Utilizing Passage-Based Evidences in Information Retrieval Tasks

Ghulam sarwar

Supervisor: Dr. Colm O'Riordan

School of Computer Science
University of Galway

This dissertation is submitted for the degree of
Doctor of Philosophy

University of Galway January 2023

I would like to dedicate this thesis to two women and two men of my life. The women are my wife Shaima Niaz and my loving mother Farah Azhar. And the men includes my late father Azhar Rasheed and my uncle Nasir Ameer who was always there for me as my dad. Thank you for being there every step of the way. Your consistent encouragement and believe in me really pushed me beyond to achieve my goals.

Declaration

I hereby declare that the contents of this dissertation are original and my own work, except where specifically stated otherwise in the text. This work in this thesis is neither submitted in whole nor in part for consideration of any other degree or professional qualification.

Ghulam sarwar
January 2023

Acknowledgements

Throughout the writing of this dissertation and in the last couple of years, I have received immense support and a great deal of encouragement and assistance.

At first, I would like to express my gratitude to my supervisor Dr. Colm O'Riordan for his constant guidance and support. His openness and enthusiasm towards research has inspired me a lot and made me a much better researcher and a critical thinker. He has always given me the freedom to explore my ideas and pushed me to strive for better solutions; for that, I am truly grateful. Thank you for being a friend and a great mentor. I would also like to thank Dr. John Newell for his support during the early days of my PhD and for guiding me throughout my time as a research scientist at Orreco.

I am very grateful to Irish Research Council for giving me the opportunity and funding my research through their Employment-Based Program. It shaped my thinking and gave me exposure to both academia and industry standards. Special thanks to my GRC committee, Dr. Seamus and Dr. Alberto for their support and encouragement. I would also like to thank Tina Earls, Mary Hardiman, and Peter O'kane from the IT department for always being extremely helpful and supportive.

I would also like to give my sincere thanks to all my lab mates in Room 311; Dr. Marcos, Dr. Maud, Dr. Jane, Dr. Aidan, Dr. Stephen, and Afnan. Also, i am very grateful to my friends Dr. Shariq, Dr. Hamda, Amir, Daniyal, Tashfeen, Hashim and Dr. Haroon for always being there for me at the time of need.

Lastly and most importantly, I would like to thank my family. My wife, Shaima for being an amazing partner and for constantly encouraging me and pushing me to go beyond my limitations. She has always given me hope at times whenever I feel down. I am eternally grateful to you for everything. My mother, Farah Azhar who has been a source of my strength all my life. I cannot thank you enough for raising me and my brother Faheem as a single mother and always believing in us. Thank you for letting me always follow my dreams even if costs you to stay far from me. I will always be indebted to you for your constant prayers, your wisdom, and for all the opportunities you have provided me throughout my life. I also want to thank my uncle, Mamo Nasir and Aunti Papi for treating me like their own son. Uncle Nasir, I cannot thank you enough for being a source of inspiration, constant encouragement, and a fatherly figure in my life. Finally, I want to bow down to ALLAH almighty SWT for giving me the strength, courage, and blessings me with good health. Verily, it is He who grants us according to his mercy and favour what we do not deserve.

Abstract

In the age of Information overload accessing information is a few clicks away from us. This wealth of information at hand can certainly have its advantages. However, for a search engine, identifying the relevant information from that Big Data is still a challenging task. Particularly, finding pertinent information from lengthy documents is tricky due to the natural language nature of searched queries and the topical diversity of the documents. Rather than considering the document as a whole, one viable method is to measure the pertinence of a query to the concise units (passages) in the given document and utilize that measuring process for evaluating the query-document relevance. This thesis aims to utilize these smaller units of a document known as passages in different information retrieval tasks.

A passage is defined as a sequence of sentences or words that start and end at any place within a given document. Passage retrieval deals with identifying and retrieving small but explanatory portions of a document that answers a user's query. In this thesis, we first present a novel approach to improving the document ranking by using different passage-based evidence. We evaluated our approach with the existing passage retrieval methods and more in-depth analysis was undertaken into the effect of varying specific. We have also explored the notion of query difficulty to understand whether the best performing passage-based approach helps to improve, or not, the performance of certain queries.

Secondly, we presented a novel graph approach that utilizes the similarity of passages within their parent document to form a cohesion structure. We discussed that the relevant documents tend to be more cohesive than the non-relevant documents. Furthermore, we also re-ranked the documents by applying the cohesion score with a document similarity score to inspect its impact on the system's performance.

Moreover, we carried out experiments by using different sliding windows around words in each passage to determine the context and semantic relatedness. We then compared the state of the art pseudo relevance feedback (PRF) technique with our proposed passage-based sliding window approach for query expansion. The usage of top-ranked passages for query expansion was motivated due to the reason that relevant passages for query expansion would remove elements of noise found in a text document that contains a number of topics. We extended our approach by including a popular word embedding (WE) approach i.e the word2vec and have demonstrated that the passage-based *PRF* and *WE* approach outperforms their document-based equivalent.

Lastly, we utilize the passage answer-set for each query as a graph and applied different graph-based measures to identify a correlation between the relevance of a document and those calculated graph measures. Our approach was inspired by the cluster hypothesis which states that similar entities are more likely to be closer to each other. We also discussed an application of our answer-set graph approach for the Query Performance Prediction tasks and a future avenue to apply it for the topic visualization. We have shown that our passage-based graph features outperforms the existing state of the art QPP approaches and generate a positive correlation in determining the easy and the hard queries.

List of Publications

- Sarwar, G., O'Riordan, C., and Newell, J. (2017). Passage level evidence for effective document level retrieval.In Proceedings of the 9th International Joint Confer-ence on Knowledge Discovery, Knowledge Engineer-ing and Knowledge Management, pages 83–90.

- Sarwar, G. and Bradshaw, S. (2018). Investigating the Use of Semantic Relatedness at Document and Passage Level for Query Augmentation. In Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - KDIR, ISBN 978-989-758-330-8; ISSN 2184-3228, pages 237-244

- Sarwar G., O'Riordan C., Newell J. (2019) Investigation of Passage Based Ranking Models to Improve Document Retrieval. In: Fred A. et al. (eds) Knowledge Discovery, Knowledge Engineering and Knowledge Management. IC3K 2017. Communications in Computer and Information Science, vol 976. Springer

- Sarwar, G. and O'Riordan, C. (2021). A Graph-based Approach at Passage Level to Investigate the Cohesiveness of Documents. In Proceedings of the 10th International Conference on Data Science, Technology and Applications - DATA, ISBN 978-989-758-521-0, pages 115-123

- Sarwar, G. and O'Riordan, C. (2021). Passage Based Answer-Set Graph Approach for Query Performance Prediction. In ADCS '21: Australasian Document Computing Symposium, December 9, 2021, Virtual Event, Australia, ISBN 978-1-4503-9599-1/21/12

Publications under Review

- Sarwar, G. and O'Riordan, C. (2022) Finding Document Cohesion through Passage Graph Approach to Re-Rank Search Results. Springer Nature Computer Science Journal

Table of contents

List of figures

adjustbox

List of tables

Abbreviations

AC	Answer-set Cohesion
BIM	Binary Independence Model
BM25	Okapi BM25
BM	Boolean Model
BOW	Bag of Words
BRF	Blind Relevance Feedback
CBOW	Contineous Bag of Word
C	Cohesion
DL	Document Level
EC	Edge Count
e	Edge of a Graph
HAL	Hyperspace Analogue to Language
HITS	Hyperlink-Induced Topic Search
Idf	Inverse Document Frequency
IRF	Incremental Relevance Feedback
IR	Information Retrieval
JC	Jaccard Coefficient
KL Divergence	Kullback Leibler Divergence

LDA	Latent Dirichlet Allocation
LM	Language Model
LTR	Learn To Rank
LVSM	Lucene Vector Space Model
MAP@K	Mean Average Precision at Top k Documents
MAP	Mean Average Precision
ML	Machine Learning
MMR	Maximum Marginal Relevance
MPL	Max Passage Level
MRR	Mean Reciprocal Rank
MST	Minimul Spanning Tree
NN	Neural Network
NQC	Normalized Query Commitment
NR	Non-Relevant Documents
P@K	Precision at Top K Documents
PL	Passage Level
POS	Part of Speech
PRF	Pseudo Relevance Feedback
psg	Passage
QA	Question and Answering
QE	Question Expansion
QL	Query Liklihood
QPP	Query Performance Prediction
R	Relevant Documents

sim	Textual Similarity
SOPL	Sum of Passage Level
TF	Term Frequency
TREC	Text Retrieval Conference
VSM	Vector Space Model
v	Vertic/node of a Graph
WE	Word Embedding
WEG	Weighed Expansion Gain
WIG	Weighed Information Gain
wt	Weight of an Edge in a Graph
|E|	Number of Edges in a Graph

Chapter 1

Introduction

1.1 Motivation

One of the fundamental objectives in the field of artificial intelligence is to provide aid in the decision making process. More importantly, obtaining the most relevant information to the task at hand is what drives scientists to delve deeper in the AI domain. We now live in the era of data where every day the amount of information available online in digital form is increasing exponentially. The web has become the primary source of this abundance of knowledge and information and the ease at which all that unstructured data (in form of text) is put on to the Web is one of the reasons for this rapid growth of data on Web. Search engines are the tools that are used primarily in locating the required information. The search engines require the mechanisms to find the information and to estimate whether that information is relevant to the user's needs.

Information Retrieval Information Retrieval (IR) deals with the organization, representation, and retrieval of information from a large set of text documents. Given the huge growth of unstructured data and information, there is an essential need for better techniques to access this information. The retrieval of relevant information from large collections is a difficult problem, given that search queries are often presented in natural language. This introduces many problems such as ambiguity caused by synonyms, abbreviations, and translation errors where the user expresses their information need in a language different from that of the document collection. Several models have been shown to be very effective in ranking documents in terms of their relevance to a user's query. The user formulates the query by expressing their information need in natural language. Approaches include different mathematical frameworks (vector space model [187], probabilistic models) to represent documents and queries and to formulate a comparison approach. The BM25 [171] weighting scheme derived within a probabilistic framework is a well-known effective one in estimating the relevance of a document to a query. Similarly, the query likelihood language model [220] has also been used in the past where each document is represented as a probability distribution over the vocabulary, or a language model.

In IR, the traditional approaches consider the document as a single entity. However, some people choose to split the document into separate passages, mainly because, sometimes a highly relevant passage may exist in a large document, which when compared to the whole document, is no longer as relevant to the query. For example, consider a news online forum that obtain numerous news related web articles and perhaps a small portion of one article is relevant to a certain query; in this case, if we consider the whole article as one single document, that small portion (passage) would possibly have a very little influence on the overall relevance against the user query. In addition to that, long documents often contain multiple topics discussed in it. Similarly, if the returned document is too long, it can be difficult for the users to find the appropriate passage of the document that reflects the most relevant section. In other words, returning a large relevant document, while useful, still, puts an onus on the user to find the relevant passages.

