the query article are infrequent in that candidate. Accordingly, for those type of query articles, the exact same vocabulary used in the article is not actually essential for matching the background links, rather than the actual context of the text. Presumably, using semantic relevance signals while ranking candidates for those articles ensures that the background articles are favored solely based on the context, regardless of the vocabulary being used.

Those observations introduced here though need to be supported by another extensive analysis of many query articles and the effect of being an event-driven or not article on the retrieval process. However, our simple analysis here, again, sheds the light on the importance of distinguishing different types of query articles, which is a potential future direction for working on our problem.

Finally, to see how far we can go if the optimal value of α is predicted, we computed the overall nDCG@5 score by choosing the best α achieved by each query. In that case, the overall nDCG@5 reaches **0.5487**, which is quite a significant increase over the SOTA baseline method (nDCG@5 value of 0.4856)[9]. Even if we restrict the values of

[9]Significance was measured using a Paired t-test with 95% confidence

α to be only 0 or 1 per query (i.e, to use either a lexical or semantic-based reranking of candidate background articles), we can reach an nDCG@5 score of **0.5262**, which would still be significantly higher than the baseline performance. This shows a potential significant improvement with models that can adaptively predict a better integration of the lexical and semantic signals per query.

4.4. Conclusion

In this chapter, we introduced our first proposed approach for an effective news background linking. This approach adopts a sentence transformer model, in a zero-shot setting, to semantically match the query articles to its background links. The proposed approach exhibited a performance that significantly outperforms the full-article retrieval approach, marking a new SOTA for the news background linking problem. While effective, this proposed approach was based, mainly on the observation that lexical similarity is not the only relevance signal in news background linking. It did not, however, benefit from the other insights on news background linking that we showed in Chapter 3. Next, we will introduce our second proposed approach for an effective news background linking that, based on these aforementioned insights, analyzes the query articles and the candidate background links using Large Language Models to reveal the relation between its discussed subtopics and rerank the candidate links accordingly.

CHAPTER 5: LARGE LANGUAGE MODELS FOR RERANKING BACKGROUND LINKS

In the previous chapter, we showed how we adopted a transformer-based model in a zero-shot setting to increase the effectiveness of the full article search method for news background linking, marking a new SOTA for this problem. The proposed approach integrates both the semantic and the lexical matching signals between the query article and its background links. Capturing the semantic relevance signal ensures that relevant articles that do not exhibit high lexical similarity with the query article are properly ranked during the background linking reranking process. In this chapter, we introduce another approach that also integrates both the lexical and semantic relevance signals for reranking the candidate background links. Though, rather than just encoding the content of the query article and its candidate background links and computing its similarity in the latent space, as with the previous proposed approach, the approach introduced in this chapter depends on analyzing the contents of the query article and the candidate background to find its actual fine-grained semantic relation and produce the final background links accordingly.

Very recently, research has been conducted on harnessing the reasoning capabilities of Large Language Models (LLMs) in many information retrieval tasks, and the results are very promising [26], [100], [101]. Using the knowledge within these models, one can simply analyze a text document, extract its context and even compare documents for the context they share. Accordingly, we aimed to investigate how to use an LLM for a simple and effective reranking of the links produced through the full article lexical-based approach. The literature in reranking text documents though using LLMs mostly experimented on datasets for passage retrieval tasks such as the MS MARCO dataset[102]. Some researchers used black-box LLMs directly or indirectly for reranking the passages [103]–[105]. Others introduced open source models to overcome the non-deterministic results that may come from black-box models [106], [107]. Compared to passages that are limited in size though, news articles are relatively long to fit in the context length of many models. Whether or not the LLM will be able to effectively analyze news article for the news background linking problem, up to our knowledge, is still an open issue.

In this chapter, we introduce our approach that adopts a large language model to analyze the query article and a set of candidate links, and effectively rerank the candidate links based on this analysis. Through our proposed approach, we address the research question:

RQ3: *Can we harness the potential of the recently developed Large Language Models (LLMs) to effectively rerank the links retrieved through the full-article search approach?*

To address this question, we investigated the answers to the following sub questions:

- **RQ3.1**: Can we leverage LLMs to directly rerank candidate background links for a given query article?
- **RQ3.2**: Can we effectively extract useful background information from within each candidate link using LLMs, and use this information for better reranking of the candidate links?

.

Our proposed approach in this chapter is based on the insight, drawn during our qualitative analysis of news background relevance, that a highly relevant background article does not need to discuss all what is mentioned in the query article, and it rather shows related information to specific subtopics mentioned in the query article, in addition to introducing its own new subtopics. Following, in Section 5.1, we show how to benefit from this insight through a detailed description of our proposed framework for adopting LLMs for news background linking. Then, in Section 5.2, we detail our experimental evaluation addressing each of the research questions aforementioned in this chapter.

5.1. Proposed Framework

In this section, we show how we adopt LLMs to rerank candidate background articles for a query article. Similar to our transformer-based reranking approach, we obtain the candidate links through the full-article ad-hoc search method explained earlier. As shown before, post this initial retrieval method, we will have a list B of k candidate background articles, ranked by their lexical similarity with the query article. Each candidate b_i is assigned a rank $L(b_i, q)$ based on its similarity with the query article q. Next, we explain, in detail, our proposed framework for both the direct and the analysis-based reranking of those candidate background links using an LLM.

5.1.1. Direct Reranking

A simple and straightforward approach to address the news background linking problem using LLMs, is to directly ask a model to rank a set of candidate background articles for a query article q. Following this approach, we simply prompt the LLM to rerank the retrieved candidate background links based on its background relevance to the query, getting a new LLM rank $M(b_i, q)$ for each candidate b_i, as follows:

```
I will give you below a number of news articles from [Article 1] to
[Article k]. I will also give you another news article: [Query].
Rank the Articles by their context and background relevance to the
Query. Provide your answer in JSON with a field ranked_list. DO NOT
justify your ranking.
[Article 1] <Article 1>
[Article 2] <Article 2>
...
[Query] <Query>
```

Note from the prompt above that we strictly ask the LLM to provide its answer in JSON format to have a unified response for all query articles, leading to easier post-processing of them. We further ask the model not to justify the response, otherwise we get unnecessary output.

The ranking produced by the LLM for the candidate articles can be presented to the reader as is. Alternatively, it can be aggregated with the original ranking of the candidates. From our preliminary experiments on this task, we found that sometimes LLMs rank higher some low-relevant candidates that provide general contextual relevance to the query, but do not actually mention the very specific incident/topic reported in the query. For some queries, this lower the ranking effectiveness. Therefore, we propose again to aggregate the rank produced by the LLM with the original lexical similarity-based ranks of the candidate links to ensure, on average, that the recommended background articles do not lexically deviate much from the topic discussed within the query article. Note that we can not aggregate scores here as most of the state of the art LLMs are not open source (i.e. they do not provide the ranking scores).

To obtain the final rank for a candidate article b_i, we aggregate its full-query search ranks $L(b_i, q)$ and its LLM rank $M(b_i, q)$. There are several strategies to aggregate the ranks, as shown in Chapter 4; in this proposed approach, we opted to simply add them: $R(b_i, q) = L(b_i, q) + M(b_i, q)$. We finally sort the candidate articles given their new aggregated ranks. If two candidate articles got the same aggregated rank, we favor the one with the higher lexical similarity with the query article.

5.1.2. Analysis-based Reranking Approach

Motivation

The context length of many LLMs may not fit the query article in addition to all candidate articles if a direct ranking of the articles is to be adopted as discussed in the previous section. Besides, asking an LLM simply to rank candidate articles based on its background "relevance" to the query might not be as simple as ranking passages that contain an answer to a question. Even the word "relevance" is a tricky word here as, most probably, all candidate articles, retrieved through lexical similarity with the query, have somehow contextual relation (in other words, relevance) with the query article. This might hinder the model's ability to determine which candidate contains *significant* background information, and accordingly which one should be ranked higher.

The insights we drew on the notion of background relevance, introduced in Chapter 3 show that useful background articles do not need to have the same main topic as the query article. Yet, these articles often expand the information given on one or more of the query article's subtopics. Additionally, highly relevant articles usually introduce new subtopics related to the subject matter of the query article (whether this subject matter is a reported event or a general topic). Hypothetically, LLMs can be used to analyze the query article, reveal its discussed subtopics, then provide a better ranking for the candidates based on the salient information it provides on these subtopics. To test this hypothesis, we qualitatively assessed the performance of LLMs in extracting the subtopics discussed within a set of news articles against a human analysis of the same set. Details of this assessment are given below.

In our qualitative analysis of the LLMs ability to analyze the news articles, we compared the LLMs responses against our manual annotation of a sample of query news articles that was shown in Chapter 3. Recall that in this work, the main topic of each query article was identified along with a number of discussed subtopics. We hence used the prompt below to ask the LLM to extract the sample query articles topics:

```
I will give you a news article identified by the tag [Article]. What
is the main subject matter of the article? List also the title of 3
subtopics that the article discusses. Provide your answer in JSON
with the following fields: Main_subject, Subtopics_List.
DO NOT explain or say any more words.
[Article] <Article>
```

For each LLM we experimented with, and for each main topic/subtopic annotated per query, we manually assign a score from 0 to 3 that indicates to what extent the extracted topic matches the human annotation. Precisely, a score 3 is assigned if the extracted topic is almost exactly as the one manually annotated, score 2 is assigned if it is very closely related to the manually annotated topic, score 1 is assigned if it is somehow related to the annotated topic and score 0 is assigned if the extracted topic is not at all related to the annotated topic. Finally, for each LLM and given the assigned scores, we compute the Mean Squared Error (MSE) between the LLM's annotation and the human annotation[1].

Selected LLMs Upon investigating the available LLM models, we opted to experiment with models at the top of the Chatbot Arena [2]. We excluded LLMs that do not provide APIs for processing prompt requests, as it needs extensive in-house computing resources that were not available for this work. Specifically, we experimented with gpt-4-1106-preview [108] (denoted as **GPT-4 Turbo** hereafter), and gemini-1.5-pro (denoted as **Gemini Pro 1.5** hereafter).[3] We further opted to experiment with the models: gpt-3.5-turbo-1106 [108] (denoted as **GPT-3.5 Turbo** hereafter), and gemini-1.0-pro (denoted hereafter as **Gemini Pro 1**) to compare its performance with its successor and more expensive models.

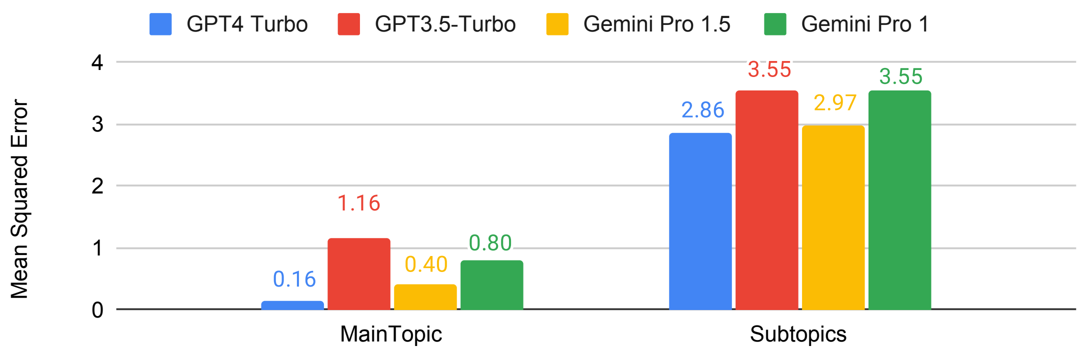


Figure 5.1: Assessment of LLMs ability to analyze news articles.

Assessment Results Figure 5.1 shows the results of our experiment. As illustrated, the GPT-4 Turbo model showed very high performance when detecting the main topic of the query article (MSE of only 0.16). It is furtherly the best model to identify the different subtopics discussed as compared to the human annotation. The error in detecting the subtopics, however, is higher than the error in detecting the main topic. This shows that identifying the salient subtopics within an article is still a challenging process for the LLMs. Gemini Pro 1.5 also showed a comparable performance to GPT-4 Turbo when analyzing the articles. We can further notice that the performance of both models

[1]We reviewed the manual annotation of articles to ensure that it has between 3 and 4 subtopics per query. For computing the MSE for the subtopics, we averaged the LLM's performance over all subtopics of all annotated queries.