There are many situations in which the passage level retrieval comes in handy. However, finding the right passage can be difficult, and many problems can occur in the process of indexing or retrieving the right passage from the system. For example, sometimes, it is hard to identify the boundary of passages while trying to figure out what is the best way to divide the passages in a document. Passages can be divided by using separators (sentences, paragraphs, indentations, etc.). However, it requires consistency from writers since the structure of the document can be disorganized and sometimes paragraphs can be ranged from one line to many lines depending on the writer and the test collections. One can also use the number of words approach based on the window size [32], but you may end up dividing the same piece of text into different paragraphs and lose the context. Similarly, if a passage is indexed as a document, the number of documents stored and indexed would increase significantly and as a result, it will affect the speed and cost of retrieval [170]. Moreover, as mentioned previously, the queries are expressed in natural language and this introduces confusion to the system. This problem becomes more common while retrieving passages because the amount of text is less and that makes it harder for the system to return the best possible passage.
One prominent challenge in ad-hoc document retrieval is to utilise the evidence retrieved from the part of the documents (passages) to identify if that document, in fact, is relevant against the given query. Numerous approaches were adopted to utilise the passage evidence within the document. Considering the top most relevant passages within a document is one such example of utilising the passage evidence to improve the ranking. However, the queries are expressed in natural language and this introduces confusion to the system. This problem

becomes more common while retrieving passages because the amount of text is less and that makes it harder for the system to return the best possible passage.

Similarly, it is common to lose the context of information around the returned passages. It may happen that while dividing the passage with separators, the returned results do not capture the context around that passage [32]. In some cases, the returned answer have all the keywords in it; however, it still does not satisfy the query. Also, there is some knowledge or evidence available in the document from which we try to extract the passage; this information can be augmented with the other external sources of evidence to improve the results. A document can consist of many topics or subtopics and these can be divided into several passages as well. Normal passage level retrieval does not consider the structural similarities of the documents and their passages with each other. This thesis focuses on the utilization of passage evidence and its application to a few ad-hoc retrieval tasks, including document re-ranking, document relevance classification, query expansion, and query performance prediction.

1.2 Research Questions

Passage level retrieval has been studied in a number of tasks. Scientists have used passage retrieval for ad-hoc information retrieval problem [32, 100] and also explored its application in Question and Answering(QA) tasks [235, 72], Query Expansion (QE) [80, 144, 135] and classification [235]. Ad-hoc information retrieval is the task of locating information sources that are relevant to a user's query expressed in natural language [9]. For ad-hoc retrieval, approaches such as boosting the document higher in ranking if the relevant passage(s) exists in it have been considered in different ways. To provide that boost, researchers have mainly utilised the term weighting based similarity score between the passage and the query to estimate the similarity between the query and the document. However, it is still not clear what is the best possible way to define that similarity score. An open question here is that are there any other useful measures that can be used as passage evidence for ad-hoc retrieval? (Q1)

Oftentimes the answers retrieved with the passage retrieval are generic in kind and were covering the abstract topic rather than narrowing it down to particular information that was required [85]. Some researchers have used semantic approaches to utilize the context by defining different similarity-based relationships around the passage to improve the ranking process. Since the amount of text in passages is small, it is common to lose the context and the relationships between passages. It is neither fully explored nor thoroughly discussed that the most suitable similarity measures to identify and represent those contextual relationships within a particular chunk of documents, i.e. passages. The cluster hypothesis states that relevant documents tend to be more similar to each other than to non-relevant documents [213]. An open question prompted here is how do relevant documents compare to non-relevant documents when passage evidence is applied? (Q2)

In addition to that, recent research has attempted to explore the use of passage retrieval in Query expansion by utilizing semantic approaches [85] as well as machine learning base solutions [72, 142, 121]. However, it has been shown that these semantic approaches for passages are a good measure for query expansion. However, to the best of our knowledge, previous research efforts have not considered combining the semantic or ML approaches with the fundamental passage level evidence used separately for ad-hoc retrieval. That raises an open question that can combining such passage evidence with different word embedding approaches (such as word2vec, term-occurrences matrix, HAL) produce better knowledge for query expansion and improve the performance of an IR system? (Q3)

Lastly, we studied the use of passage retrieval for Query Performance Prediction(QPP). The QPP task has been studied extensively, with the focus mainly divided between the post or the pre-retrieval prediction [34]. Similarly, research has been done to use the passage information from the documents to predict the query performance [176]. However, utilizing the relationship among the returned passages for a given query has not been studied as a measure for query performance prediction. This introduces an open question that given the passages returned (answer-set) for the respective query (q), could combining a relationship between the passage answer-set be used as an effective measure for the Query Performance Prediction task? (Q4)

1.3 Hypotheses

This thesis examines the use of passage level knowledge either separately or in conjunction with different semantic approaches to augment the passage knowledge to improve ad-hoc retrieval. Given the aforementioned research questions, we have formulated and tested the following hypothesis:

$[H1]$: The rank of a passage can be a useful measure of its similarity to the query. When used separately or in conjunction with the document score, it can give better results than other passage or document level similarity approaches.

$[H2]$: A graph can be used to represent the connection between passages within the retrieved documents. The connection between passages within the same document from a relevant set is stronger than the document from the non-relevant set. Hence, we can utilize the graph measure like cohesion to re-rank search results and improve the performance of the system.

[$H3$]: For the query expansion task, capturing the semantic relatedness with passage evidence can produce better results than the standard document level retrieval.

[$H4$]: Considering a graph created from items in the answer-set (returned passages for a given query), where each item/node represents a passage, certrain graph's features can be used as an effective measure for Query Performance Prediction (QPP) task.

1.4 Research Methodology

Research methodology is defined as a scientific process of studying a phenomenon or event and a systematic way to solve the research problem [82]. Through research methodology, the researcher examines numerous procedures used to investigate the research issue as well as the rationale behind the techniques employed in the study. In this thesis, the quantitative research method is applied for data collection and analysis. A quantitative study is defined as a study that includes a systematic examination for gathering numerical data about a subject. In Quantitative Research (*QR*), data that can be expressed numerically and can be measured, such as statistical findings, demographic information, and financial results are gathered and analyzed [69]. It puts great emphasis on the causal relationship between variables and entails the estimation of the processes involved. Furthermore, a quantitative method is used in order to get more quantifiable and measurable results hence, to answer research question through percentage or frequency of occurrence. We used Experimental and Correlation Design of *QR* to analyse the outcome of experimental results. Statistical analysis (Student t-test) was also performed to identify the significance of proposed methods against the baseline approaches.

1.5 Research Ethics

In this thesis, we employed the publicly available test collections to test our hypothesis and research questions. No amendments or changes were performed in the employed test collections. Although the WebAp collection is derived from the non-public 2004 TREC Terabyte Track Gov2 collection and its corresponding description-length queries. This collection is publicly available by the Center for Intelligent Information Retrieval group at University of Massachusetts Amherst. Furthermore, the Cranfield and the Ohsumed test collection are publicly available by the Information Retrieval Group at University of Glasgow. The thesis does not contain any studies with human participants performed by the authors.

1.6 Contributions

We developed a system that implements the state of the art passage retrieval approaches and compares them to document level retrieval. We then extend this system to incorporate the graph-based structure of passages and integrate our passage-based retrieval system with few baseline algorithms for QE and QPP (like Rocchio, *word2vec*, Hal, WIG, NQC, Standard Deviation). Following contributions are made to reflect the effectiveness of passage-based evidences in Information Retrieval tasks.

$[C1]$: We addressed question 1 by proposing two new passage similarity measures, which considered the rank of passage instead of the similarity score. We discovered that our inverse rank (lower the rank, higher the relevance) approach at passage level performed better than the previously used passage-based similarity functions.

$[C2]$: To answer question 2, we presented a graph-based approach at passage level and defined the cohesion of a document based on how well different passages from the same document are connected. We used that as a measure to differentiate the relevant and non-relevant documents. We defined this graph-based approach flexibly to utilise different proximity between the edge weight and change the graph structure based on that. In this way, we can define the contextual information around the passage using several approaches and compare them.

$[C3]$: We tackled question 3 by combining the semantic approaches at the passage level and using them to augment the query. We introduced a mechanism to include more terms in the query incrementally and observe the system's performance change. We have seen that passage evidence is useful for query expansion and an increase in performance (for top results i.e. P@5, P@10) was shown compared to the counterpart i.e. using documents for QE.

$[C4]$: For question 3, we extended our approach to utilise a machine learning approach to map words on different embedding (word2vec) and trained the models at document level as well as at passage level for the QE task. We also compared the occurrences of common expansion terms that are extracted from a standard Pseudo Relevance Feedback approach like Rocchio Algorithm and Word2vec model. The intent here was to check how often both approaches provide similar query expansion words.

$[C5]$: To answer question 4, we introduce a new approach that utilises the different features of a graph generated with the answer-set of passage. Then we use these features as a measure for the Query Performance Prediction (QPP) task. We have shown that our passage-based graph features exhibit a positive correlation for all test collections. Furthermore, against the Cranfield and Ohsumed collection, it outperforms all the traditional QPP approaches.

1.7 Thesis Flow

In this dissertation, the motivation, research questions, hypotheses and contributions are introduced in this chapter. Chapter 2 addresses the background in IR and describes the basic language models and the passage level retrieval task. Chapter 3 presents state of the art related to passage level retrieval and its applications in ad-hoc retrieval. We will also discuss the general query expansion approaches and extend the query expansion as well as query performance prediction related work motivated by the passage level retrieval in more detail. Furthermore, we will conclude this chapter by highlighting the current research on machine learning in information retrieval tasks for ad-hoc retrieval that specifically takes passage evidence into account.

Chapter 4 presents our work on passage-based evidences with document level retrieval. We will present a detailed explanation of different similarity functions used to evaluate the system and discuss how we explored the notion of query difficulty to understand whether the best performing passage-based approach improves, or not, the performance of specific queries.

Chapter 5 and chapter 6 discuss how we applied passage-based evidence in finding the contextual similarities between passages using a graph-based approach and for tasks like query expansion. Chapter 5 describes the details of our graph-based approach at the passage level. We will first present our hypothesis regarding the difference between the relevant and non-relevant set via passage-based cohesion score. We will then discuss how we utilised the passage-based cohesion score in the information retrieval task and its implications. Finally, we explain our graph approach's extension by introducing a bipartite graph using the query-passage similarity. We used this new graph to re-define our cohesion approach and compared the results to our original graph approach.

Chapter 6 presents two different scenarios in which we used query expansion approaches and compared them to the baseline, i.e. Rocchio Algorithm [175]. In the first scenario, we present a graph model that we introduced that considers the term counts on the proceeding and preceding terms with the passage level approach. We compare the results with Hyperspace Analogue to Language (HAL) approach proposed by Lund [136]. In the second scenario, we use the word embeddings approach like *word2vec* [142] to train based on the passage evidence and use the newly generated models for the QE task.

Chapter 7 describes a newly generated graph approach derived from the answer-set of passage. We used the features of that passage-graph as a measure for the Query Performance Prediction (QPP) task. In this thesis, we use the term 'answer-set' to represent the top ranked k passages as a retrieved list for a given query q where k is a parameter that can be changed as per the need.

Chapter 8 will conclude this thesis by outlining the main contributions and findings of this work and discussing the future work.

Chapter 2

Background

In this chapter, we start by giving a review of major IR models that we employed for passage level retrieval. To understand the text, it is important to convert it into a representation that is easy to manipulate and is in a structured format. We will start by giving a general overview of the vector space model. This is followed by an overview of statistical language modeling as it has been used extensively in areas like text categorization, information retrieval, and text parsing. We will then further explain concepts like topic modeling, entity recognition, cosine similarity. which we use as a similarity measure within our proposed model. Finally, present an overview of query augmentation, a graph representation of text, and word embedding approaches that we employed as an application for passage level retrieval to improve the document ranking

2.1 Information Retrieval Models

The main purpose of an Information Retrieval (IR) system is to provide users with the correct documents as per their needs. The most common approach to represent a document is the Bag-of-Words model (BOW). It does not consider any term order, context, or grammar and view each term as independently in the document. The Boolean Model (BM) is referred to as the first and most adopted model at the start. The retrieval is based on whether the document contains the terms in the query or not. After BM, statistical models like Vector Space Model (VSM) [187] or Probabilistic Models like BM25 [171] are the major examples of statistical retrieval approach.

2.1.1 Boolean Model

The Boolean model is considered as a classical Information Retrieval model that employs the fundamental concepts of Boolean logic and classical set theory. A document is deemed relevant if a query has term(s) that are found in the same document. As per George Boole, in Boolean Algebra [27], there are three basic operators *AND*, *NOT*, and *OR* which can be denoted as the logical product, logical difference, and logical sum [41]. For instance, a query *cricket* AND *entertainment* will return the set of documents that contains both the terms *cricket* AND *entertainment*. Combining the query terms with *OR* will produce the set of documents with any of the specified terms. There is no consideration of partial match and therefore, it does not provide a ranking of retrieved documents. So a document is either relevant or not relevant in relation to the given query. To overcome this problem, an extension of this model called 'Extended Boolean Model'[185] was introduced to take into account the weighting of terms, which produced a set of ranked documents that could possibly be relevant to a given query.

2.1.2 Vector Space Model

The vector space model represents the documents and queries as vectors in a multidimensional space, whose dimensions are the terms used to build an index to represent the documents [187]. Unlike the Boolean Model where the model checks whether the query term exists in the document, in VSM each term corresponds to a different dimension and a certain weight is assigned to them. These weights on each term are used to identify the significance of a document to the query. Each document d_j and query q are represented vectors:

$$\vec{d_j} = (w_{1j}, w_{2j}, \dots, w_{nj}) \tag{2.1}$$

$$\vec{q} = (w_{1q}, w_{2q}, \dots, w_{tq}) \tag{2.2}$$

where n is the total number of terms in the document. As the terms are represented as vectors, it is possible to measure the closeness between vectors. This closeness can be represented as measuring the similarity score between the documents. The cosine of the angle between the vectors is used as it is a normalized measure and has no effect of the length of a document. The similarity S_{ij} between a document d_i and d_j can be represented as follows:

$$\vec{S_{ij}} = \frac{\vec{d_i}.\vec{d_j}}{|\vec{d_i}| \times |\vec{d_j}|} \tag{2.3}$$

$$= \frac{\sum_{l=1}^{n} w_{il} \vec{\times} \vec{w}_{jl}}{\sqrt{\sum_{l=1}^{n} w_{il}^2} + \sqrt{\sum_{l=1}^{n} w_{jl}^2}} \tag{2.4}$$

where the numerator of the equation is the dot product of the vectors and $|\vec{d}_j|$ and $|\vec{d}_i|$ represents the norms of these vectors. As all vectors under consideration by this model are element-wise non-negative, a cosine value of zero means that the query and document vector are orthogonal and have no match. Figure 2.1 visualizes a document and a query vector in the 3-dimensional space which is spanned by the terms *social*, *political*, and *economics* [95].

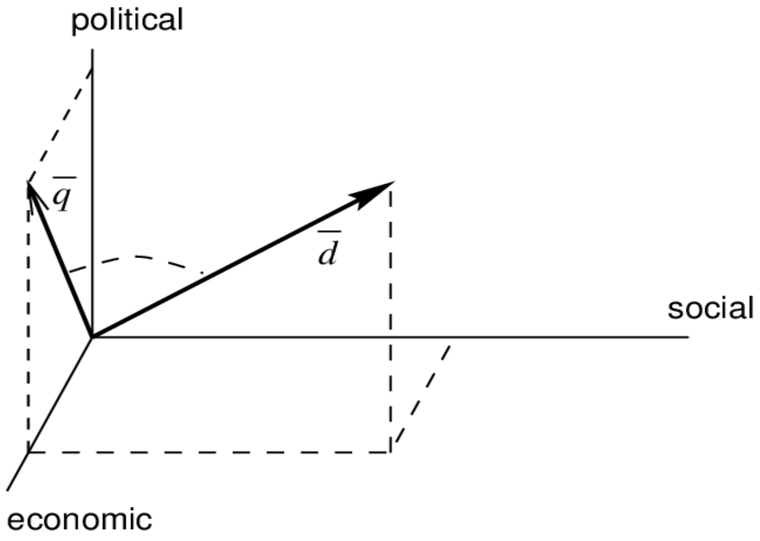


Fig. 2.1 Document and Query portrayal in the VSM

For VSM, the term specific weights reflect the combination of local and global parameters. The method named tf-idf (explained in section 2.2.1) is used where tf represents $termfrequency$ i.e. how many times a term is being found in the document and idf represents $inversedocumentfrequency$ which is a global measure that is used to see how much information words provide as in is it a rare word or a commonly used across the document.

Probabilistic Model

The probability of occurrence of an event E, denoted by $P(E)$ is formalized through the concept of experimentation, which is the process by which the observation is made. In IR, probabilistic model considers that the indexing terms are independent, which means that each term can occur with the same probability with our with the presence of other terms [174]. The first classic probabilistic model is the Binary Independence Model (BIM) [173]. BIM acts as a classifier for relevant (*R*) and non-relevant (*NR*) documents provided the term independence is assumed. Using *Bayes* rules, a document is taken as a relevant document if

$$P(D|R)P(R) > P(D|NR)P(NR)$$

Given a query q and document D, we can rank documents by their odds of relevance. The documents are ordered based on the likelihood ratio of a document, which can be denoted as:

$$P(R|q,D) = \frac{P(D|q,R)}{P(D|q,NR)} \tag{2.5}$$

As each document is composed of set of words i.e. $D = \{w_1, w_2, ..., w_n\}$, BIM considers each document as a vector of binary feature where a word $w_i = 0$ if the word does not exist in the document D and vise versa if the word exists in the document $(w_i = 1)$. The ranking of the document is calculated by:

$$P(R|q,D) = \Pi\, P(w|q,R) \tag{2.6}$$

$$P(w|q,R) = \frac{P(w|q,R)}{P(w|q,NR)} \tag{2.7}$$

The resulting quantity used for ranking in this model is called Retrieval Status Value (RSV) which is equivalent to:

$$RSV = \sum_{w:d_w=1} \log \frac{p_w(1-s_w)}{s_w(1-p_w)} \tag{2.8}$$

where p_w is the probability that the word w_i occurs in the R document and s_w occurs in the NR document.

2.2 Term Weighting

The weighting of an indexing term is the association of numerical values with these terms which represent its power of discrimination of each document in the collection [183]. The term weights depict a significant role in the performance of an IR system. Different IR models adopted different term-weightings schemes. Here we will discuss the widely used schemes like tf-idf and $BM25$.

2.2.1 Tf-idf

The relative frequency of a term in a document is representative of the power of the term in the document. Similarly, the absolute frequency of a term in the collection is the characteristic of the term's power of discrimination for documents [183]. Therefore it is important to take into account the relative frequency and the absolute frequency of a term during its weighting. There are several ways to associate a weight to a term. One can represent the weight for a term in a document with 1 or 0 that expresses the presence and absence of a term in the document. Salton et al. [183] introduced the notion of Term Frequency(tf) and Inverse Document Frequency(idf). Tf is the frequency of the term in the document i.e. the number of occurrences of a term t_i in the document d_j that has a total n_j terms. The idf is the inverse absolute frequency. It is a factor that varies inversely proportional to the number n of documents where a term appears in a collection of N documents. It can be denoted as follows:

$$tf_i j = \frac{n_{ij}}{\sum_{k=1}^{nj} n_{kj}} \tag{2.9}$$

$$idf = \log\left(\frac{N}{n}\right) \tag{2.10}$$

The combination of tf and idf is used for term weighting in VSM as follows:

$$tf - idf = tf \times idf \tag{2.11}$$

2.2.2 BM25

In the probabilistic model, documents and queries are represented as a probability distribution. BM25 is a commonly used probabilistic retrieval framework based weighting scheme introduced by Robertson and Walker [171]. It is a ranking function used by many search engines to measure the relevance of documents to a given query. Given a query Q, containing keywords $q_1, \ldots, q_n$ the BM25 score of a document D is:

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i,D)+k_1\cdot(1-b+b\cdot\frac{|D|}{\text{avgdl}})}$$

where $f(q_i,D)$ is term frequency in document D, $|D|$ is the number of words in D, $avgdl$ is the average document length in the text collection from which documents are drawn, N is the number of documents in the collection, k_1 and b are model parameters, and IDF is the weight of the query term q_i. The IDF is computed as follows:

$$IDF(q_i) = log\frac{N - df_w + 0.5}{df_w + 0.5} \tag{2.12}$$

where df_w is the number of documents in which term w is present. Different interpretation of IDF has been used by the researcher to improve the ranking function.