[2]At the time of running our experiments, the models we experimented with were on top of the arena https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard

[3]We used the latest available versions of the models from OpenAI and Google at the time of running our experiments

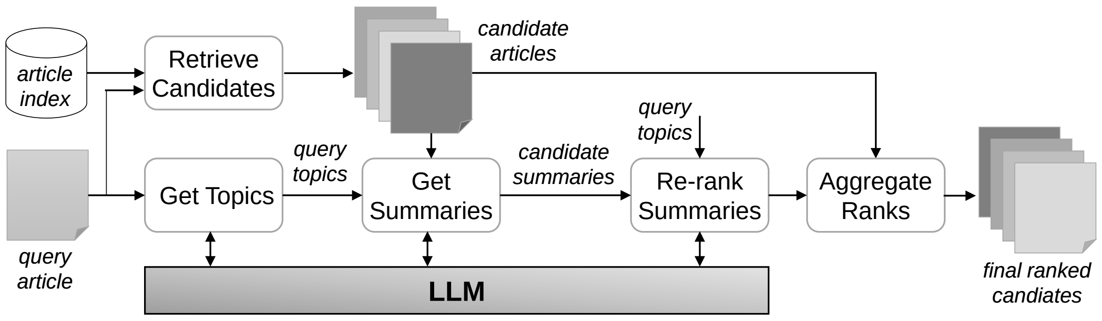


Figure 5.2: Proposed framework of analysis-based reranking.

quiet much exceeds its predecessor models. This is somehow expected as those new models are further trained to allow for high performance on different reasoning tasks. The performance of the predecessor models, however, did not deviate much from the human annotations, and can still be acceptable. This encouraged us to experiment with these LLMs in our proposed approach, as will be explained next.

Proposed Reranking Approach

Having seen the potential in LLMs when prompted to analyze news articles, we propose our *analysis-based* reranking approach that ranks a set of candidate background links for a query article through a number of adequately-crafted prompts to the LLM (see Figure 5.2).

As in the direct reranking approach, we first retrieve a set of candidate links for the given query article through the full-article search over an indexed collection of news articles. The query article is then analyzed to extract the topics discussed within. Each candidate article is further analyzed to extract a passage that is related to the query article. The candidate passages (links) are then re-ranked. All of the steps after the initial retrieval are performed using the LLM. Finally, the initial candidate ranks along with the LLM-based ranks are aggregated to produce the final ranked list of background links. Following, we describe the pipeline of our framework in more detail. We skip the description of the candidate retrieval step and the query analysis step in this framework as both steps are explained before.

Analyzing Candidate Articles Relevant candidate background articles may contain information that is not of a benefit to the reader of the query article (i.e., considered noise in relation to our task). That means that getting rid of such additional information might help in our task. Accordingly, we propose to convert the news articles reranking problem into a passage reranking problem, by extracting a passage from each candidate background link and rerank the extracted passages. Shortening the candidate links might make them fit in the context length of many LLMs. That can also simplify the reasoning process for the LLM during the reranking of candidate links. The prompt below shows how we obtain the passages by requesting the LLM to provide a summary for each candidate article, showing only related information to the subjects extracted from the query article in the former step of our framework:

```
A reader is seeking information related to the following related
subjects: <Main_subject>, <Subtopic1>, <Subtopic2>, <Subtopic3>
Provide an extractive summary from the article, identified by the
tag [Article], showing what may benefit the reader. Do not exceed
150 words in your summary. Provide your answer in JSON with the field
extracted_summary.
[Article] <Article>
```

We pass to the LLM the main subject matter discussed in the query article as the first subject, and the subtopics discussed within as another three subjects. We ask the model to limit the provided summary to 150 words, to avoid truncation of summaries later during the reranking process. Similar to the earlier step, we ask the model to provide its response in JSON format.

Reranking Candidate Background Articles Having the summaries extracted from the candidates, we then ask the LLM to re-rank those summaries based on their benefit to the reader of the query article, given its main topics we extracted earlier. We provide in the prompt to the LLM the summary of each candidate article, as a passage, in the order of its initial retrieval, and ask the model to re-rank them without justification.

```
A reader is seeking information related to the following related
subjects: <Main_subject>, <Subtopic1>, <Subtopic2>, <Subtopic3>
Rank the passages below from [Passage 1] to [Passage k] given how much
they provide useful information to the reader. Provide your answer in
JSON with the field ranked_list. DO NOT justify your ranking.
[Passage 1] <Passage 1>
[Passage 2] <Passage 2>
....
```

Aggregating Ranks Eventually we apply the rank aggregation, described in Section 5.1.1, in an attempt to ensure that the final recommended articles preserve some kind of lexical similarity with the query article, assuming it is reporting very specific incidents or events.

5.2. Experimental Evaluation

In this section, we show our experimental setup and discuss the results addressing each of our aforementioned research questions:

5.2.1. Initial Retrieval of Candidate Background Links

As with our transformer-based reranking approach, we retrieved the candidate background links using the full-article search approach. Again, we filtered out articles that were published before the query, as well as *Opinion*, *Letters to the Editor* and *The Post's View* articles. For budget constraints with the LLMs and to make the ranking process much simpler, this time we retrieved only 15 candidate articles per query article for reranking. The optimal reranking for these 15 candidates though still shows relatively high potential for reranking. As explained in the motivation for the analysis-based reranking approach, we used the 2018 set from the Washington Post dataset to curate the prompts for the LLM and to assess its news articles analysis potential; accordingly, we report our results here on the other 3 sets.

5.2.2. Candidates Articles Reranking

For reranking the retrieved candidate articles, we experimented with both **GPT-4 Turbo** and **Gemini Pro 1.5** in the direct ranking approach as they allow for up to or more than 128k input tokens. We further experimented with **GPT-3.5 Turbo** and **Gemini Pro 1**, for the analysis-based approach. Both models limit the input context to 32k tokens. Since a single news article do not often exceed this input restriction, truncation was not actually needed when working with these models.[4] To allow for as much deterministic output as possible, we set the temperature hyperparameter in all models to 0.

5.2.3. Results and Discussion

Table 5.1: nDCG@5 performance of the direct ranking approach against Baseline. Results written in bold show an increase in performance over the baseline.

	Method	2019	2020	2021	Average
SOTA	Baseline	0.593	0.544	0.345	0.498
GPT-4 Turbo	LLM Only Ranking	0.577	0.516	0.338	0.481
	+ Lexical Ranking	**0.601**	**0.551**	**0.351**	**0.504**
Gemini Pro 1.5	LLM-only Ranking	0.548	0.505	0.334	0.465
	+ Lexical Ranking	0.592	**0.553**	**0.348**	**0.500**

Direct Reranking (RQ3.1) Table 5.1 presents the performance results of the proposed direct ranking approach compared with the SOTA baseline. We used one-tail paired t-test to measure statistical significant improvements over the baseline with 0.05 significance level. The table shows the performance of the GPT-4 Turbo model and the Gemini Pro 1.5 model with and without the ranks aggregation step explained in Section 5.1.1. First, it can be noticed that the final ranking performance using either model increases post aggregating the LLM ranks of the candidates with their original lexical ranks. This somehow highlights the importance of considering the lexical similarity between the query article and the candidate links as one of the relevance signals when addressing news background linking. It can be further noticed that the increase in performance over the baseline, post the aggregation step, is slight and not significant.

It is important to highlight here that the large context length that the GPT-4 Turbo and the Gemini Pro 1.5 models take in addition to its big knowledge base may make researchers issue direct prompts to it when addressing new unseen tasks such as news background linking. However, the above results show that crafting a single prompt for these LLMs to address unseen new problems in a zero-shot setting may not be a

[4]except for GPT-3.5 Turbo and Gemini Pro with only one surprisingly very long candidate article.

straightforward task. It further confirms our hypothesis on the challenges that face the LLMs during the reasoning process when directly asked to rank candidate articles for background relevance.

Table 5.2: nDCG@5 performance of the analysis based approach against baseline. Results written in bold show an increased performance over the baseline, with starred results showing statistically significant improvements.

	Method	2019	2020	2021	Average
SOTA	Baseline	0.593	0.544	0.345	0.498
GPT-4 Turbo	LLM Only Ranking	**0.601**	0.543	0.333	0.496
	+ Lexical Ranking	**0.616***	**0.575***	**0.353**	**0.518***
Gemini Pro 1.5	LLM Only Ranking	0.557	0.504	0.317	0.462
	+ Lexical Ranking	0.578	0.544	**0.350**	0.493
GPT-3.5 Turbo	LLM Only Ranking	0.581	0.498	**0.363**	0.484
	+ Lexical Ranking	0.579	0.522	**0.359***	0.490
Gemini Pro 1	LLM Only Ranking	0.511	0.490	0.319	0.442
	+ Lexical Ranking	0.568	0.534	**0.349**	0.489

Analysis-based Reranking (RQ3.2) Table 5.2 shows the results of our proposed analysis-based ranking approach. The significant results are annotated by * in the table. First, it can be noticed again from this table, that regardless of the model being used, the lexical ranking aggregation step always increases the performance of the ranking results. Additionally, the performance of this proposed approach using GPT-4 Turbo LLM, post the aggregation step, exhibited statistically-significant improvement over the baseline, establishing a new SOTA performance. The performance of this approach though with the other three models was inferior to the SOTA baseline.

We can further notice from the table that each of the more advanced models (GPT-4 Turbo and Gemini Pro 1.5) showed an increase in performance, on average, over its predecessor models using the proposed approach. While the increase in performance is not high, the results here shed the light on the potential of LLMs in addressing challenging problems with further enhancements.

Failure Analysis It is worth noting that LLMs with APIs often suffer from failure patterns while processing the given prompts [107]. We randomly selected some of the queries and candidates analyzed by the models and checked its output. Interestingly, we found that in some cases when the models Gemini Pro 1.5 or Gemini Pro 1 found no relation between the candidate article and the given query subtopics, it attached sentences like *"the article is not relevant to the reader's interests..."* to the candidate's summary. This is in comparison to the output of the models GPT-4 Turbo and the model GPT-3.5 Turbo that showed actual summaries without attaching the model's opinion on the relevance of the information given to the query topics. Since, as per our approach, we feed the generated summaries of candidates as is to the model, we hypothesize that the former models got confused during the ranking process with such negative sentences. This might also justify why, in some cases, we found skipped passages in the rankings produced by these models.

The APIs for Gemini Pro 1 and Gemini Pro 1.5 further blocked some prompts for safety restrictions (e.g., ones with "dangerous" content), resulting in blocking our requests for two queries and 16 candidate articles while processing the output from Gemini Pro 1 and 2 candidate articles while processing the output from Gemini Pro 1.5. [5] Note that for producing the reported results, we added the missing passages in their original lexical ranking at the tail of the models' produced ranked lists. For the queries that the API blocked, we left the baseline ranking as is. As for the blocked candidate articles, we set the summaries to the concatenation of the article's title and its leading paragraph. Note that while the number of blocked passages is not big, this might hinder using these models during a production process. Finally, since GPT-4 Turbo was the least to produce noise, formatting mistakes in the output JSON objects, or missing passages, we conclude that it is the most reliable model, among the ones we studied, to use within our proposed approach.

Comparison With Transformer-based Approach When comparing the results that we obtained using our LLM Analyis-based reranking approach with the one that we got through our previously proposed transformer-based approach presented in Chapter 4, for the 2019, 2020 and 2021 sets of query articles, we found that the results, surprisingly, while being different per query, are the same on average over all queries, (0.518 vs 0.514) with no statistically significant difference.[6] This suggests that either of the approaches can be chosen for an effective background linking process. The advantage of using the transformer-based approach is that it did not require any learning process, compared to the LLM-approach that required us to use the 2018 set of query articles for curating the model prompts. The advantage of using an LLM though with our proposed analysis method lies in its simplicity. No need to encode any news article nor to compute matching scores. Additionally, it is important to note that the results we obtained through the LLM-based approach was post reranking only 15 candidate background articles per query article, compared to the transformer-based approach that reranked 100 candidate background articles per query. Due to budget constraints and context-length limitations of the LLMs as mentioned before, we could not experiment with reranking the 100 candidate set using the LLMs. Hence, the results obtained through the LLM may reverse positions if more candidates are considered during the reranking process. We leave this as a future work. There are, however, budget constraints when working with black-box LLMs like GPT4-Turbo, and the risk of API response failures. Whether or not the LLM analysis-based reranking approach will be as effective with open-source LLMs is still an open issue that needs to be investigated.

5.3. Conclusion

In this chapter we showed how we adopted LLMs to propose our second state of the art effective approach for news background linking. Both of our transformer-based

[5]Even though we selected a "*block-none*" setting at the time of running our experiments.

[6]The candidates for the transformer-based approach though were 100 and were retrieved using V3 of the Washington Post dataset that contained slightly less number of news articles

and LLM-based news background linking approaches work on reranking a candidate set of background links retrieved for each query article using the full-article search approach. This retrieval approach is very inefficient though given the very long search query that needs to be issued to the indexing system to retrieve the candidate links. For some query articles that do not report hot news and rather elaborate on a specific topic, the constructed search query becomes full of unnecessary terms that do not actually represent the actual subject matter of the query nor the important subtopics (noise).