2.3 Evaluation Measures

To measure the performance of an Information Retrieval system, evaluation measures are required. It can be used to measure and analyze at what certain degree the system achieves the user's satisfaction. A traditional way to evaluate the system is to compare the metric that we are testing against the ground-truth which is annotated by a human. Evaluation measures that are used throughout the experiments and explanations of this thesis are defined as follows:

2.3.1 Precision@K

To define a precision(p) at K, at first, it is important to understand what is precision as a metric when we evaluate an IR system. Take D_r as a set of relevant documents and D_s are the documents returned by an IR system for a given query. Precision is defined as follows:

$$Precision = \frac{|D_r \cap D_s|}{|D_s|} \tag{2.13}$$

where the numerator denotes the total common documents in D_r and D_s. Precision p consider all the retrieved documents which can be cut off at a given rank as well. So to measure the precision of only the topmost documents k, we use the measure called $p@k$. It is defined as follows:

$$p@k = \frac{|D_r|}{|topKresults|} \tag{2.14}$$

2.3.2 Mean Average Precision(MAP)

Precision is a single-value metric and does not provide any information regarding the ranking of a document in the returned document list. Average Precision(AP) is used to take into account the order in which the documents are returned. Take D_{rk} as a set of relevant documents for a query q_k and for a position i we measure the precision value for each document, which can be denoted as $P(D_{rk}i)$. The Average precision for a query k can be presented as follows:

$$AP_k = \frac{\sum_{i=1}^{|D_{rk}|} P(D_{rk}i)}{|D_{rk}|} \tag{2.15}$$

Now if we calculate the Average Precision for a set of queries Q, it will give us the overall performance of the system across all queries. This metric is known as Mean Average Precision(MAP). It is defined as follows:

$$MAP = \frac{\sum_{k=1}^{|Q|} AP_k}{|Q|} \tag{2.16}$$

2.3.3 R-Precision

R-Precision is another useful metric to measure the performance of an IR system. To measure it, one should know all the relevant documents to a given query. Assume that R is the total number of relevant documents for a given test collection. For a given query q, we retrieve top R documents and count the number of relevant documents returned r for a query q. The R-Precision of a query q can be calculated as:

$$R\text{-}Precision = \frac{r}{R} \tag{2.17}$$

2.3.4 Mean Reciprocal Rank (MRR)

Reciprocal Rank (RR) is the multiplicative inverse of the first relevant document's rank found in the answer-set. In comparison, Mean Reciprocal Rank (MRR) is the mean of RR across multiple results. It is used as a measure to capture the significance of the answer-set (similar to AP). For a sample of Queries Q, MRR can be calculated as follows:

$$MRR = \frac{1}{|Q|} \sum_{k=1}^{|Q|} \frac{1}{rank_k} \tag{2.18}$$

2.3.5 Kullback–Leibler(KL) divergence

KL divergence is a way to measure the difference between two probability distributions. If the two distributions are identical, the KL divergence of those probability distributions will be 0. In information theory, the aim is to quantify the useful information in the data-set. Entropy is a useful measure that reflects the uncertainty associated with a random variable or a specific outcome of an event. Given l_q and l_d are unigram language models (probability distribution over all vocabulary terms w) generated from query q and document d. We can compare the query and document language models to drive the ranking of the documents.

KL divergence is used as a similarity measure between the q and d. KL divergence between two language models is calculated as follows:

$$D_{KL}(l_q, l_d) = \sum_w P(w|l_q) \cdot (\log P(w|l_q) - \log P(w|l_d)) \tag{2.19}$$

$$= -\sum_w P(w|l_q) \cdot \log \frac{P(w|l_q)}{P(w|l_d)} \tag{2.20}$$

The higher KL values correspond to decreased similarity. Therefore, the negative KL divergence is used.

2.3.6 Jaccard Coefficient

Jaccard similarity Coefficient is a statictic measure that is used to identify the diversity and the similarity between two sets. In Information Retrieval, Jaccard Coefficient is used to measure the similarity between the retrieved documents. It is calculated by dividing the size of intersection with size of union of two sets. Consider A and B as two documents with different set of terms. The Jaccard Coefficient between A and B is calculated as follows

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{2.21}$$

For example if the document A has 3 words and the document B has 4 words, with only 2 words in common in both sets, the Jaccard index ($J(A,B)$) will be $2/7 = 0.285$.

2.4 Language Models

The VSM in general works well. However, as the model is based on the standard bag of words (BOW) approach, it discards the word order resulting in a loss of context and semantic meaning of words. The term (statistical) 'language model' (LM) refers to a probability distribution over the sequence of terms taken from an established vocabulary [179]. It is used to provide the distinction between the words and phrases that seems familiar. Language models provide a probability of how likely from a given vocabulary V, a word could appear in a sentence when surrounded with some other sequence of words. A model named Query-Likelihood (QL) was introduced by Ponte and Croft [163] that estimates the probability of a query is established by considering the likelihood distribution from a fixed vocabulary of a document. To rank the documents Bayes's is applied and for a given query q and document D, the probability is calculated as follows:

$$P(D|q) = \frac{p(q|D)p(D)}{p(q)} \tag{2.22}$$

2.4.1 N-Gram Model

A commonly used language model is the n-gram model. It assumes that for the ith position of text, a term x_i is dependant on the n-1 preceding terms. A $bi-gram$ or a $tri-gram$ model denote an n-gram model with the value of n=2 or n=3. So for a $bi-gram$ model, the probability of a term to be at a specified position in the sequence of words will depend on the 2 preceding words. A bag of words [87] assumption can be taken as an n-gram(uni-gram) model which follows the notion of term independence in the text. In an n-gram model, the probability $P(s_1,\dots,s_m)$ of observing the sentence with sequence $s_1,\dots,s_m$ is approximated as $\prod_{i=1}^{m} P(s_i|s_{i-(n-1)},\dots,s_{i-1})$. Following is the formula used to calculate the conditional probabilities of n-gram model. This conditional probability is also denoted as Maximum Likelihood Estimation(MLE) technique.

$$P(s_i|s_{i-(n-1)},\dots,s_{i-1}) = \frac{tf(s_{i-(n-1)},\dots,s_{i-1},s_i)}{tf(s_{i-(n-1)},\dots,s_{i-1})} \tag{2.23}$$

where tf denotes the term frequency of a sequence. An illustration of n-gram models is shown in the figure 2.2 below. As shown in the picture, the likelihood of word sequences like ‘this is a’is greater then the sequence ‘is a this’.

It is to note that sometimes, the documents and queries can be very short, which produced insufficient estimates of term probabilities. To refine the performance techniques like model interpolation and data smoothing were adopted [112, 163, 194]. Similarly, the order of words to reflect a topic in the documents may differ from the word of words reflecting the query. To overcome this problem, language models try to capture the word order implicitly the adaptation of proximity measures to spots the related terms [148, 47] and relaxing the order of words while imposing the term adjacency [204].

2.4.2 Topic Modeling

Topic Modeling is a popular statistical modeling method that is designed to extract the latent topics from the text and can be used for document clustering and text summarizing. It aids in discovering and annotating the hidden topical patterns that are present across the document collection. Latent Dirichlet Allocation [22] (LDA) is a known example of a topic model and is used to classify text to a particular topic. In LDA, every document is considered as

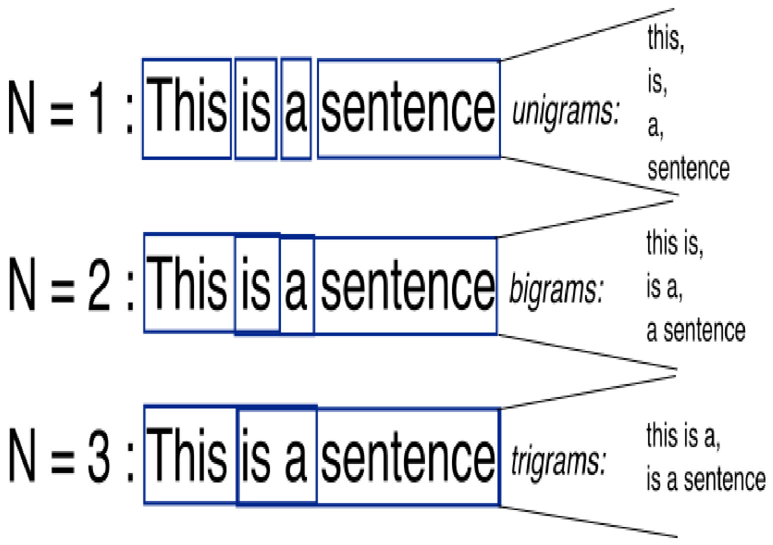


Fig. 2.2 N-Gram Language Model Interpretation

a combination of topics and the model suggests that every word in the document refers to one of the document's topics. For n number of topics $T_1,\ldots,T_n$ and a given document D_i, the likelihood of a document D_i belonging to topic T_j is denoted as $P(T_j|D_i)$. Figure 2.3 shows the plate notation of LDA. The boxes in the figure represent plates whereas the outer plate represents documents, while the inner plate denotes repeated words position in a given document, each of which position is referred to a word and a topic. Collapsed Gibbs sampling [79] is one way the LDA model learns the information on topics and their representation of each document. The procedure for it is as follows:

- At first, for each document, randomly assign each word with one of the K topics. K is a parameter and specified at the start. In this way, all documents have a topic representation and all topics are specified with word distribution.
- Iterate through each document Di and for each word w in Di compute
    1. $P(t|D_i)$ which is a proportion of words in D_i which are allocated to topic t.
    2. $P(w|t)$ which is a proportion of allocation of topic t at all documents d which will be taken from word w.
- Reassign each word w to a new topic, where as the selection of this new topic t is based on the probability shown by the following equation.

$$p(w|D_i) = \sum_{t=1}^{k} p(w|t) \times p(t|D_i) \tag{2.24}$$

A steady state is reached after repeating the reassignment step numerous times and the topic assignments are then chosen to separate the documents.