In the next chapter, we propose our work on reducing the size of the search query into a much shorter search query that can be used to efficiently and effectively retrieve the background links. These links can be represented to the reader as is, or can be reranked further for better news linking effectiveness.

CHAPTER 6: EFFICIENT NEWS BACKGROUND LINKING

In the previous chapters of this dissertation, we showed how we addressed the effectiveness aspect of the news background linking problem. We proposed two approaches for reranking a candidate set of background links obtained through the full-article retrieval method, and we showed that both of our proposed approaches exhibited a significant improvement over this baseline method, marking new SOTA for the news background linking problem. In this chapter, we present our work for addressing the efficiency of the news background linking process. Precisely, we focus here on how to *efficiently* retrieve candidate background links for a query article, while maintaining the same (or comparable) effectiveness reached by using the full input article as a search query. Enhancing the efficiency of the lexical retrieval phase will enhance the overall efficiency of any news background linking approach that adopts candidate links reranking, including our two news linking approaches proposed in Chapter 4 and Chapter 5. To our knowledge, our work is the *first* that addresses the efficiency issue in the domain of news background linking problem; all other related studies focused on the effective retrieval of the background links, regardless of the time taken to obtain those links.

For an efficient background links retrieval, we hypothesize that selecting a much fewer number of terms from the query article, instead of its full content, might be enough to create an effective search query for our task. In other words, we aim to *reduce* the long query article to a much shorter search query while maintaining its effectiveness. While *query reduction* was studied before in literature [109], [110], most of that work assumed that the long queries are a description of the user's information need in Web search, which is maximum of 30 terms in length. Therefore, previous work considered techniques for assessing short sub-queries (2 to 6 terms), which in many cases typically involve post-retrieval features. This is, indeed, infeasible to apply for the lengthy news articles that have hundreds of terms, as the number of sub-queries to be assessed will pose even more time for retrieval compared to the full article, which contradicts our aim from the reduction.

In this chapter, we precisely address the research question:

RQ4: *How we can increase the efficiency of the ad-hoc based retrieval of news background links?*

To address this question, we addressed the following sub questions:

- **RQ4.1**: Can we effectively retrieve the background links by just using the *lead paragraphs* of the query article to construct the search query?
- **RQ4.2**: How effective and efficient are *typical keyword extractions* techniques for this task?

We designed a number of experiments to address the above research questions. We first explored the idea of using only the lead paragraphs of the news article as the reduced search query. We next experimented with the state of the art unsupervised keyword extraction techniques for constructing a weighted search query out of the input news article. Our results show that, with adopting a recently introduced statistical keyword extraction technique, we can reach an effectiveness that is not significantly different from using the full query article as a search query, but with much fewer search terms. Next, we explain our search query reduction methodology in detail in Section 6.1. Then, we show our experimental evaluation that addresses each of our aforementioned research questions in Section 6.2.

6.1. Proposed Methodology

As illustrated in Chapter 2, the most common approach to date to retrieve a set of candidate background links for a query news article follows an ad-hoc retrieval approach, in which, the query article q is analyzed to produce a *search query* S_q, which is issued against the collection index to retrieve the required background links. The simplest and most effective reported method to date to retrieve a set of background links either to present it as is to the reader or to consider it for reranking as we did in our earlier proposed linking approaches, constructs S_q as a concatenation of the article's title and its full content. This method is clearly *inefficient*, though, as we outlined earlier, since the retrieval query includes all the terms of the article, even if they are noisy or uninformative. Therefore, in this work, we assume that a search query that captures only the informative terms can be (at least) as effective as the full article query, yet much more efficient. Moreover, since terms in the article can have different impact in representing the different background aspects of the query article, we assume that constructing a *weighted* search query, where each unique search term has a real-valued weight, that can be different from other terms, will be helpful in improving the retrieval effectiveness. In fact, adding weighted terms to queries has been shown previously to increase the retrieval effectiveness, specially with the increase in the number of search terms [111]. In this section, we first describe how we formulate a weighted search query and show how we score the background articles given that query. We then briefly describe the different keyword extraction methods that we chose to adopt in extracting and weighting the search query terms from the query article. Finally, we present a complexity analysis for the adopted methods to contrast their efficiency in extracting the search terms.

6.1.1. Query Formulation and Document Scoring

Given a query article q, we formulate a search query S_q as follows. Let n_q be the set of unique terms in q. We apply a keyword extraction algorithm to assign a weight w_{t,S_q} for each term $t \in n_q$. We then choose $k < |n_q|$ terms with the highest weights to construct the search query S_q as follows:

$$S_q = \{(t_1, w_{t_1,q}), (t_2, w_{t_2,S_q}), ..., (t_k, w_{t_k,S_q})\} \tag{6.1}$$

Given S_q, a news background link $b \in C$ can be scored as follows:

$$score(b, S_q) = f(\{(t, w_{t,S_q}, w_{t,b} \mid t \in S_q \cap b)\}) \tag{6.2}$$

where $w_{t,b}$ is the weight of term t in b according to the retrieval model f (e.g., the term frequency of t in b). Finally, the news articles with the highest scores can be retrieved as background links for the article q.

6.1.2. Unsupervised Keyword Extraction Techniques

The most significant choice to make when applying our proposed methodology is the keyword extraction technique. We reviewed recent studies that addressed the problem of keyword extraction, focusing on those that compared the performance of state of the art techniques on the gold-standard keyword extraction datasets [117]–[120]. We also

Method	Type
TF	Statistical
TF-IDF	Statistical
k-Core [41]	Graph-based
k-Truss [42]	Graph-based
PositionRank [112]	Graph-based
TopicRank [113]	Graph-based
MPR [114]	Graph-based
sCake [115]	Graph-based
Yake [116]	Statistical

Table 6.1: The keyword extraction methods we studied in this work.

checked the methods that were reported by the recent techniques as effective baselines. Accordingly, we made our selection of the techniques based on the following criteria:

- **Unsupervised Methods:** Since the problem we address is very recent, and there is no labeled data for supervised learning, (i.e there is no golden set of keywords extracted from the query articles that can be used to retrieve the best set of background links), we focus only on selecting keyword extraction techniques that are mainly unsupervised.
- **Effective Number of Keywords:** Our goal from extracting the keywords is to use them to form search queries to retrieve candidate background documents from a big news articles collection. Hence, we prefer the keyword extraction technique that can provide a large number of good representative keywords. In our preliminary experiments, using few search keywords in a retrieval query, even of good quality or representation of the query topic, considerably lowered the retrieval effectiveness. Accordingly, techniques that failed to provide large number of good keywords (30 in our preliminary experiments) were excluded, such as **Teket** [121]).
- **Effectiveness on News Articles**: When reporting its effectiveness, many keyword extraction studies conduct the experiments on scientific paper datasets or even books; however, news articles have special features. They are shorter, less cohesive, and they may discuss multiple subtopics. Hence, we selected the recent techniques that worked best when tested on *English news articles* datasets.

Given the above criteria, we experimented with the techniques listed in Table 6.1 along with their types. We note that most of the methods are graph-based (i.e., they construct a graph of terms from the input document, then analyze it to extract the required keywords). We further experimented with two simple and standard keyword extraction methods, which are Term Frequency (*TF*) and Term Frequency-Inverse Document Frequency (*TF-IDF*), as additional baselines to compare against.

A brief description of the evaluated methods and how they weigh the terms is given below. Since we need boosting weights for individual terms to construct search queries, we further show how, for some methods, we computed those boosts given the weights assigned for the extracted keyphrases.

k-*Core*: *k*-Core [41] is a graph-based keyword extraction method that depends on the construction of a graph-of-words for the document being processed, and analyzing this graph to extract the required keywords. Nodes in the graph-of-words are simply the unique words in the text after pre-processing (e.g., stop words removal), and edges are added by sliding a window over the text, creating an edge between words that co-occur within this sliding window. In general, the underlying assumption of creating graphs from text and analyzing it is that terms co-occurring within a relatively small window of text have some kind of *semantic relatedness* regardless of their roles in a sentence, and that this relationship influences the importance of each single term within the text document, leading to better document analysis. To analyze the constructed graph of words, *k*-Core decomposes the graph into a hierarchy of nested *subgraphs* using graph degeneracy methods, and goes down this hierarchy to identify nodes that are at the *core* of the graph. This is based on the assumption that nodes in the core are reachable through the graph by many other nodes, making them influential. *k*-Core decomposition relies on *peeling away* weakly connected nodes to gradually get the core of the graph. The *k*-Core of a graph is the maximal subgraph such that every node has a degree at least *k*, where the degree of a node is the sum of the weights on its connecting edges. Starting with the initial graph and with *k*=1, the subgraphs are iteratively created with increasing *k* and pruning nodes, and the algorithm stops when reaching *k*-max which is the maximum subgraph that can be created (i.e., the subgraph after which it becomes empty).

Assuming that the graph was decomposed into cores, each node is then assigned the core number of the maximum subgraph at which it resides. For keywords extraction, nodes in the *k*-max subgraph or nodes in the lower level of the hierarchy of subgraphs can be marked as the most influential, and used as keywords [41]. However, in this paper, we adopt the work done by Tixier, Malliaros, and Vazirgiannis [42] which assigns a score to a node n_i as the sum of the core numbers of its neighbors in the original graphs (nodes adjacent to it). This scoring aims at decreasing the granularity in nodes selection. As suggested, for keyword extraction, if a score is assigned based only on the node's own core/truss number, then many nodes will end up having the same weight, increasing the number of selected nodes [122].

k-*Truss*: *k*-Truss is another graph decomposition method [42] that is triangle-based, and is based on a cohesive social and communication pattern in graphs introduced by Cohen [123]. A triangle in a graph G is the set of three nodes a, b and c that are pairwise-connected. Instead of pruning weak nodes in the decomposition process as in *k*-Core, *k*-Truss peels weak edges first, then removes nodes with no more connecting edges in the resulting subgraph. Precisely, *k*-Truss prunes an edge from the k-1 subgraph if it is not supported by at least k-2 other edges that form triangles with that edge. Similar to *k*-Core, we follow the same scoring method to compute the node scores given their truss numbers.

PositionRank: *PositionRank* is a also a graph-based keyword extraction algorithm [112] that uses the graph-of-words structure. However, after constructing the graph, *PositionRank* adapts the *PageRank* algorithm [124], a well known iterative random walk-based algorithm for ranking web pages, to assign scores to nodes (words) in the graph that indicate its importance. Aside from *PageRank* that assigns equal initial scores/weights to all nodes in the graph, *PositionRank* assigns initial scores to nodes (words in this case) based on their positions in the text being analyzed. Precisely, it favors more nodes that occur at the lead of the text, based on the assumption that keywords occur frequently very close to the beginning of a document. In constructing the graph of words, *PositionRank* selects only nouns and adjectives to be added as nodes to the graph. After the graph analysis, *PositionRank* considers noun phrases that match the regular expression (adjective)*(noun)+ of length up to three to create unigrams, bigrams, and trigrams. It assigns bigrams and trigrams the sum of the scores of its individual unigrams. To create the news background linking search queries using *PositionRank*, and since we only care about weighting single terms, we skip this process of creating bigrams and trigrams, and use the weights assigned by *PositionRank* to the unigrams as is.

TopicRank: *TopicRank* is another graph-based keyword extraction algorithm [113] that is also based, as *PositionRank*, on the idea of the *PageRank* algorithm. However, in *TopicRank*, the graph nodes are not only single words; instead, a document is represented as a complete graph of topics and the relations between those topics. A topic is defined as a cluster of similar single and multi-word phrases. To define these phrases, *TopicRank* uses a part-of-speech (POS) tagger and extracts the longest sequences of nouns and adjectives from the document as keyphrase candidates. Keyphrase candidates are then split into clusters using a Hierarchical Agglomerative Clustering (HAC) algorithm. After creating the different topics and representing them as nodes, a complete graph is created, where edges are added between topics given their closeness to each other. The weight on the edge between topics is calculated using the reciprocal distances between the offset positions of their keyphrases. The basic *PageRank* on this graph is then applied to weigh the different topics. Finally, *TopicRank* selects the keyphrase from each topic that appears first in the document, and assigns it the topic's weight. For constructing the search query, we split the extracted keyphrases into unigrams, then we compute the score or boost value of a unigram term *u* as follows: $score(u) = sum_{i=1}^{k} TR(i)$, where k is the number of the extracted keyphrases that contain the unigram *u*, and $TR(i)$ is the weight assigned by *TopicRank* to an n-gram keyphrase i.