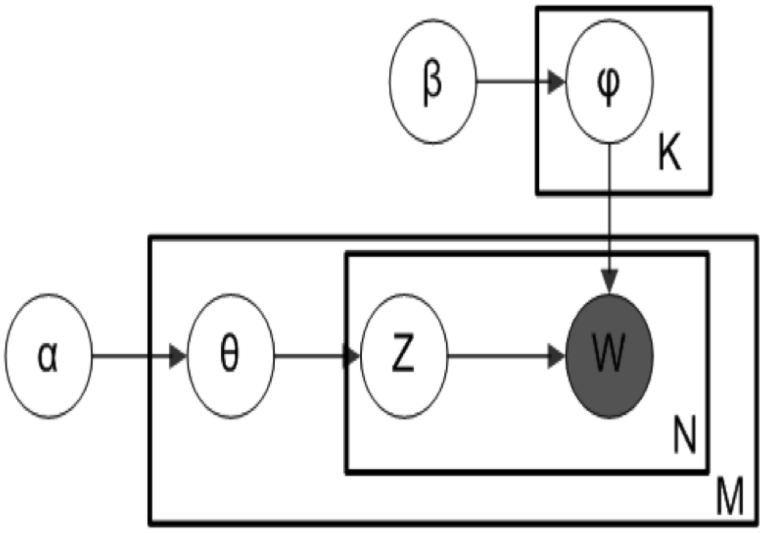


Fig. 2.3 Plate notation of Latent Dirichlet Allocation

2.5 Query Reformulation

The quality of an IR system relays upon how well it can find the relevant documents against a given query. That relies on the choice of words the user has made to formulate the query. Different users may use dissimilar words for the same topic or concept i.e synonyms. So when a search engine tries to locate a word it may be the case that the document uses a different word for the same concept; hence the document may not be chosen as a relevant document for the given query. To reformulate the query approaches like stop words (words like is, a, the, etc.) removal, tokenization, and stemming of words from the document and query are used traditionally to ensure the matching of terms. Following is a brief background on query formulation techniques like term weighting and query expansion that are relevant to approaches considered in the upcoming chapters.

2.5.1 Query term weighting

The reformulation can be done without expanding the query by simply re-weighting the terms such as, terms that are appearing in the relevant documents are assigned with maximum weight and by minimizing the terms that are present in non-relevant documents [14, 47]. The objective of weighting the terms is to give a higher weight-age to the query terms that can be a good discriminator of relevant documents. Tf-idf are the most commonly used term weighting schemes. BM25 [171] retrieval model and the Pivoted document length normalization [202, 203] utilizes the tf-idf schemes. Predictive term weighting is an example of term weighting that incorporates supervised learning approaches that are trained on different IR evaluation metrics. Bendersky et al. [13] showed improvement in performance with verbose queries by using term weights. Similarly, a context-dependent weighting scheme[51] considers the notion of surrounding text in the vicinity of a query term to predict the document relevance.

2.5.2 Query Expansion

Another way to reformulate a query is to automatically add new terms to the query which can overcome the vocabulary mismatch and improve the performance of the query. Pseudo relevance feedback is a widely used query expansion approach that is used to expand the query by utilizing the user interaction and improving the query based on the feedback given by the user. This approach is categorized under local query expansion techniques. The Rocchio Algorithm [175] is the classic algorithm for implementing relevance feedback. It models a way of incorporating relevance feedback information into the vector space model. Rocchio suggested the following algorithm for relevance feedback:

$$\vec{q}m = \alpha\vec{q} + \frac{\beta}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j - \frac{\lambda}{|D_n|}\sum_{\vec{d}_j \in D_n} \vec{d}_j \tag{2.25}$$

Where $\alpha\vec{q}$ represents the initial query, $\frac{\beta}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j$ represents the value of the word as determined in the related document set, and $\frac{\lambda}{|D_n|}\sum_{\vec{d}_j \in D_n} \vec{d}_j$ are the values for each word from the non-related results. The query will expand to a length equal to the number of all unique words present. The parameters α β and λ can be tuned before the process begins. An ideal query is one that returns all of the relevant documents while avoiding all of the irrelevant ones. To estimate this ideal query, Rocchio suggests an iterative feedback process whereby the positive feedback and negative feedback provided by a user is used to guide the query modification. To achieve this, the author suggests giving each word a weight relative to its presence in either the relevant or irrelevant document set. The limitation of the relevance feedback approach is that it expects some of the top retrieved documents to be relevant. However, the quality of newly added terms is substandard if more irrelevant documents are retrieved in the top answer set. One way to resolve this challenge is to consider terms from the whole corpus that can be added to the given query. This approach is known as Global Query Expansion. It takes into account the co-occurrences of terms throughout the corpus and drives the relationship between them based on that co-occurrence. A relationship between two terms can be measured via cosine similarity measure given below:

$$cos(t_i,t_j) = \frac{Nd_{ij}}{\sqrt{Nd_i \times Nd_j}} \tag{2.26}$$

where Nd_{ij} is the total number of documents in which the term t_i and t_j occurs. One limitation of considering the global co-occurrences is that there is a lack to identify the semantic sense of words. The words that appear frequently in the corpus can have many senses and could create unnecessary bias and therefore are poor discriminator between relevant and non-relevant documents [159].

2.6 Query Difficulty

Estimating the query difficulty is an attempt to quantify the quality of search results retrieved for a query from a given document collection [36]. In IR, query difficulty estimation is also referred to as Query Performance Prediction. Oftentimes, information retrieval systems exhibit a substantial variance in accuracy across a set of queries. Systems may display a similar MAP, but quite a considerable variance in performance when considered in a query-by-query manner. A large body of work exists in predicting query performance; i.e. given a particular query, can one predict the expected MAP from a particular IR system? A range of techniques have been considered; these can be broadly categorised into two main categories: pre-retrieval and post-retrieval. Pre-retrieval techniques consider examining the query and looking at features of the query and the query term; these include linguistic approaches [147] and statistical approaches [90, 89]. Post-retrieval, on the other hand, examines features of the returned answer and attempts to gauge the quality of the answer as a measure of the query difficulty.

2.7 Graphs in Information Retrieval

Finding the relevant information against a given query is a difficult task and is considered complex due to a lack of contextual information spread across the document. It is crucial to find the structural similarities within the text and because graphs are considered to be a usual way to represent the structure of knowledge, graph theory is commonly used for information retrieval tasks. In this section, we will discuss the basic terminologies that are used in graph theory and discuss different types of graphs and commonly used measures.

A graph G(V, E) is defined by the data of a finite set of vertices V and a set of edges E which are ordered (directed graph) or unordered (undirected graph) pair of vertices i.e., an edge is associated with two distinct vertices. A vertex can also be called *node* and an edge can also be called an *arc* or *link*. If an edge e combines two different nodes x and y, that edge is considered as incident to x and y and the nodes adjacent to e. A *cycle* in a graph is a series of consecutive edges i.e. chain whose two end vertices are identical. In valued graph G(V, E,w), there is a weight w (usually positive) is associated with each edge that indicates the strength of relationship within the corresponding vertex pair. In this dissertation, we used a valued graph throughout where the arcs between the nodes (where each node is a passage) are valued by the weights that are calculated in various ways. For a finite graph G with n vertices the adjacency matrix is defined as a binary Matrix $M = [v_{i,j}]_{n\times n}$, with $v_{i,j} = 1$ shows that is an edge between the vertices i and j and $v_{i,j} = 0$ if edge exists between i and j. Similarly, for the valued (weighted) graph, in the adjacency matrix, isn't binary anymore and, $v_{i,j} > 0$ indicates the edge weight (non-zero) between the i-th and j-th vertices and $v_{i,j} = 0$ shows that there isn't any edge there. Consider a situation in which the set of vertices V is separated into two disjoint subsets (U,V) such that there are no links between the vertices of the same set and every edge connects a vertex in U to one in set V. This type of graph is called Bipartite graph. One noticeable characteristic of bipartite graph is that the graph is bipartite if and only if it does not contain an odd cycle. Figure 2.4 illustrates a bipartite directed graph.

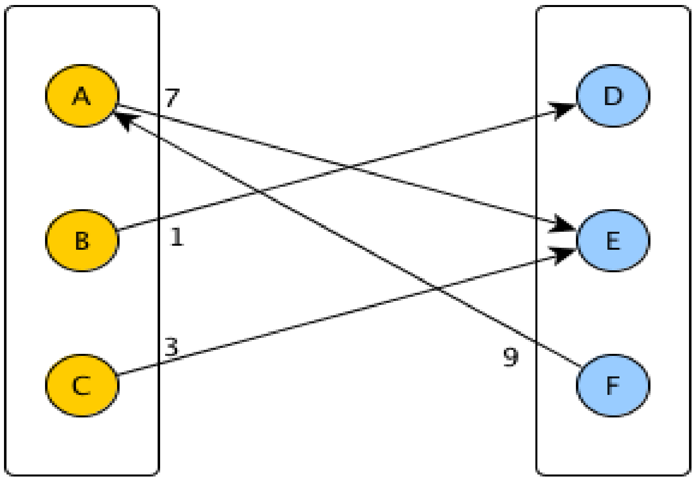


Fig. 2.4 Bipartite Directed Graph

Graph Measures

Graphs are mathematical representations of networks. Different graph measures are used to reflect different graph properties. The two main components of a graph that allows us to analyze the graph are the number of nodes $|V(G)|$ that is denoted as NV and the number of edges $|E(G)|$ denoted as NE.

2.7.1 Degree of a graph

To understand the degree of a graph, we first define the neighborhood of a node. The neighborhood of a node v is the set of nodes to which the node v has an existing link (edge). It is denoted as $N(v)$. The *degree* of a node v corresponds to its number of neighbors. In other words, the degree of a vertex v ($\delta(v)$) is the number of edges adjacent of v.

$$\delta(v) = |N(v)| \tag{2.27}$$

In a directed graph, it is important to distinguish between *indegree* and *outdegree*. Figure 2.5 shows an example of a directed graph with 5 vertices. It has the following degrees, indegrees, and outdegrees.

Table 2.1 Graph degrees, outdegrees and indegrees

Vertex Id	Vertex Degree	Indegree	OutDegree
1	2	0	2
2	4	3	1
3	1	0	1
4	2	1	1
5	1	1	0

The Average Degree of a graph G is another measure to identify the number of edges NE against the total number of vertices NV. For the undirected graph, as each edge connected to two vertices is considered to calculate the degree of both vertices, the average degree ($\delta(G)$) of a graph is computed as:

$$\delta(G) = 2 \times \frac{NE}{NV} \tag{2.28}$$

Similarly for the directed graph, each edge accounts to 1 degree and therefore the average degree is simply the number of edges NE divided by the number of vertices NV.

$$\delta(G) = \frac{NE}{NV} \tag{2.29}$$

2.7.2 Graph Density

The density of a graph G ($den(G)$) measures the number of edges in a set compared to the maximum possible number of edges between vertices in set V. The total number of edges in an undirected graph without loops is calculated as $\frac{NV\times(NV-1)}{2}$ and similarly, for a directed graph it's $NV \times (NV-1)$. Therefore, the density of a directed graph is calculated as follows:

$$den(G) = \frac{NE}{NV \times (NV-1)} \tag{2.30}$$

The density represents the probability that two seperate nodes taken at random are linked. A graph is complete if all the nodes of the graph are connected, which reflects that the $den(G) = 1$.

2.7.3 Clustering Coefficient

The clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together. For each vertex, the clustering coefficient measures the proportion of its neighbors that are themselves neighbors [20]. There are two types of measure exist: the global and the local. The global version indicates the overall clustering in the network, whereas the local indicates the fixity of a single node. To measure the strength of connectivity in the graph, the overall average of the clustering coefficient of all vertices in the graph is used.

The local clustering coefficient (lcc) of a vertex v is calculated as:

$$lcc(v) = \frac{2 \times \varepsilon(v)}{\delta(v) \times (\delta(v) - 1)} \tag{2.31}$$

where $\varepsilon(v)$ is the number of edges connecting the neighbors of node v. The computation of lcc is something referred to as triangle counting that is used as a feature in finding influencers in social networks [70]. Similarly, to measure the average clustering coefficient in a graph $lcc(G)$ we can use the average degree $\delta(G)$ of a graph that reflects the average ratio neighbor nodes that are connected to each other.

$$lcc(G) = \frac{lcc(v)}{NV} \tag{2.32}$$

In a social network, if an average clustering coefficient is low, this reflects that the friends are not tightly knit together, but rather sparse.

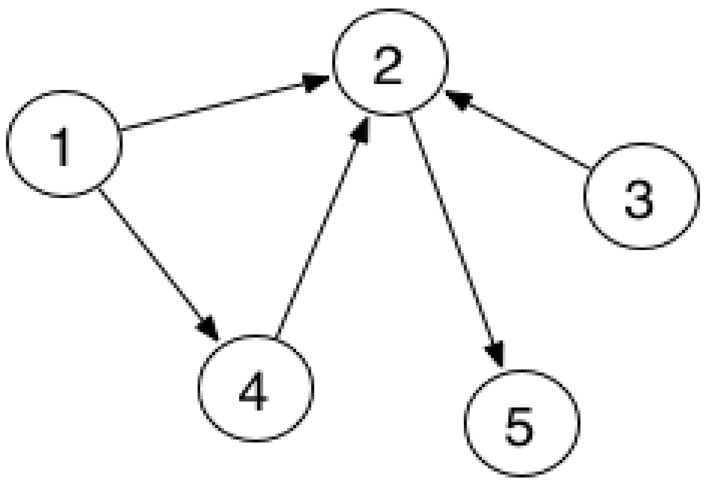


Fig. 2.5 Simple Directed Graph

2.8 Word Embedding

Word embedding (WE) is a popular feature learning technique in Natural Language Processing that is used to map words out of vocabulary into vectors of real numbers. Words that have similar meanings can have the same representation of word embedding. Different methods can be used to generate the embedding like co-occurrences matrix [125], neural networks [142], probabilistic models [77]. One major limitation of word embedding is that different meanings words are combined into a single representation. To resolve this problem, researches looked into multi-sense embedding to adjust words with multiple meanings in different vectors [98]. The following are some of the keyword embedding models used for NLP and Information Retrieval tasks.

Word Embedding Models

To facilitate the comparison between different models, the following notations were considered. A training corpus T with $w_1, w_2, w_3..., w_n$ belongs to vocabulary V of size $|V|$. The dimensions are denoted as d with an output embedding of v'_w. The objective function J_θ is optimized to θ which is the model parameter that for every input x generates the output score of $f_\theta(x)$.

2.8.1 Neural Network Model

Bengio et al. [16] proposed a neural network-based language model that has a feed-forward hidden neural network layer that is used to predict the next word in the sequence. The goal was to learn a model in a way that it produces a high out of sample likelihood. They were the first to introduce what we now refer to as word embedding, as a real-valued word feature vector associated with each word in the vocabulary. A lot of the current approaches have been build upon the architecture they introduced. The fundamental building blocks upon which the current approaches are built are layers based approaches whereas the $Embedding$ $Layer$ is used to generate the word embedding, the $MiddleLayer(s)$ that is used to apply the non-linearity of the concatenation of WE of p preceding words. Lastly, the $Softmax$ $Layer$ is used to produce the positive probabilities for the words in vocabulary V. Computation of the softmax layer is found to be very costly as it is directly proportional to the vocabulary size and it is considered to be even greater than the computational cost required from the $n-gram$ models [222, 237]. The utilization of parallel processing helps in overcoming that issue.

2.8.2 Continuous-Bag-of-Words (CBOW) and Skip-Gram Model

Machine learning algorithms cannot accept text directly. A numerical word representation is needed. A technique named Word2vec was introduced by Mikolov et al. [141] that convert words into vectors that can be used to measure the relationship between vectors by applying different mathematical operations. It is a 2 layer neural network. In simple terms, Word2vec tries to make similar embedding for words that has a similar context. For example, if we take two sentences like ‘The kid said he would grow up to be a pilot’and ‘The child said he would grow up to be a pilot’. In both sentences, the word *kid* and *child* are different words but share the same context. Therefore, by using word2vec *kid* and *child* may have similar embeddings.

Two different iteration-based algorithms are used.

1. Continuous-Bag-Of-Words (CBOW) model: This model predicts the center word from its surrounding context. The main aim of this model is to take a word in a specific context and maximize its probability. The probability of the word w_i is considered in the following way

$$P(w_i|w_{i-s}, w_{i-c+1}, ..., w_{i+1}, .., w_{i+s-1}, w_{i+s})$$

where w_i is the ith word and s is the window size. The CBOW model applied to train two matrices: One is the input word matrix, which is indicated as $I\varepsilon \mathrm{R}^{N\times|W|}$. Here the ith column of I is the N-dimensional vector against the given word v_i. Second is the output matrix, indicated as $O\varepsilon \mathrm{R}^{|W|\times N}$, such as the jth row of O is also the N-dimensional vector for the word u_j. The following figure illustrates the input and output layers of the CBOW model.

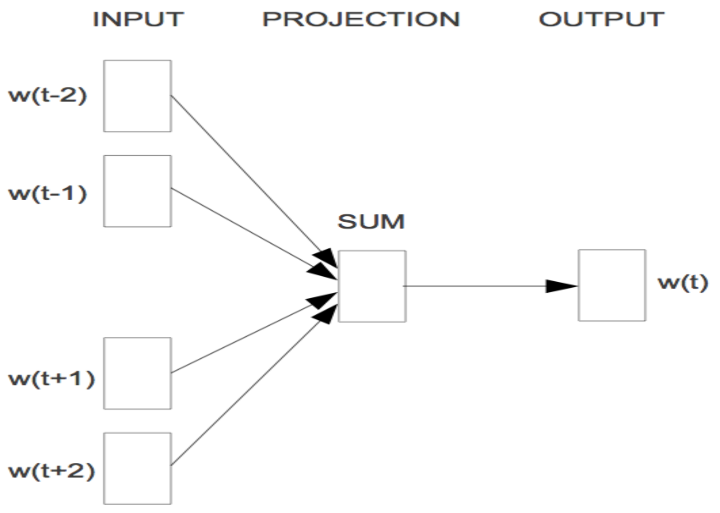


Fig. 2.6 Continuous bag-of-words [141]

As shown in figure 2.6, Mikolov et al. used both the s (window size) words proceeding and preceding the target word w_t to predict it. The object function of CBOW is given below.

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} logP(w_i|w_{i-s}, w_{i-c+1}, ..., w_{i+1}, .., w_{i+s-1}, w_{i+s}) \tag{2.33}$$

2. Skip-Gram: In skip-gram model (shown in Figure 2.7), instead of using the surrounding words to predict the center word, the center word was used to predict its surrounding words. Unlike CBOW, given a center word, this model aims to maximize the probabilities of the context word, which can be written as:

$$P(w_{i-s}, w_{i-c+1}, ..., w_{i+1}, .., w_{i+s-1}, w_{i+s}|w_i)$$

It sums the log probabilities of the surrounding s words based on the window and the object function of this model can be denoted as follows:

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c,\neq 0} logP(w_{t+j}|w_t) \tag{2.34}$$

where c is the size of the training context. Higher accuracy can be achieved with further training examples i.e. larger value of c. The softmax function of the skip-gram model can be defined as:

$$p(w_O|w_I) = \frac{exp(v_{wO}^{'}{}^\top v_{wI})}{\sum_{w=1}^{V} exp(v_w^{'}{}^\top v_{wI})} \tag{2.35}$$

In the above equation, w_I denotes the center word and the surrounding words are written as w_O. Moreover, v_w and $v_w^{'}$ are the input and output vector, and representations of w V is the size of words in the vocabulary. This softmax is not the most efficient one in terms of time complexity because as per the equation above, we are taking the sum over all the words in vocabulary to normalize this function. To make this approximation computationally efficient, the Hierarchical Softmax and Negative Sampling approach were suggested by Mikolov [142]. In Negative Sampling, the log probability of the softmax is maximized by considering the subset of vocabulary. Similarly, in Hierarchical softmax, instead of evaluating all V words, it only evaluates $log_2(V)$ nodes. It employs the binary tree representation of the output layer with the V words as its leaves and the tree nodes represent the relative probabilities of its child nodes and that's how the overall time complexity is reduced for the softmax function.

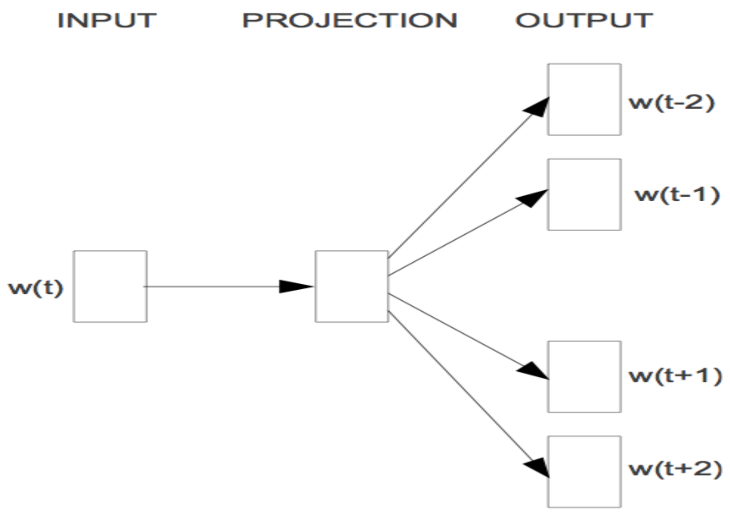
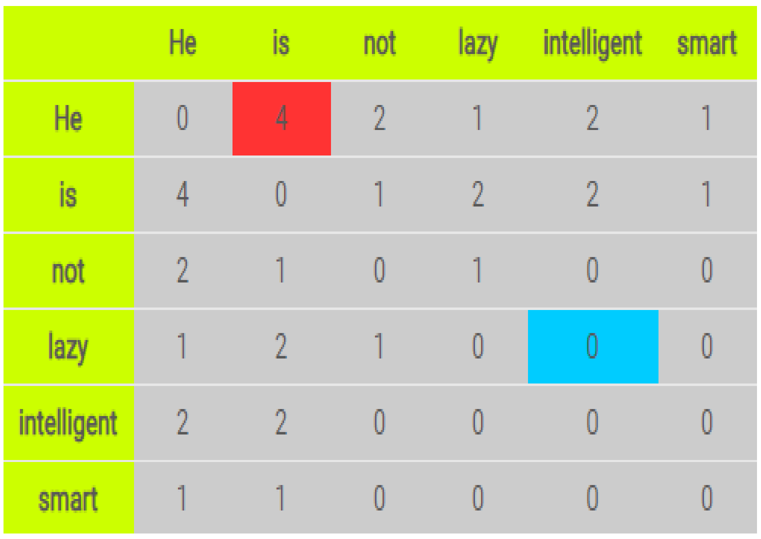


Fig. 2.7 Skip-Gram Model [141]

2.8.3 Co-occurrence Matrix

To capture the semantic relationships, co-occurrences matrix is a commonly used technique in the IR domain. The co-occurrence matrix (CM) counts how often words co-occur in all the documents. In this technique, co-occurrences correspond to e.g. pair of words $w1$ and $w2$ is the number of times these words appeared together in a selected corpus provided there is a context window. The context window is identified by the direction and the size of the window. The window of size 3 with preceding direction means for each word the co-occurrence of the previous 3 words is considered to generate the count in the co-occurrence matrix. Consider our corpus is: "He is not lazy. He is intelligent. He is smart". Figure 2.8 co-occurrence table of this corpus with a context window equal to 2.