Multipartite Rank: Based on the idea of creating better representation of keyphrases within the different topics in the a text, Boudin [114] proposed *Multipartite Rank* (*MPR*). *MPR* uses a more dense graph structure than with *TopicRank*, called *multipartite* graph to model the topics with their keyphrases. In that graph, the nodes are candidate keyphrases that are connected only if they belong to different topics. Accordingly, the document is represented in this graph as tightly connected sets of topic-related candidates/keyphrases. *MPR* also adds an adjustment on the weights of the keyphrases that occur first on each topic. This is based on the assumption that candidates keyphrases that occur at the beginning of the document are better representatives of the topics. We follow the same method as with *TopicRank* to select the required terms and weights.

sCAKE: Following the line of graph-based keyword extraction methods, *sCake* [115] was motivated by the need to design a paramterless graph construction method. Instead of a sliding window parameter as in *k*-Core or *k*-Truss, in *sCake*, the window slides over two consecutive sentences and the terms co-occurring in these sentences are linked (i.e., the edge between a node n_i and node n_j holds how many times the terms t_i and t_j occurred in two consecutive sentences). Similar to *PositionRank*, *sCake* only considers nouns and adjectives as candidate terms for constructing the graph of words. To score nodes in the graph for keyword extraction, *sCake* initially applies the *k*-Truss method explained above, then uses the created truss-hierarchy of subgraphs to assign different features to the nodes, and aggregates those features for each node to compute its final score. These features are the maximum truss number at which an edge between the node and any of its neighbors resides, the sum of truss levels of the node's neighbors, a positional feature that attempts to favor nodes at the beginning of a document as in *PositionRank*, and a semantic connectivity score that measures the number of distinct concepts that a node links to, assuming that the more the node's neighbors belong to different concepts, the more important it is. Since *sCake* creates only unigrams, for constructing our search queries, we use the score assigned by *sCake* to each output term as is.

Yake: Unlike graph-based methods, *Yake* is a completely statistical method that was introduced recently for keyword extraction [116]. It depends on capturing five statistical features of terms in a document, and integrating those feature scores in a final score for keyphrase determination. The first feature captures how frequent a term was mentioned starting with a capital letter excluding the beginning of a sentence, or marked as an acronym. This is based on the assumption that uppercase terms are usually more relevant as keywords than lowercase ones. The second feature is term position, which captures, as in *PositionRank*, the position of terms in the document, favoring terms that occur at the beginning of the document. Unlike *PositionRank* though that captures the position of terms itself, *Yake* captures the position of the sentences in which the terms occur. The next feature is a normalized version of the term frequency in a document. The fourth captures the context around the term, based on the assumption that the higher the number of different terms that co-occur with the candidate term t on both sides, the less significant term t will be. Finally, the last feature captures the percentage of different sentences in which a term occurs, assuming that terms which occur in different sentences are more important. In *Yake*, the smaller the value of a term, the more significant it is. Since in our work we assign boost weights to extracted terms, we assign a score to the term t as follows: $score(t) = 1/Yake(t)$.

Symbol	Meaning
N	Length of the article in words after preprocessing
n	Number of unique terms in the article
s	Number of sentences in the article
w	Width of the sliding window used for constructing the graph of words
c	Number of candidate keyphrases determined through POS tagging
m	Number of edges in the constructed graph

Table 6.2: Symbols used for representing time complexity.

6.1.3. Time Complexity of Keyword Extraction Techniques

Although we assume an offline extraction of keywords from the query article to construct the search query for background links retrieval, it is worth comparing the time complexity of the aforementioned keyword extraction techniques in case it is performed online, or there is a large number of query articles to be processed by the news provider. Table 6.2 shows the terminology we used for this analysis, and Table 6.3 shows the time complexity for each method, along with any specific requirements before the keyword extraction process.

Method	Time Complexity	Pre-Requirements
Yake	$\mathcal{O}(N + s\ n\ w)$	Segtok sentence segmenter
PositionRank	$\mathcal{O}(N\ w^2 + c + m)$	POS tagging
TopicRank	$\mathcal{O}(c^3 + m)$	POS tagging
MPR	$\mathcal{O}(c^3 + m)$	POS tagging
k-Core	$\mathcal{O}(N\ w^2 + m)$	-
k-Truss	$\mathcal{O}(N\ w^2 + n + m^{1.5})$	-
sCake	$\mathcal{O}(N + s\ c^2 + m^{1.5})$	POS tagging

Table 6.3: Time complexity of keyword extraction algorithms.

Most of the graph-based keyword extraction methods are quadratic in time. For instance, constructing a graph of words for *PositionRank*, *k*-Core, or *k*-Truss is of time complexity $\mathcal{O}((N - w + 1)\ (w(w-1)/2))$. Since w is relatively small with respect to the length of the article, the complexity is $\mathcal{O}(Nw^2)$. After constructing this graph and assuming m edges in the constructed graph, applying *PositionRank* on the constructed graph additionally requires $\mathcal{O}(c+m)$ [125], as it is a variation of the well known *Pagerank* algorithm with only favoring nodes that appear at the beginning of the article. As for *k*-Core, there is a number of implementations proposed in the literature to decompose the graph of words into cores. The most efficient applies the decomposition in $\mathcal{O}(m)$ time [126]. While *k*-Truss is relatively more complex than *k*-Core when computed on large graphs, an in memory-based complexity of $\mathcal{O}(m^{1.5})$ can be considered ([127]), which suits the relatively small graph of words for news articles.

As for *TopicRank* and *MPR*, both need POS tagging before constructing the text graph. Although the number of candidate keyphrases c, obtained after POS tagging, is less than the number of unique terms n, both algorithms apply agglomerative clustering to the candidate keyphrases, which is of worst time complexity $\mathcal{O}(c^3)$. The construction of the topic graph then requires $\mathcal{O}(N.c)$ for computing the candidate keyphrase offset positions, and $\mathcal{O}(c^2)$ for computing the positional distance between pairs of candidate keyphrases. Both algorithms additionally require $\mathcal{O}(c + m)$ for applying *PageRank*.

For *sCake*, aside also from the need to apply POS tagging, it requires $\mathcal{O}(N)$ to calculate term frequencies, $\mathcal{O}(c^2 * s)$ to construct the graph, $\mathcal{O}(m^{1.5})$ for *k*-Truss, and $\mathcal{O}(N)$ for computing the positional weights.

Finally, *Yake* depends (as mentioned in [116]) on the rule-based segtok algorithm[1] that segments a text into sentences based on orthographic features, and tokenizes the text into terms with delimiter tags (e.g., Acronyms, Uppercase, Digits, etc.). After identifying the sentences, *Yake* takes $\mathcal{O}(N)$ for computing the term frequencies and casing features. It additionally requires $\mathcal{O}(s.n.w)$ for context feature calculations. We omit here the complexity to generate n-gram keyphrases as our goal is to obtain unigram terms. Hence, compared to the methods mentioned above, it is considered the most efficient method for extracting unigram terms.

As for *TF*, the time complexity is $\mathcal{O}(N)$ with no additional requirements, making it the most efficient method among the others. For *TF-IDF*, assuming a constant time ($\mathcal{O}(1)$) to get the document frequency of each term from the index, the complexity remains $\mathcal{O}(N)$, making it similarly efficient as *TF*.

[1]https://pypi.org/project/segtok/

6.2. Experimental Evaluation

In this section, we present our experimental setup, covering the baseline method and some implementation issues, followed by a discussion on the experimental results that address our research questions.

6.2.1. Experimental Setup

Baseline: We used, in our experiment here, the set of queries released in 2018, 2019 and 2020 in the Washington Post dataset. Again we constitute the full-article retrieval approach as the baseline method with the same setup shown in Chapter 4, aiming at achieving the same or comparable effectiveness, but with much shorter, hence efficient, search queries.

Implementation Issues: Due to lack of open source implementation, we implemented our *k*-Core and *k*-Truss keyword extraction methods in Java. For *PositionRank*,[2] *Yake*,[3] and *sCake*,[4] we used the authors'-provided implementation developed in Python for the first two methods and *R* for the third. For *TopicRank* and *MPR*, we used the PKE Python library.[5] We set the sliding window to 3 for methods that required it (i.e., *PositionRank*, *k*-Core, and *k*-Truss). For *Yake*, we set the maximum n-gram size to 1 to get only unigram keywords, and the window size to 1, as recommended in [116]. For the other methods, we used the default parameters settings as suggested by their given libraries. For instance, for *MPR*, we set alpha to 1.1, and the threshold to 0.74. For *PositionRank*, we set alpha to 0.85.

Evaluation: We used nDCG@5 as our primary evaluation measure for effectiveness, as shown in Chapter 3. For reliable evaluation, we evaluated the studied methods using 3-fold cross validation over the TREC query articles. For easier future comparisons, the folds were chosen to be the query sets released by TREC in each year for the background linking task (2018, 2019 and 2020), i.e., two TREC query sets of two years were used for parameter tuning, and the third was used for testing. The only parameter we tuned in our experiments was the number of extracted terms that are used to construct the search query. To report efficiency, we use the query response time in milliseconds. We ran all our experiments on a MacBook Pro machine with a 2.5GHz Quad-Core Intel Core i7 processor, and 16GB 1600MHz DDR3 memory. The reported time per query is the average time of 3 runs of the same experiment.

6.2.2. Experimental Results

We discuss in detail now the different experiments that we conducted to address our research questions RQ4.1 and RQ4.2. We first test the effectiveness of the simple lead-paragraphs extraction method. Then, we discuss the effectiveness and efficiency of the queries generated by the different keyword extraction methods for our task.

Lead Paragraphs as a Search Query: When authors write news articles, they often adopt the inverted pyramid style [92], [128], in which the essential and most attention-grabbing elements are introduced *first* in the article. Accordingly, in this section we

[2]https://github.com/ymym3412/position-rank
[3]https://github.com/LIAAD/yake
[4]https://github.com/SDuari/sCAKE-and-LAKE
[5]https://github.com/boudinfl/pke

address **RQ4.1**, that is to test if the lead paragraphs of the articles can simply and sufficiently be used as a search query instead of the whole article for the retrieval of the background links. In other words, we wanted to check if applying keyword extraction is at all required.

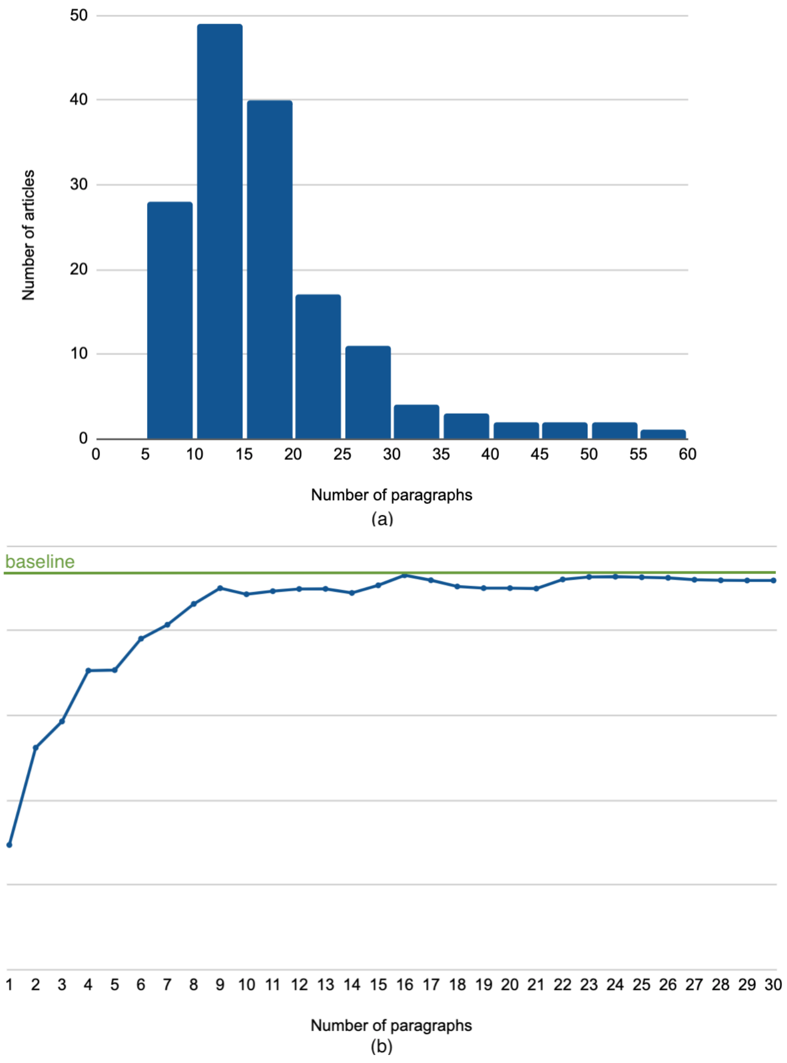


Figure 6.1: (a) Histogram of the length of the query articles. (b) Performance using leading paragraphs as search query.