	He	is	not	lazy	intelligent	smart
He	0	4	2	1	2	1
is	4	0	1	2	2	1
not	2	1	0	1	0	0
lazy	1	2	1	0	0	0
intelligent	2	2	0	0	0	0
smart	1	1	0	0	0	0

Fig. 2.8 Co-Occurence Matrix Example
1

From a mathematical point of view, one way to process CM is the singular value decomposition (SVD) [17]. The matrix is indicated in a way that the first k columns of a matrix U and V are word embedding matrices and the matrix SDV($\hat{X}$) is computed as a product of matrices i.e. USV^T (Shown in Figure 2.9). Where U and V are both orthogonal matrices. An important thing to note here is that the product of U and S represents the word vectors and V represents the word context.

Hyperspace Analogue to Language (HAL)

Hyperspace Analogue to Language was first proposed by Lund et al. [136] in 1996 as a theoretical concept of mapping term co-occurrences to infer meanings about words. Lund used a sliding window to map the co-occurrence of terms. They use a matrix to store the associative terms. With rows being used to store preceding terms and columns contain a record of all following terms within range of the window parameter w. In their initial paper they experimented with varying the w to obtain the ideal window size. The distance metrics used in the all come from the Minkowski family of distance metrics, which include the familiar Euclidean (if r = 2) and city-block (if r = 1) metrics:

$$distance = \sqrt[r]{\sum(|x_i - y_i)^r} \tag{2.36}$$

The vector comparison procedure utilized in their experiment was the Euclidean distance between vectors, which corresponded with $r = 2.0$. For each target word, a Euclidean distance was computed from the word to each vocabulary item. The terms were then sorted, and the neighbors with the smallest distances were examined.

$$\underset{m \times n}{\overset{\hat{X}}{\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} \approx \underset{m \times r}{\overset{U}{\begin{pmatrix} u_{11} & \cdots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix}}} \underset{r \times r}{\overset{S}{\begin{pmatrix} s_{11} & 0 & \cdots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix}}} \underset{r \times n}{\overset{V^{\mathsf{T}}}{\begin{pmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix}}}$$

Fig. 2.9 Singular Value Decomposition Matrix
2

2.9 Test Collections

Three different test collections are used in this dissertation. The test collections vary in document length. Each test collection contains documents, queries, and relevance judgments. To evaluate the performance of the system, these pre-defined relevance judgments against each query are used. In given test collections, topics i.e queries are represented as *title* and *description*. Queries defined under title are categorised as key word based queries whereas the description queries depicts the full written queries in natural language by the user to the search engine [233]. We employed the WebAp, Cranfield, and Ohsumed test collections in this thesis because they were publicly available. As the collections vary in length and characteristics, it was suitable to test our hypotheses and passage-based approaches in a relatively quick manner. Other test collections like GOV2 and ACQUAINT are large-size test collections and due to the computational expense of applying the passage boundary and the retrieval approaches, we opted to use the small to medium-size test collections. Test collections like Cranfield and Ohsumed have been used extensively in the past in literature for IR tasks. WebAp is relatively a new test collection and is used for passage retrieval tasks. As the test collection is a subset of the GOV2 collection (which is commonly used in IR), it was

Table 2.2 Document Collections

	Total Docs	Total Passages	Total Queries	Passage Window Size
WebAp	6399	146000	150	250 words
Cranfield	1400	7722	225	30 words
Ohsumed	233,445	1404440	97	30 words

suitable for us to apply it for ad-hoc retrieval by using the passage evidence approaches. The characteristics of the opted test collections is specified in Table 2.2. Also, a brief overview of all the test collections used in this dissertation is mentioned below.

2.9.1 WebAp

This test collection was acquired from the GOV2 collection of TREC 2004 Terabyte track. The dataset includes 6399 documents and 150 query topics and relevance judgment of the top 50 documents per query topic. It has been used for passage level retrieval [108, 190], document re-ranking task [126, 44, 190], and also for sentence-level retrieval in question answering (QA) task [38, 228]. Annotation at passage level (GOOD, FAIR, PERFECT) is also included to differentiate the query-passage relevance in this test collection. The annotators found 8027 relevant answer passages to 82 TREC queries, which is 97 passages per query on average. From these annotated relevant passages, 43% of them are perfect answers, 44% are excellent, 10% are good and the rest are fair answers. The annotations are useful to evaluate the sentence or passage retrieval task [108, 228]. However, in this thesis, we employed passage evidence to improve the performance in ad-hoc retrieval task. A ranking approach or an error analysis can be devised by using those annotations. But since the other employed test collection in this thesis does not contain the passage based annotations, therefore, we did not take the annotation into account in our evaluation process. As the document length is large compared to the other test collections we employed in this thesis, therefore, we divided passages using overlapping window based approach of size 250 words. For each query a title (small query) and a full description (longer query) was given. In our experiments, we considered full description text to send as a query for our graph based experiments. However, for chapter 4 and chapter 5(part (a) query expansion), we used small queries.

2.9.2 Ohsumed Collection:

The Ohsumed collection consists of titles and abstracts from 270 Medline referenced medical journals. It contains 348,566 articles along with 106 search queries. In total, there are 16,140 query-documents pairs upon which the relevance judgments were made. These relevance judgments are divided into three categories i.e. definitely relevant, possibly relevant, or not relevant. For experiments and evaluation, all the documents that are judged here as either possibly or definitely relevant were considered as relevant. Additionally, only the documents with textual content were used for retrieval because there was a small percentage of empty documents (documents with no textual content) as well in the test collection. Therefore, the experiments were conducted on the remaining set of 233,445 documents from the Ohsumed test collection. Also, to calculate the overall performance we considered only those queries, which had any relevant document(s) listed in the judgment file. Out of 106 queries in total, 97 of them were found to have the relevant document(s) associated with it. This document collection is fairly large in terms of document size but shorter in terms of document length as compare to the WebAP test collection. Given the relatively small document lengths, in defining passage boundaries, half overlapping window size of 30 words is used for this collection which creates a document set of passages of size 1.4 million pseudo-documents that gives 4-5 passages per document. Moreover, it does not include any annotation at the passage level. Ohsumed collection is employed for many key IR tasks, such as Text Classification, Ad-Hoc Retrieval, Query Performance Prediction, and Passage Retrieval/Re-ranking task [189, 59, 191, 63, 151, 209, 225, 66].

2.9.3 Cranfield Collection

This collection contains 14,00 documents and 225 queries along with the relevance judgment file [214]. The collection also includes the relevancy scale, which we did not use for our experiments. It is consider as a small size test collection and similar to the Ohsumed collection, the document length is short. Therefore, passages length for the cranfield collection is also measured by the overlapping window based approach of size 30 words. As the Cranfield collection is small in size, it has been used mainly for Text Clustering [55, 83], Query Expansion [53, 5] and Ad-hoc Retrieval task [76, 86, 94].

2.10 Experimental Setup

To perform experiments in our work, there are few tools and libraries that we employed. Following is a brief description of the tools used in this dissertation.

2.10.1 Apache Solr

We indexed all documents and passages to perform the search by using SOLR[3], which is built on top of Apache Lucene[4]. In this system, a vector space model(VSM) is adopted with a weighting scheme based on the variation of tf-idf and Boolean model (BM) [122]. We used the default settings of Apache Solr and indexed all the documents and retrieved results by using the Spring Framework Implementation of SolrJ API of the Java platform. In this thesis, we refer to the Lucene document-query similarity score as *LVSM* as one of the baseline methods. Since the Lucene query document similarity score (*LVSM*) can be greater than 1 in SOLR, we normalized this score to be in the range of 0-1. Lucene's documentation does not use a well-defined notation to represent how the similarity score between a query Q and a document D is calculated. Equation 2.37 denotes Lucene's scoring formula as described in Lucene's documentation.

$$score(Q,D) = coord(Q,D) \cdot qnorm(Q) \cdot \sum_{t \in Q} tf(t \in D) \cdot idf(t)^2 \cdot boost(t) \cdot norm(t,D) \quad (2.37)$$

In the above equation, t denotes to term. The functions, in order from left to right, are the coordination factor between query and document, query normalization factor, a term-frequency transformation of terms in a document D, inverse document frequency transformation in a document D, query boost, and document-length normalization factor to the document D. A significance of the coord-factor is that many query terms can be matched within a document. Through a coordination factor, documents matching more query terms can be rewarded by a factor value i.e. coordination factor, which is usually larger when more terms are matched. Query boost on a term specifies the contribution of a query term to the score of a document by multiplying it by the boost value of that query term. Furthermore, the document length normalization factor is used, to normalise the vector equal to or larger than the unit vector.

2.10.2 DeepLearning4J

Deeplearning4j [5] is a deep learning programming library written for Java and the Java virtual machine and a computing framework with wide support for deep learning algorithms. It also provides an implementation for the Word2vec model and also extends its API with Apache

[3]http://lucene.apache.org/solr/5_2_1/index.html
[4]https://lucene.apache.org
[5]https://github.com/eclipse/deeplearning4j

Spark[6] that is an implementation to support the distributed computation of machine learning models. We used the Apache Spark's MLlib implementation of word2vec to generate models.