To answer this question, we experimented with constructing search queries simply using the top paragraphs of the query article, after stop words removal. Figure 6.1(a) shows the histogram of the length of the query articles in paragraphs.[6] The figure indicates that most articles have less than 20 paragraphs, with an average of about 18 paragraphs. Figure 6.1(b) illustrates the performance when we vary the number of top paragraphs from 1 (just the first paragraph of the article) up to 30 (almost the full article). We observe that increasing the number of paragraphs used for constructing the search query improves the performance. We also note that the peak nDCG@5 occurred at using 16 paragraphs, which is very close to the average number of paragraphs. This indicates

[6]Only one very long query article (141 paragraphs) was omitted from the histogram for clarity

that using only a few lead paragraphs is not sufficient for an effective background links retrieval.

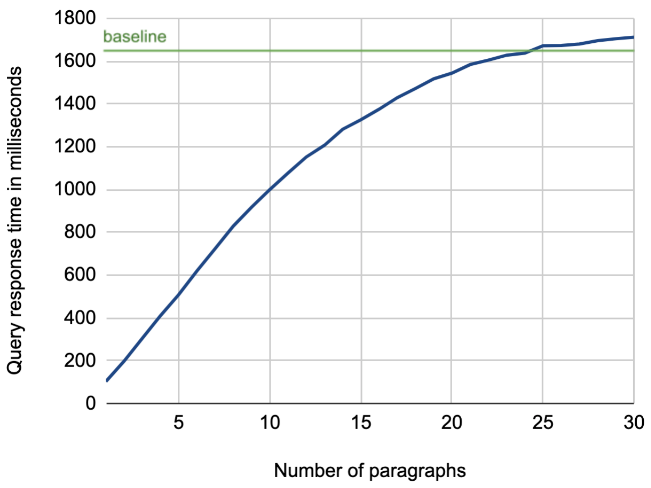


Figure 6.2: Average query response time for retrieving background links using lead paragraphs as a search query.

As for the efficiency of the retrieval process, Figure 6.2 shows the average query response time using the lead paragraphs as search queries. As illustrated, there is a high growth in response time while increasing the number of paragraphs, which increases the number of terms in the search query[7]. This experiment confirms the need for a keyword extraction mechanism that can select few terms to construct the search query while maintaining the effectiveness of using the full article.

Note that while we might expect that the baseline method should always exhibit the highest response time, Figure 6.2 shows it is not the case when using 25 to 30 paragraphs. This is due to the optimization of query processing performed by the Lucene platform, which adopts *BlockMax WAND* approach that skips scoring some documents in the posting lists of terms when their score contribution is expected to be not competitive (hence not affecting the document ranking), resulting in a faster retrieval process [129]. This indeed is more evident in long queries, as in the baseline case.

Keywords Extraction for Search Query Construction: The next experiment aims at addressing ***RQ4.2***, which is concerned with measuring the effectiveness of the keyword extraction techniques in search query reduction for background linking. To conduct this experiment, we applied each keyword extraction method explained earlier to analyze the query articles and construct search queries. Figure 6.3 shows the cross-validation results, varying the number of extracted keywords from 30 (since our preliminary experiments in this regard showed that the effectiveness degrades noticeably if the number of search terms are less than 30) to 100 (for efficiency purposes). It also reports the performance of the *Baseline* method. The range of the optimal (after tuning) values of the number of extracted keywords (over the 3 folds) for each method is reported on top of the respective bar. Surprisingly, all methods, except *sCake*, exhibit very similar performance with small differences in effectiveness. Accordingly, we applied the paired two-sample t-test with

[7]There were two outlier queries that took more time compared to others for processing. However, removing both did not noticeably affect the average processing time

5% significance level between the baseline and each of the other methods; we found that the differences in performance between the baseline and each of *Yake*, *k*-Truss, *k*-Core, *PositionRank*, and *MPR* methods are *not* statistically significant, which means that those five methods essentially exhibit the same effectiveness of the baseline despite using much shorter queries. This, in turn, achieves our goal of having no sacrifice in the retrieval effectiveness as a result of the query reduction process.

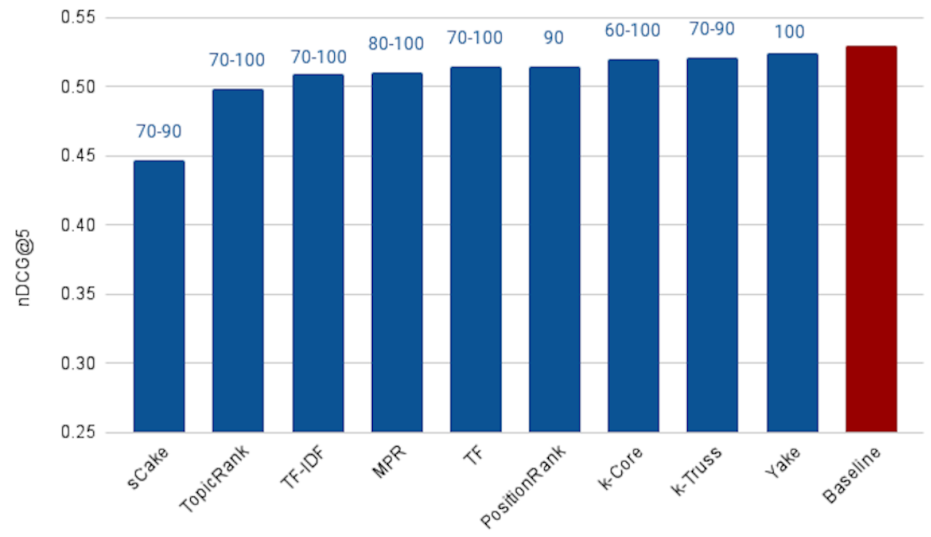


Figure 6.3: Effectiveness of keyword extraction techniques, varying the number of extracted keywords from 30 to 100.

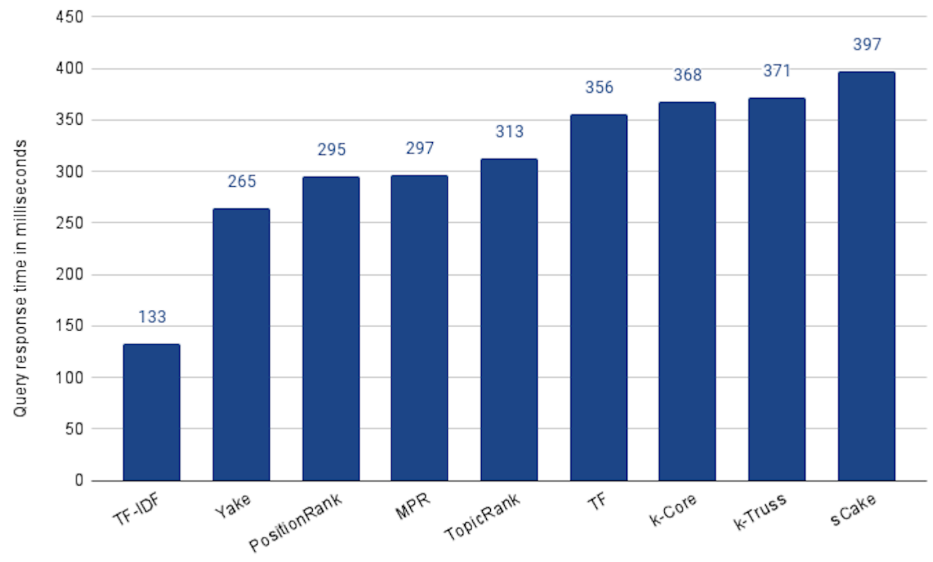


Figure 6.4: Average query response time for 100-term search queries generated by the keyword extraction methods.

Being effectively similar, using any of the methods that exhibit no statistical significant difference with the baseline is expected to be more efficient than the baseline while maintaining the retrieval effectiveness. This is due to the much shorter search queries. The average length of the articles in the Washington Post collection was around 400 terms (after removing stop words). This means that there is a 75% reduction in the query length, adopting the above keyword extraction technique, as it produces a maximum of 100 terms as seen above. However, the selection of the query terms might impact the speedup. To check that, we computed the average query response time for each of the keyword extraction methods for 100-term queries, as shown in Figure 6.4. In this experiment, we did not tune the number of terms, as we choose to test the worst case scenario, which is using our ceiling number of terms (i.e., 100). It is then intuitive that the time taken for processing a subset of those terms will be lower. Among the five methods above that exhibit the indifferent performance with the baseline, *Yake* is the fastest with about **6.2x** speedup over the baseline (which takes 1645ms, but omitted from the figure due to scale). Recall that *Yake* is also the closest in nDCG@5 performance to the baseline, among all other methods (as illustrated in Figure 6.3), and also the fastest (after the two *TF*-based methods) in the keyword extraction process (as shown in the time complexity analysis earlier). It indeed yields a perfect solution for the practical scenarios. We also notice that the queries generated by the *TF-IDF* method are the most efficient ones. This is expected as the method prioritizes terms of high IDF, i.e., short postings lists. Therefore, if we need even higher speedup (**12.4x** over the baseline in this case) while tolerating some degradation in effectiveness, *TF-IDF* is then the choice.

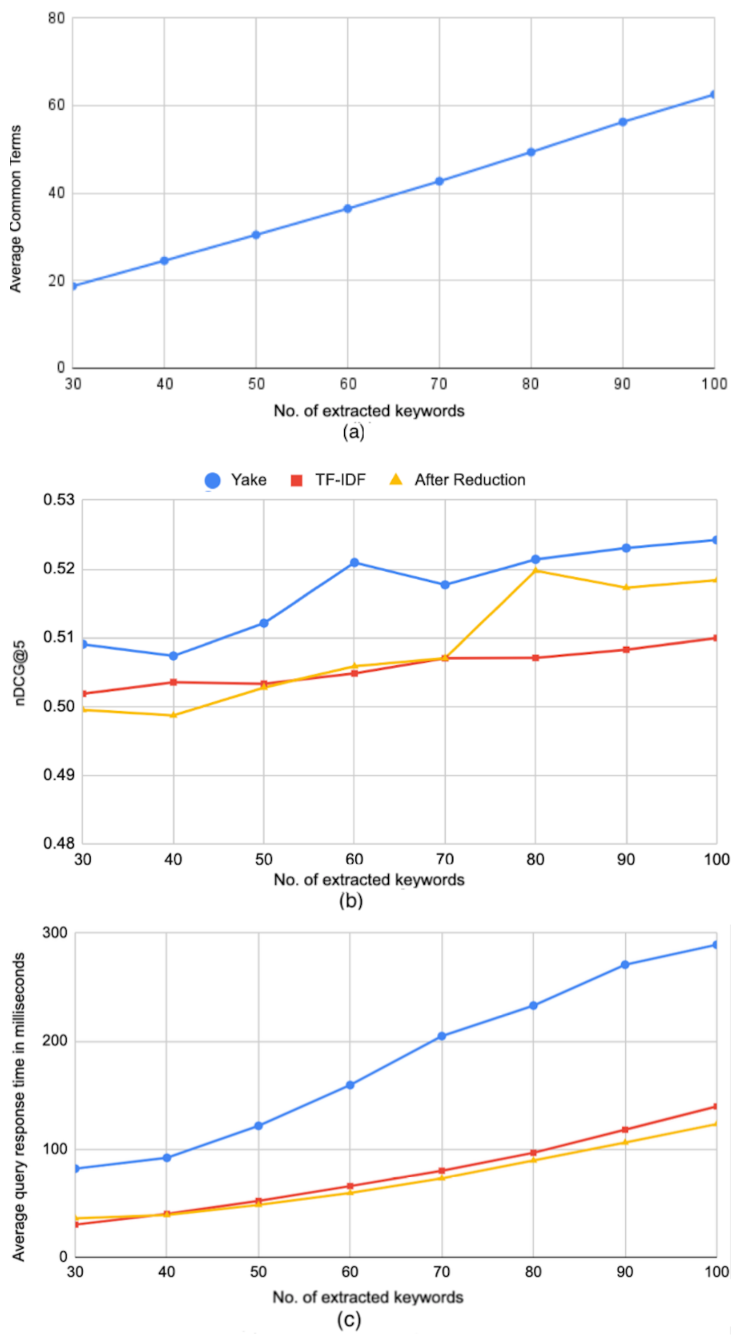


Figure 6.5: Further reduction of search query terms. (a) The average query length after keeping only the common extracted terms by *TF-IDF* and *Yake* methods. (b) Effectiveness of *Yake* and *TF-IDF* methods before reduction, and *Yake* after reduction. (c) The average processing time of queries after applying the reduction.

The results above inspired us to check if the terms in the queries generated by *Yake* are different than the ones generated by the *TF-IDF* method (hence the difference in the processing time), and if those different terms actually influence the retrieval effectiveness. To answer this question, we computed the average cosine similarity between the search query vectors generated by both methods over queries of different lengths. The resulted average similarity was 0.8, indicating that *TF-IDF* yields somewhat different queries. That motivated us to conduct another exploratory followup experiment, where we only keep the terms that are *commonly* extracted by *TF-IDF* and *Yake*. Figure 6.5-(a) shows the average length of the queries after this filtering for different number of originally extracted keywords. The figure shows considerable reduction; for example, 27 terms were filtered out on average when the original queries have 100 terms, keeping only 63 terms in common. As for the effectiveness past the reduction of the queries, it can be seen in Figure 6.5-(b) that when the number of the originally-extracted terms exceeds 80, the reduced queries yield very close effectiveness, in terms of nDCG@5, to the original queries generated by *Yake*. In fact, the difference in performance between the baseline and the query reduced from 100 extracted terms is *not* statistically significant, indicating that the removed terms were quite ineffective, yielding a more efficient retrieval. This can be clearly noticed in Figure 6.5-(c), which shows that the reduced queries are much more efficient than the queries generated by *Yake*, and a bit more efficient than the queries generated by *TF-IDF*, yielding about **13.3x** average speedup over the baseline.

In conclusion, several keyword extraction methods exhibit similar effectiveness (with no statistical significance difference) to the baseline while being much more efficient. Among those, the queries generated by *Yake*, in particular, yield 6.2x speedup over the baseline. Moreover, *Yake*'s queries can be further reduced by filtering out terms that were not selected by *TF-IDF* method, yielding an even better average of 13.3x speedup in response time over the baseline, with a little cost of running both (already fast) methods of keyword extraction. Again, the difference in effectiveness in that case is not statistically significant. It is important to note here that while the reported reduction in the response time is in terms of milliseconds in this in-lab experiment, such speedup is highly and typically needed during Web search tasks (such as searching for background sources over the Web) in real-time online scenarios.

6.3. Conclusion

In this chapter, we showed how we can construct a relatively short search query that can be efficiently used for retrieving a set of background links for a query article. We showed also that, compared to using the full-article as a search query, the constructed short query does not sacrifice or significantly reduce the retrieval effectiveness. The results produced through the constructed query can be represented to the reader as is, or it can be further considered for reranking purposes. It is worthy to note here, that when proposing our approaches for effective news background linking (Chapters 4 and 5), we opted to rerank the set of candidate links retrieved through the full article research approach instead of the links produced through any reduced query, as an example initial retrieval phase. The work proposed in either approaches though, that addressed the semantic-based matching between the query article and its candidate background links, can still be integrated with any initial retrieval process. Next, we conclude our dissertation, highlighting the limitations that our work faces and showing future research directions accordingly.

CHAPTER 7: CONCLUSION AND FUTURE WORK

In this chapter, we conclude the dissertation with a summary of the proposed work in Section 7.1, the limitations of our work in Section 7.2, and the future research directions in Section 7.3. We finally list our research publications in Section 7.4.

7.1. Conclusions

News background linking aims to find useful resources that can be linked to a news article to give its readers more background and context on its content. This problem is an important research problem as it helps providing readers with access to relevant background information, aiding them in gaining more knowledge and making informed decisions about the issues being covered in the news everyday. This will enhance the overall quality and depth of news reporting. Often, authors or journalists add the background links manually to their stories. If these links are not available, readers tend to lookup the web for background resources. The work proposed in this dissertation helps in paving the way to automate this background linking process for authors and news providers in order to aid readers to easily navigate to the required information they seek.

Since news background linking is a relatively new and challenging problem to the research community, we conduced the first, up to our knowledge, formal qualitative analysis of a subset of the query articles and its background links to understand the notion of background relevance. Through our study, we observed that useful background articles do not need to have the same main topic as the query article. They further do not necessarily need to have high lexical similarity with it. Instead, they need to provide more details on one or more of the subtopics discussed within the query article, in addition to introduce new perspectives or subtopics related to its context. We further found that query articles that are driven by the occurrence of some timely event are generally harder to process than other articles that discuss general topics. For those query articles, we found that the majority of its highly-relevant background articles do not just mention the event, but rather discuss subtopics related to its occurrence. We further found some relevant background articles for those query articles that do not at all mention the event, but often discuss the context at which the event appears and in many cases, they add knowledge on the future consequences of the event.

Driven by the insights that we drew through our qualitative analysis of news background relevance, we proposed two approaches to address the effectiveness aspect of news background linking. Our proposed approaches depend on reranking a set of candidate background links retrieved through using the full query article as a lexical search query, in an ad-hoc setting. To rerank this candidate set of background links, both of our proposed approaches integrate the lexical matching signal between the query article and the candidate background link, obtained through ad-hoc retrieval, with a semantic matching signal calculated between both articles.

For the semantic matching process, our first proposed approach depends on finding dense representations of news articles and computing matching scores between the query article and its candidate links. We investigated a number of Transformer-based encoder models, in a zero-shot setting with no further pre-training nor fine-tuning, to represent the news articles in the semantic space. We further investigated a number of score aggregation functions for reranking the candidate articles. We found that using a hierarchical aggregation of sentence-level representations is best for semantic matching. We further found that by simply summing the normalized lexical and semantic matching scores of candidate background links, we achieve an effective background linking (nDCG@5 of 0.5016 vs. 0.4856 achieved by the full-article search baseline approach), marking a new state of the art for the problem. We, moreover, discovered that query articles differ in the degree by which they need the incorporation of the semantic relevance signal in the reranking process of its candidate background links. While some articles considerably benefit from the semantic signal, others get hurt. We show through our analysis of the results we obtained, that if the degree by which a semantic matching signal is needed for a query article is determined, the performance of background linking can be further increased (up to nDCG@5 of 0.5487).

Our second proposed approach for addressing the effectiveness aspect of news background linking leverages the potential of large language models (LLMs) for semantically matching the query article to its candidate links. Precisely, we employed an LLM to analyze the input article to reveal its discussed topics, then analyze the candidate links to reveal information related to the query article topics producing a summary of related information, then, finally rerank the guided LLM-generated summaries of the candidate links. In this approach, we aggregate the ranks produced by the LLM with the original ranks of the candidate links, preserving also the effect of the lexical matching signal, and ensuring that the background links do not deviate much from the content of the query article. We experimented with four black-box LLMs: Gemini-Pro 1, Gemini-Pro 1.5 GPT-3.5 Turbo and GPT-4 Turbo. We used a subset of query articles for curating the prompts to the LLMs. On the test set of queries, our results show that using the GPT-4 Turbo LLM, our proposed approach again mark a state of the art performance for the news background linking problem (nDCG@5 of 0.518 vs. nDCG@5 of 0.498 by the full-article search baseline approach). We further showed through a failure analysis why the other LLM models did not perform as well in our problem.

Both of our aforementioned approaches showed almost the same effectiveness , on the test set of query articles, with no statistically significant difference in performance. While the LLM-based approach is simpler, it comes with the cost of using a black-box model, with its cost constraints and the chance of API response failures.

As for the efficiency aspect of news background linking, we worked on retrieving an effective set of candidate background links with a much shorter search query than the full query article. We experimented with a number of unsupervised keyword extraction approaches for search query reduction. Our results showed that we reach large speedup in query response time (6.2x speedup over the full-article search approach), using simple statistical keyword extraction techniques, such as *Yake*, with a difference in effectiveness that is not statistically significant from using the full query article as a search query. We further showed that by adopting *TF-IDF* traditional keyword extraction method to filter *non-informative* terms from the queries generated by *Yake*, one may further achieve even higher speedup (13.3x over the full article search approach), still with similar effectiveness.

7.2. Limitations

In this section, we list the limitations of our study, given our proposed solutions for addressing the news background linking problem.

- **News Articles Semantic Representation** We showed in our proposed approach for reranking a candidate set of background links for a query article using pre-trained models, how we can best represent news articles semantically for news background linking. Specifically, we showed that using a pre-trained encoder model, which takes short fragments of text (sentences) as input, one could have a representation for news articles that, to a great extent, captures its semantics. It is important to highlight here though that our evaluation of the different models that we experimented with was based mainly on the effectiveness achieved when matching news articles in the news background linking task. This task is different from other tasks that involve text matching, for which those models may have been or may be used. In other words, the conclusions that we drew might not generalize to other tasks that also requires the semantic representation of news articles or generally textual documents.

- **Encoder-based Models for News Articles Representation** Our study using encoder-based models is also limited in the number of models involved in the comparison. We believe that it can be further improved by including encoders from models that were introduced recently and that were mainly trained for long text matching tasks, such as Match-ignition [90], and CoLDE [91]. As far as we know though, the checkpoints of those models have not yet been released to the research community.

- **Differentiating Query Articles** In our study in Chapter 4, we showed that the effectiveness of the proposed reranking approach can be enhanced if query articles were differentiated according to their need to incorporate the semantic relevance signal in the matching process of its candidate links. We attempted to build a model that can distinguish query types using, as labels, a parameter that controlled the aggregation between the lexical and the semantic relevance signals during the reranking process in our experiments. We experimented with both pre and post retrieval query features to train the desired model. Nonetheless, we were not able to build this model due to the limited training and testing samples (only 207 queries).

- **Single Dataset** We limited our experiments in news background linking to the Washington Post dataset released by TREC for the news background linking task. In addition to the limited number of query articles in this dataset as mentioned before, the representation of the different relevance levels is very unbalanced. This limits the capability of building a multi-class model that can effectively determine the actual relevance of a background link to a query article.

- **Using Black-box LLMs** Our proposed LLM-based news background linking approach showed state of the art effectiveness when we tested it using a black-box LLM. This LLM is not cost effective though, not to mention the possibility of API response failures. Other open-source LLMs though should be considered.

7.3. Future Research Directions

The effectiveness reached so far for this problem is still far from being optimal, even with our proposed approaches. Accordingly, there is still large room for improvement in this problem, and there are several directions for future work. We elaborate on some of them here.

Query Type Distinction The type of query articles needs to be further investigated as it clearly affects the linking process as shown in our qualitative analysis for news background linking, and also from our analysis of the results we obtained through our experiments. For instance, our qualitative analysis showed that queries that are driven by the occurrence of some events are the hardest to find background links for. Identifying the discussed events from within those articles and their direct related context might lead to find the background links for it. Further criteria may be investigated for distinguishing query articles, such as whether a query article reports hard news (ex: politics, economy) or soft news (sports, entertainment). Also, an article may be distinguished considering its type and length. A news article can be a long feature article giving an author's analysis on a specific old topic or event, hence including multiple subtopics, or it can be a short news article reporting straight facts on a specific event or incident without including any comments on opinion from the author.

Deep Topics Analysis : Our qualitative analysis showed that the extent to which the candidate background article discusses the subtopics of the query article, and to what extent it introduces new topics that add to the readers' knowledge affect the background relevance. We actually adopted this insight to build our second proposed linking approach. Another future direction that can be investigated to benefit from this insight might be to split the query article into different topics, acquire topic relevance scores for each candidate document, then aggregate those scores to obtain overall document scores.

Adopting Long-Document Matching Models : Recently, a number of researchers introduced models that aim to match long documents for different tasks. A future research direction for news background linking is to experiment with those models for the task, as they were never tested for it. Moreover, those models may be fine-tuned to achieve a task specific increase in performance.

Adopting LLMs for News Background Linking : A future work in the direction of working with LLMs for news background linking will be to experiment with open-source LLMs within our proposed approach to see if they are as effective as Black-box LLMs. Another future work is to experiment with shuffling the order in which candidates are presented to the model for reranking, as it might affect the ranking effectiveness. The LLMs can further be adopted to analyze the query articles and its candidate background links, considering other relevance criteria and with better crafted prompts.

Expanding the Washington Post Dataset As mentioned before, we faced the scarce data problem in the Washington Post dataset when attempting to proposing approaches for addressing the news background linking problem. To build a classification model to differentiate query types for example, or to build a model that predicts the relevance of a background link to a query article, more data is needed for training and testing such models. Accordingly, a very important future direction for this problem is to annotate more query articles from the Washington Post dataset, or to find other datasets of news articles that can be adapted for the news background linking problem.

Building Diverse Datasets : Other resources than news articles should also be considered as background links. The Washington Post dataset contains only news articles and blog posts as background links. Linking to only resources that are published by the same newspaper might help a news provider that aims to restrict readers to his website for competition purposes. However, allowing the reader to have links to other external background resources will broaden his knowledge on the news story he/she is reading and allow him/her to get more point of views. One of the future directions in this problem, hence, is to build datasets that contain different types of background links, written with also other languages than English.

7.4. Publications

In this section, we list the related publications to this work.

Published Papers:

- **Marwa Essam**, Tamer Elsayed. Why is That a Background Article: A Qualitative Analysis of Relevance for News Background Linking. Proceedings of the 29th ACM International Conference on Information & Knowledge Management. 2020 [6].
- **Marwa Essam**. Background Linking of News Articles: Advances in Information Retrieval. 43rd European Conference on IR Research, ECIR. Lecture Notes in Computer Science(), vol 12657. Springer. 2021. [27] **Best DC paper award.**
- **Marwa Essam**, Tamer Elsayed. Unsupervised query reduction for efficient yet effective news background linking. PeerJ Computer Science. 13 Jan 2023; 9, p.e1191 [9].
- **Marwa Essam**, Tamer Elsayed. Zero-Shot Reranking with Dense Encoder Models for News Background Linking. (Journal article accepted for publication in PeerJ Computer Science - November 2024) [7].
- **Marwa Essam**, Tamer Elsayed. bigIR at TREC 2019: Graph-based Analysis for News Background Linking. Proceedings of the 28th Text REtrieval Conference TREC. 2019 [40].
- Fatima Haouari, **Marwa Essam**, Tamer Elsayed. bigIR at TREC 2020: Simple but Deep Retrieval of Passages and Documents. Proceedings of the 29th Text REtrieval Conference TREC. 2020 [130].
- **Marwa Essam**, Tamer Elsayed. bigIR at TREC 2021: Adopting Transfer Learning for News Background Linking. Proceedings of the 30th Text REtrieval Conference TREC. 2021 [65].

The last three publications in the above list show our participation in TREC with some preliminary work for both the news background linking, and the document retrieval tasks. The exact details of all the runs submitted and reported in these papers though are not covered in this work.

Under Review:

- **Marwa Essam**, Tamer Elsayed. Can Large Language Models Effectively Rerank News Articles for Background Linking? (Conference paper submitted in October 2024)

REFERENCES

[1] indianexpress, *TREC conference by NIST*, `https://indianexpress.com/section/explained/`, 2020.

[2] NIST, *TREC conference by NIST*, `http://trec-news.org/`, 2019.

[3] I. Soboroff, S. Huang, and D. Harman, "TREC 2018 news track overview," in *Proceedings of the Twenty-Seventh Text REtrieval Conference (TREC)*, 2018.

[4] I. Soboroff, S. Huang, and D. Harman, "TREC 2019 news track overview," in *Proceedings of the Twenty Eighth Text REtrieval Conference TREC*, 2019.

[5] I. Soboroff, S. Huang, and D. Harman, "TREC 2020 news track overview," in *Proceedings of the Twenty-Ninth Text REtrieval Conference (TREC)*, 2020.

[6] M. Essam and T. Elsayed, "Why is that a background article: A qualitative analysis of relevance for news background linking," in *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, 2020, pp. 2009–2012.

[7] M. Essam and T. Elsayed, "Zero-shot reranking with dense encoder models for news background linking," *PeerJ Computer Science*, In press.

[8] M. Essam and T. Elsayed, "Can large language models effectively rerank news articles for background linking?," (Submitted).

[9] M. Essam and T. Elsayed, "Unsupervised query reduction for efficient yet effective news background linking," *PeerJ Computer Science*, vol. 9, e1191, 2023.

[10] A. Chakraborty, S. Ghosh, N. Ganguly, and K. P. Gummadi, "Optimizing the recency-relevance-diversity trade-offs in non-personalized news recommendations," *Information Retrieval Journal*, vol. 22, no. 5, pp. 447–475, 2019.

[11] A. Chakraborty, D. Ganguly, A. Caputo, and G. J. Jones, "Kernel density estimation based factored relevance model for multi-contextual point-of-interest recommendation," *Information Retrieval Journal*, pp. 1–47, 2022.

[12] L. Hu, C. Li, C. Shi, C. Yang, and C. Shao, "Graph neural news recommendation with long-term and short-term interest modeling," *Information Processing & Management*, vol. 57, no. 2, p. 102 142, 2020.

[13] M. Ma, S. Na, H. Wang, C. Chen, and J. Xu, "The graph-based behavior-aware recommendation for interactive news," *Applied Intelligence*, vol. 52, no. 2, pp. 1913–1929, 2022.

[14] T. Nicholls and J. Bright, "Understanding news story chains using information retrieval and network clustering techniques," *Communication Methods and Measures*, vol. 13, no. 1, pp. 43–59, 2019.

[15] A. Naskar, R. Saha, T. Dasgupta, and L. Dey, "Ontology guided purposive news retrieval and presentation," in *CEUR Workshop Proceedings*, 2019.

[16] Y. Qian, X. Deng, Q. Ye, B. Ma, and H. Yuan, "On detecting business event from the headlines and leads of massive online news articles," *Information Processing & Management*, vol. 56, no. 6, p. 102 086, 2019.

[17] F. K. Örs, S. Yeniterzi, and R. Yeniterzi, "Event clustering within news articles," in *Proceedings of the Workshop on Automated Extraction of Socio-political Events from News 2020*, 2020, pp. 63–68.

[18] S. Lu, S. Li, Y. Xu, K. Wang, H. Lan, and J. Guo, "Event detection from text using path-aware graph convolutional network," *Applied Intelligence*, vol. 52, no. 5, pp. 4987–4998, 2022.

[19] D. B. Bisandu, R. Prasad, and M. M. Liman, "Clustering news articles using efficient similarity measure and n-grams," *International Journal of Knowledge Engineering and Data Mining*, vol. 5, no. 4, pp. 333–348, 2018.

[20] N. M. Salih and K. Jacksi, "State of the art document clustering algorithms based on semantic similarity," *Jurnal Informatika*, vol. 14, no. 2, pp. 58–75, 2020.

[21] M. Khan, A. U. Rahman, M. Ullah, and R. Naseem, "The role of named entities in linking news articles during preservation," in *International conference on the Sciences of Electronics, Technologies of Information and Telecommunications*, Springer, 2018, pp. 50–58.

[22] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[23] A. Bimantara, M. Blau, K. Engelhardt, *et al.*, "Htw saar @ TREC 2018 news track," in *Proceedings of the Twenty-Seventh Text REtrieval Conference (TREC)*, 2018.

[24] M. Zaheer, G. Guruganesh, K. A. Dubey, *et al.*, "Big bird: Transformers for longer sequences," *Advances in neural information processing systems*, vol. 33, pp. 17 283–17 297, 2020.

[25] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[26] Y. Zhu, H. Yuan, S. Wang, *et al.*, "Large language models for information retrieval: A survey," *arXiv preprint arXiv:2308.07107*, 2023.

[27] M. Essam, "Background linking of news articles," in *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*, Springer, 2021, pp. 672–676.

[28] R. Baly, G. Karadzhov, J. An, *et al.*, "What was written vs. who read it: News media profiling using text analysis and social media context," *arXiv preprint arXiv:2005.04518*, 2020.

[29] A. Yadav, C. Jha, A. Sharan, and V. Vaish, "Sentiment analysis of financial news using unsupervised approach," *Procedia Computer Science*, vol. 167, pp. 589–598, 2020.

[30] J. C. Reis, A. Correia, F. Murai, A. Veloso, and F. Benevenuto, "Supervised learning for fake news detection," *IEEE Intelligent Systems*, vol. 34, no. 2, pp. 76–81, 2019.

[31] K. Nam and N. Seong, "Financial news-based stock movement prediction using causality analysis of influence in the korean stock market," *Decision Support Systems*, vol. 117, pp. 100–112, 2019.

[32] S. Mehta, M. R. Islam, H. Rangwala, and N. Ramakrishnan, "Event detection using hierarchical multi-aspect attention," in *The World Wide Web Conference*, 2019, pp. 3079–3085.

[33] P. Yang and J. Lin, “Anserini at TREC 2018: Centre, common core, and news tracks,” in *Proceedings of the Twenty-Seventh Text REtrieval Conference (TREC)*, 2018.

[34] P. Yang, H. Fang, and J. Lin, “Anserini: Enabling the use of lucene for information retrieval research,” in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 2017, pp. 1253–1256.

[35] P. Lopez-Ubeda, M. C. Diaz-Galiano, M. T. M. Valdivia, and L. A. Urena-Lopez, “Using clustering to filter results of an information retrieval system,” in *Proceedings of the Twenty-Seventh Text REtrieval Conference (TREC)*, 2018.

[36] R. Mihalcea and P. Tarau, “Textrank: Bringing order into text,” in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004. [Online]. Available: `http://aclweb.org/anthology/W04-3252`.

[37] K. Lua and H. Fang, “Paragraph as lead - finding background documents for news articles,” in *Proceedings of the Twenty-Seventh Text REtrieval Conference (TREC)*, 2018.

[38] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes, “Improving efficiency and accuracy in multilingual entity extraction,” in *Proceedings of the 9th International Conference on Semantic Systems*, 2013, pp. 121–124.

[39] K. Lu and H. Fang, “Leveraging entities in background document retrieval for news articles,” in *Proceedings of the Twenty Eighth Text REtrieval Conference (TREC)*, 2019.

[40] M. Essam and T. Elsayed, “bigIR at trec 2019: Graph-based analysis for news background linking.,” in *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC)*, 2019.

[41] F. Rousseau and M. Vazirgiannis, “Main core retention on graph-of-words for single-document keyword extraction,” in *European Conference on Information Retrieval*, Springer, 2015, pp. 382–393.

[42] A. Tixier, F. Malliaros, and M. Vazirgiannis, “A graph degeneracy-based approach to keyword extraction,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 1860–1870.

[43] S. Missaoui, A. MacFarlane, S. Makri, and M. Gutierrez-Lopez, “DMINR at TREC news track,” in *Proceedings of the Twenty Eighth Text REtrieval Conference (TREC)*, 2019.

[44] Y. Ding, X. Lian, H. Zhou, Z. Liu, H. Ding, and Z. Hou, “ICTNET at TREC 2019 news track,” in *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC)*, 2019.

[45] S. Wagenpfeil, P. Mc Kevitt, and M. Hemmje, “University of hagen@ trec2021 news track,” in *NIST Special Publication: Proceedings of the 30 th Text REtrieval Conference (TREC 2021), https://trec. nist. gov/pubs/trec30/trec2021. html. NIST*, 2021.

[46] S. Wagenpfeil, F. Engel, P. M. Kevitt, and M. Hemmje, “Ai-based semantic multimedia indexing and retrieval for social media on smartphones,” *Information*, vol. 12, no. 1, p. 43, 2021.

[47] B. Engelmann and P. Schaer, “Ircologne at trec 2021 news track-relation-based re-ranking for background linking,” *TREC. National Institute of Standards and Technology (NIST)*, 2021.

[48] L. Zhang, H. Joho, S. Fujita, and H.-T. Yu, “Selectively expanding queries and documents for news background linking,” in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 4687–4691.

[49] P. Khloponin and L. Kosseim, “The CLaC system at the TREC 2019 news track,” in *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC)*, 2019.

[50] Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” in *International conference on machine learning*, 2014, pp. 1188–1196.

[51] N. Day, D. Worley, and T. Allison, “OSC at TREC 2020-news track’s background linking task,” in *Proceedings of the Twenty-Ninth Text REtrieval Conference (TREC)*, 2020.

[52] P. Khloponin and L. Kosseim, “Using document embeddings for background linking of news articles,” in *International Conference on Applications of Natural Language to Information Systems*, Springer, 2021, pp. 317–329.

[53] A. A. Deshmukh and U. Sethi, “Ir-bert: Leveraging bert for semantic search in background linking for news articles,” *arXiv preprint arXiv:2007.12603*, 2020.

[54] U. Sethi and A. A. Deshmukh, “Semantic search for background linking in news articles,” in *NIST Special Publication: Proceedings of the 30 th Text REtrieval Conference (TREC 2021), https://trec. nist. gov/pubs/trec30/trec2021. html. NIST*, 2021.

[55] S. Rose, D. Engel, N. Cramer, and W. Cowley, “Automatic keyword extraction from individual documents,” *Text mining: applications and theory*, pp. 1–20, 2010.

[56] L. A. Cabrera-Diego, E. Boros, and A. Doucet, “Elastic embedded background linking for news articles with keywords, entities and events,” *TREC. National Institute of Standards and Technology (NIST)*, 2022.

[57] D. R. Rahul Gautam Mandar Mitra, “TREC 2020 news track background linking task,” in *Proceedings of the Twenty-Ninth Text REtrieval Conference (TREC)*, 2020.

[58] J. Foley, A. Montoly, and M. Pena, “Smith at trec 2019: Learning to rank background articles with poetry categories and keyphrase extraction,” in *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC)*, 2019.

[59] D. Metzler and W. B. Croft, “Linear feature-based models for information retrieval,” *Information Retrieval*, vol. 10, no. 3, pp. 257–274, 2007.

[60] C. Koster and J. Foley, “Middlebury at trec news’ 21 exploring learning to rank model variants,” in *NIST Special Publication: Proceedings of the 30 th Text REtrieval Conference (TREC 2021), https://trec. nist. gov/pubs/trec30/trec2021. html. NIST*, 2021.

[61] J. Qu and Y. Wang, “UNC SILS at TREC 2019 news track,” in *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC)*, 2019.

[62] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[63] A. E. Ak, Ç. Köksal, K. Fayoumi, and R. Yeniterzi, “SU-NLP at TREC news 2020,” in *Proceedings of the Twenty-Ninth Text REtrieval Conference (TREC)*, 2020.

[64] O. Ajnadkar, A. Jaiswal, P. Gourav Sharma, C. Shekhar, and A. K. Soren, “News background linking using document similarity techniques,” in *Computational Intelligence and Machine Learning*, Springer, 2021, pp. 87–95.

[65] M. Essam and T. Elsayed, “Bigir at TREC 2021: Adopting transfer learning for news background linking.,” in *TREC*, 2021.

[66] J. Ravenscroft, A. Clare, and M. Liakata, “HarriGT: Linking news articles to scientific literature,” *ACL 2018*, p. 19, 2018.

[67] J. R. Finkel, T. Grenager, and C. D. Manning, “Incorporating non-local information into information extraction systems by gibbs sampling,” in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, 2005, pp. 363–370.

[68] P. Ferragina and U. Scaiella, “Tagme: On-the-fly annotation of short text fragments (by wikipedia entities),” in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 1625–1628.

[69] M. C. Phan and A. Sun, “Conerel: Collective information extraction in news articles,” in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 1273–1276.

[70] A. Weichselbraun, P. Kuntschik, and A. M. Braşoveanu, “Mining and leveraging background knowledge for improving named entity linking,” in *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, 2018, pp. 1–11.

[71] C. Rudnik, T. Ehrhart, O. Ferret, D. Teyssou, R. Troncy, and X. Tannier, “Searching news articles using an event knowledge graph leveraged by wikidata,” in *Companion Proceedings of The 2019 World Wide Web Conference*, 2019, pp. 1232–1239.

[72] A. Y. Lin, J. Ford, E. Adar, and B. Hecht, “Vizbywiki: Mining data visualizations from the web to enrich news articles,” in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 873–882.

[73] M. Karimi, D. Jannach, and M. Jugovac, “News recommender systems–survey and roads ahead,” *Information Processing & Management*, vol. 54, no. 6, pp. 1203–1227, 2018.

[74] Y. Tian, Y. Yang, X. Ren, *et al.*, “Joint knowledge pruning and recurrent graph convolution for news recommendation,” in *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, 2021, pp. 51–60.

[75] W. Zhang, “Design of news recommendation model based on sub-attention news encoder,” *PeerJ Computer Science*, vol. 9, e1246, 2023.

[76] Y. Lv, T. Moon, P. Kolari, Z. Zheng, X. Wang, and Y. Chang, “Learning to model relatedness for news recommendation,” in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 57–66.

[77] M. S. Desarkar and N. Shinde, “Diversification in news recommendation for privacy concerned users,” in *2014 International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, 2014, pp. 135–141.

[78] C. Chen, T. Lukasiewicz, X. Meng, and Z. Xu, “Location-aware news recommendation using deep localized semantic analysis,” in *International Conference on Database Systems for Advanced Applications*, Springer, 2017, pp. 507–524.

[79] G. de Souza Pereira Moreira, F. Ferreira, and A. M. da Cunha, “News session-based recommendations using deep neural networks,” in *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, 2018, pp. 15–23.

[80] H. Narvala, G. McDonald, and I. Ounis, “Identifying chronological and coherent information threads using 5w1h questions and temporal relationships,” *Information Processing & Management*, vol. 60, no. 3, p. 103 274, 2023.

[81] J. Xu and W. B. Croft, “Improving the effectiveness of information retrieval with local context analysis,” *ACM Transactions on Information Systems (TOIS)*, vol. 18, no. 1, pp. 79–112, 2000.

[82] L. Yang, D. Ji, G. Zhou, Y. Nie, and G. Xiao, “Document re-ranking using cluster validation and label propagation,” in *Proceedings of the 15th ACM international conference on Information and knowledge management*, 2006, pp. 690–697.

[83] Y. Liu, M. Ott, N. Goyal, *et al.*, “Roberta: A robustly optimized bert pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019.

[84] Z. Dai and J. Callan, “Deeper text understanding for ir with contextual neural language modeling,” in *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, 2019, pp. 985–988.

[85] Z. A. Yilmaz, S. Wang, W. Yang, H. Zhang, and J. Lin, “Applying bert to document retrieval with birch,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, 2019, pp. 19–24.

[86] J.-Y. Jiang, C. Xiong, C.-J. Lee, and W. Wang, “Long document ranking with query-directed sparse transformer,” *arXiv preprint arXiv:2010.12683*, 2020.

[87] L. Gao and J. Callan, “Long document re-ranking with modular re-ranker,” in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 2371–2376.

[88] C. Li, A. Yates, S. MacAvaney, B. He, and Y. Sun, “Parade: Passage representation aggregation fordocument reranking,” *ACM Transactions on Information Systems*, vol. 42, no. 2, pp. 1–26, 2023.

[89] L. Yang, M. Zhang, C. Li, M. Bendersky, and M. Najork, “Beyond 512 tokens: Siamese multi-depth transformer-based hierarchical encoder for long-form document matching,” in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 1725–1734.

[90] L. Pang, Y. Lan, and X. Cheng, "Match-ignition: Plugging pagerank into transformer for long-form text matching," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 1396–1405.

[91] A. Jha, V. Rakesh, J. Chandrashekar, A. Samavedhi, and C. K. Reddy, "Supervised contrastive learning for interpretable long-form document matching," *ACM Transactions on Knowledge Discovery from Data*, vol. 17, no. 2, pp. 1–17, 2023.

[92] H. Pottker, "News and its communicative quality: The inverted pyramid—when and why did it appear?" *Journalism Studies*, vol. 4, no. 4, pp. 501–511, 2003.

[93] S. Nishikawa, R. Ri, I. Yamada, Y. Tsuruoka, and I. Echizen, "Ease: Entity-aware contrastive learning of sentence embedding," in *Conference of the North American Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2022, pp. 3870–3885.

[94] Y. Tay, M. Dehghani, S. Abnar, *et al.*, "Long range arena: A benchmark for efficient transformers," *arXiv preprint arXiv:2011.04006*, 2020.

[95] M. Yasunaga, J. Leskovec, and P. Liang, "Linkbert: Pretraining language models with document links," *arXiv preprint arXiv:2203.15827*, 2022.

[96] Y. Jiang, L. Zhang, and W. Wang, "Improved universal sentence embeddings with prompt-based contrastive learning and energy-based learning," in *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022, pp. 3021–3035.

[97] S. E. Robertson, S. Walker, M. Beaulieu, M. Gatford, and A. Payne, "Okapi at TREC-4," *Nist Special Publication Sp*, pp. 73–96, 1996.

[98] Y. Sun, S. Wang, Y. Li, *et al.*, "Ernie 2.0: A continual pre-training framework for language understanding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 8968–8975.

[99] A. Depeursinge and H. Müller, "Fusion techniques for combining textual and visual information retrieval," *ImageCLEF: Experimental Evaluation in Visual Information Retrieval*, pp. 95–114, 2010.

[100] L. Wang, N. Yang, and F. Wei, "Query2doc: Query expansion with large language models," *arXiv preprint arXiv:2303.07678*, 2023.

[101] T. Shen, G. Long, X. Geng, C. Tao, T. Zhou, and D. Jiang, "Large language models are strong zero-shot retriever," *arXiv preprint arXiv:2304.14233*, 2023.

[102] T. Nguyen, M. Rosenberg, X. Song, *et al.*, "Ms marco: A human-generated machine reading comprehension dataset," 2016.

[103] X. Ma, X. Zhang, R. Pradeep, and J. Lin, "Zero-shot listwise document reranking with a large language model," *arXiv preprint arXiv:2305.02156*, 2023.

[104] W. Sun, L. Yan, X. Ma, P. Ren, D. Yin, and Z. Ren, "Is chatgpt good at search? investigating large language models as re-ranking agent," *arXiv preprint arXiv:2304.09542*, 2023.

[105] R. Pradeep, S. Sharifymoghaddam, and J. Lin, “Rankvicuna: Zero-shot listwise document reranking with open-source large language models,” *arXiv preprint arXiv:2309.15088*, 2023.

[106] X. Zhang, S. Hofstätter, P. Lewis, R. Tang, and J. Lin, “Rank-without-gpt: Building gpt-independent listwise rerankers on open-source large language models,” *arXiv preprint arXiv:2312.02969*, 2023.

[107] Z. Qin, R. Jagerman, K. Hui, *et al.*, “Large language models are effective text rankers with pairwise ranking prompting,” *arXiv preprint arXiv:2306.17563*, 2023.

[108] OpenAI, *New models and developer products announced at devday*, https://openai.com/blog/nev models-and-developer-products-announced-at-devday, Accessed: 2023-11-06, 2023.

[109] G. Kumaran and V. R. Carvalho, “Reducing long queries using query quality predictors,” in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 2009, pp. 564–571.

[110] N. Balasubramanian, G. Kumaran, and V. R. Carvalho, “Exploring reductions for long web queries,” in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 2010, pp. 571–578.

[111] Y. Qiu and H.-P. Frei, “Concept based query expansion,” in *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, 1993, pp. 160–169.

[112] C. Florescu and C. Caragea, “Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1105–1115.

[113] A. Bougouin, F. Boudin, and B. Daille, “Topicrank: Graph-based topic ranking for keyphrase extraction,” in *International joint conference on natural language processing (IJCNLP)*, 2013, pp. 543–551.

[114] F. Boudin, “Unsupervised keyphrase extraction with multipartite graphs,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, M. A. Walker, H. Ji, and A. Stent, Eds., Association for Computational Linguistics, 2018, pp. 667–672.

[115] S. Duari and V. Bhatnagar, “sCAKE: Semantic connectivity aware keyword extraction,” *Information Sciences*, vol. 477, pp. 100–117, 2019.

[116] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, “Yake! keyword extraction from single documents using multiple local features,” *Information Sciences*, vol. 509, pp. 257–289, 2020.

[117] T. B. Sarwar, N. M. Noor, and M. S. U. Miah, “Evaluating keyphrase extraction algorithms for finding similar news articles using lexical similarity calculation and semantic relatedness measurement by word embedding,” *PeerJ Computer Science*, vol. 8, e1024, 2022.

[118] J. Piskorski, N. Stefanovitch, G. Jacquet, and A. Podavini, “Exploring linguistically-lightweight keyword extraction techniques for indexing news articles in a multi-lingual set-up,” in *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, 2021, pp. 35–44.

[119] M. Miah, J. Sulaiman, T. B. Sarwar, K. Z. Zamli, and R. Jose, “Study of keyword extraction techniques for electric double-layer capacitor domain using text similarity indexes: An experimental analysis,” *Complexity*, vol. 2021, 2021.

[120] E. Papagiannopoulou and G. Tsoumakas, “A review of keyphrase extraction,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 10, no. 2, e1339, 2020.

[121] G. Rabby, S. Azad, M. Mahmud, K. Z. Zamli, and M. M. Rahman, “Teket: A tree-based unsupervised keyphrase extraction technique,” *Cognitive Computation*, vol. 12, no. 4, pp. 811–833, 2020.

[122] J. Bae and S. Kim, “Identifying and ranking influential spreaders in complex networks by neighborhood coreness,” *Physica A: Statistical Mechanics and its Applications*, vol. 395, pp. 549–559, 2014.

[123] J. Cohen, “Trusses: Cohesive subgraphs for social network analysis,” *National security agency technical report*, vol. 16, no. 3.1, 2008.

[124] L. Page, S. Brin, R. Motwani, and T. Winograd, “The pagerank citation ranking: Bringing order to the web.,” Stanford InfoLab, Technical Report 1999-66, 1999, Previous number = SIDL-WP-1999-0120. [Online]. Available: `http://ilpubs.stanford.edu:8090/422/`.

[125] D. A. Vega-Oliveros, P. S. Gomes, E. E. Milios, and L. Berton, “A multi-centrality index for graph-based keyword extraction,” *Information Processing & Management*, vol. 56, no. 6, p. 102 063, 2019.

[126] V. Batagelj and M. Zaversnik, “An O(m) algorithm for cores decomposition of networks,” *CoRR*, vol. cs.DS/0310049, 2003.

[127] J. Wang and J. Cheng, “Truss decomposition in massive networks,” *VLDB Endowment*, vol. 5, no. 9, pp. 812–823, 2012, ISSN: 2150-8097.

[128] J. Canavilhas, “Web journalism: From the inverted pyramid to the tumbled pyramid,” *Biblioteca on-line de ciências da comunicação*, 2007.

[129] A. Grand, R. Muir, J. Ferenczi, and J. Lin, “From MaxScore to block-max wand: The story of how lucene significantly improved query evaluation performance,” in *European Conference on Information Retrieval*, Springer, 2020, pp. 20–27.

[130] F. Haouari, E. Marwa, and T. Elsayed, “Bigir at TREC 2020: Simple but deep retrieval of passages and documents.,” in *TREC*, 2020.