2.10.3 Pre-Processing

For our experiments, we used Solr-5.2.1 which is built on top of LUCENE[7]. Solr provided the functionality of removing the stop-words at indexing time. As shown in Figure 4.1, we used that functionality to remove the stop words[8] from both collections. We have seen that the ranking score after removing the stop-words was improved and the performance of the system for all the employed test collections was improved.

It is to note that the removal of the stop-words is dependent on the type of system and task at hand. For example, for a QA task or building a language model where context is important, removing stop-words can cause in loss of information. However, in this thesis, we focused on an ad-hoc retrieval task where relevant documents are returned on a given query. The stop-words we removed are mainly the standard words that provide little to no unique information that can be used for retrieval tasks. We did not remove any word domain-specific words. The standard pre-processing approach of stop words removal had been applied in the past on all the employed test collections as well [229, 231, 223].

2.11 Summary

In this chapter we presented an overview of classic Information Retrieval concepts approaches that are used for passage level retrieval. We started by giving a brief introduction of main IR models like vector space model, Boolean model, and probabilistic model. We then discussed the general term weighting techniques employed by the aforementioned IR models and the measures to evaluate the performance of an IR system. As the standard bag of words model does not consider the contextual notion around the text, we discussed different language models in section 2.4 and word embedding in section 2.7 approaches that can be used to understand the context around the segment of text i.e. passages. For any information retrieval system to perform well, transforming the query in a correct way is extremely important. We discussed briefly how term weighting can be applied to a query and the notion of query expansion in section 2.5.

In this dissertation, we explored different techniques to capture the context around passages. We used graphs to represent and measure the similarities within the text. In Section

[6]https://spark.apache.org/docs/latest
[7]http://lucene.apache.org/
[8]http://www.ranks.nl/stopwords

2.6 we briefly discussed graph theory and the principal graph measures commonly used to measure certain properties of the graph. Finally, we presented a short overview of the test collections employed in this dissertation and the tools that are integrated together to perform the experiments.

Chapter 3

Related Research

This chapter presents an overview of the state-of-the-art in passage retrieval research that is related to the contributions of this dissertation. In order to cover the related background, we start by giving a review of passage-based retrieval in the IR domain. Any passage-based retrieval system is divided into two main parts: the pre-processing and the post-processing. The pre-processing covers the division of passage from documents i.e. passage selection and the post-processing explores the use of passage-based evidence to improve the performance of a retrieval system i.e. passage manipulation. We begin by presenting an overview of both processes. This is followed by an overview of query augmentation techniques that were used in improving the accuracy of the system. After that, we continue by discussing the use of graph-based approaches in information retrieval and especially with the consideration of passage-based graphs. Finally, we conclude this chapter by highlighting the previous work done for Query Performance Prediction (QPP) task, particularly focused on the approaches that employed passage information for the QPP task.

3.1 Passage Selection

Before utilizing the passages to improve the performance of any retrieval system, it is important to identify the boundary of passages. The types of passages explored by researchers can be grouped into three classes: discourse, semantic, and window.

3.1.1 Discourse Passages

Discourse passages rely on the logical structure of the documents separated by punctuation, paragraphs, or sentences. These separators within the document are not enough to define passages. Different test collections have variation in the size of paragraphs and sometimes a separator doesn't reflect the shift in the content within the document. Callan et al. [32] introduced the bounded passage approach, where each passage is constrained to be greater than a defined minimum number of words and less than a maximum number of words. They combined short paragraphs together to have at least min of 50 words passage and divide large paragraphs into passages of a maximum size of 200 words. The minimum and maximum value can be adjusted as needed. Similarly, Yenala et al. [230] have introduced a biomedical passage retrieval method. A sentence as a separator was used to identify the length of the passage. Their system was based on cosine similarity and existence test score between the question and each sentence in retrieved documents. To find the relevant documents and rank them in the correct order, they used the PubMed search engine and cosine similarity score. Then, they have extracted the sentences from the abstracts of the 10 top relevant documents and kept only the 10 top sentences matching most with the biomedical question after finding similarity scores for all sentences.

3.1.2 Window Passages

Window size approaches consider only the word count to separate the passages from each other, irrespective of the written structure of the document. Usually, the window passages have problems regarding the division of the relevant paragraph into different passages, but that problem was tackled by using the overlapping passage approach. The idea here is to start the first passage on the basis of the first term from the given query and from there creating the passage of n/2 length at a given specified number n. For example, if the length of the passage was 200 and the first matching term starts at position 73, overlapping passages of 100 words would begin at positions 73, 123, 173, 223 and so on. Similarly, a variant of the same approach was used by Croft[135] as well.
Instead of using the fixed number of words for the passage, a more dynamic alternative to window-based is arbitrary passages proposed by Kaszkiel et al. [106]. In this approach, a passage can start at any word in the document. It is categorized into two sub-classes. Fixed-length arbitrary passages resemble overlapped windows but with an arbitrary starting point. Variable-length arbitrary passages can be of any length. Clark et al. [42] also used arbitrary passages approach in their TREC-9 experiments. Pre-processing the query is undertaken before the passage retrieval and passage selection where the top 10 passages were selected from all different documents based on the query selection rules. All the passages were ranked based on their length and on different weights of the terms assigned to it in the query. Similarly, Tiedemann et al. [210] demonstrated how a typical text can be segmented using sliding windows and overlapping parts. They have shown that this approach is sensitive to the window length which needs to be tuned on training data. After comparing with other

semantic-based approaches for document segmentation (TextTiling and conference chain), they have found that creating overlapping segments of a small length is the most successful approach. The failure of employing semantic approaches for segmentation may be due in part to the approach's tendency to produce a large number of overlapping passages, which does not appear to be ideal for passage retrieval. In addition to that, window-based passages have been used in Element Retrieval as well. Element retrieval is used to search XML documents and identify relevant XML elements. Huang et al. [99] investigated element retrieval, using a general passage retrieval algorithm. They divided the XML documents into overlapping and non-overlapping fixed window-size passages. They used INEX 2005 ad-hoc test collection to conduct experiments. They have shown that the overlapping window approach generates better results on the query generation model (GEN) and term frequency model (FREQ) as compared to the non-overlapping window approach. The best results were generated by using the Kullback-Leibler model(KL) with the high window size i.e. $size = 400$ words. As we can see from the aforementioned research that there is no standard window size applicable to all test collections. A varied approach is adopted by all researchers (same as us in this thesis) to identify the best overlapping window size to segment the document into passages.

3.1.3 Semantic Passages

Usually when writing a document, paragraphs are not always written coherently and are used to indicate a change in the discussion. Sometimes moving to a new paragraph can only be to assist reading. Therefore, paragraph separators are not always helpful in identifying passages with similar topics that share the same context. Other than the passage boundary approaches, where the passages were separated by a separator or by the number of words in it, there are other techniques that researchers have adopted in the past for passage level retrieval. The idea behind these approaches was to try and divide the passages into topics, sentences, or sub-topics that can be considered as a passage itself. As defined by Hearst et al. [91], Text-Tiling is a technique for subdividing texts into multi-paragraph units that represent passages, or subtopics. It uses the co-occurrence of lexical terms and their distribution to identify the topic shift in the document. It captures the overlapped words between adjacent windows of a defined size and the gaps are calculated by identifying the cohesion score between the two blocks. If the gap is lower, it denotes a high shift in the topic. Figure 3.1 shows how the gaps and passages are identified in the basic text-tiling approach.

Xiao-Jun Wan et al. [221] described a new retrieval model for similar documents by using the Text-Tiling approach along with the graph theory. They compared the results with BM25, vector space model, and cosine similarity. They stated that the cosine similarity does not capture the structural similarity of the document, which can be measured by using

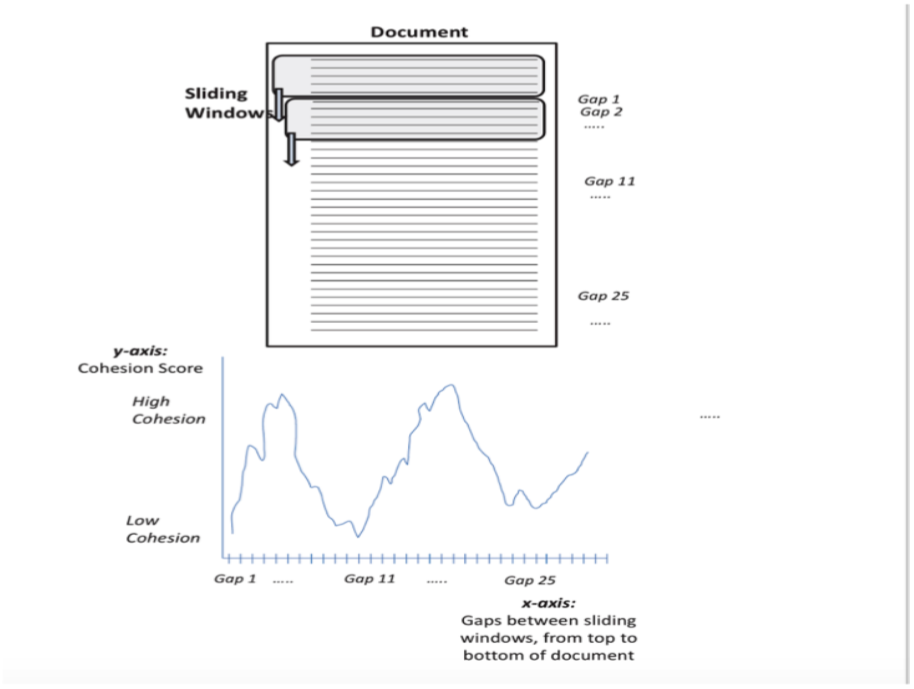


Fig. 3.1 Text Tilling for defining passages.[91]

their proposed retrieval model with the Text Tilling approach. Their model considers the structural information of the document by taking into account the different word distribution over different text segments. It calculates the similarities for a different pair of text segments and then combines them together for optimal matching by using the concept of graph theory, where the documents were divided into a set of subtopics and then a bi-pirate graph was generated for the query document and any other document in the collection.
Moreover, the Text-Tiling approach was modified by Martin et al. [169] where instead of comparing the blocks of text via bag-of-words vectors, they used topic IDs assigned by the inference method of Latent Dirichlet Allocation (LDA)[21]. LDA defines a document is just a collection of topics where each topic has some particular probability of generating a particular word. Figure 2.3 shows the plate notation of LDA. In LDA sometimes a word can be assigned to a different topic id during the inference step. Therefore, they used the most frequent topic id that is assigned to that word by keeping the track of topic-word assignment at each inference step. They also trained the topic model on similar topics and used similar documents in content for testing. The algorithm performs better in running time complexity as it performs the segmentation in linear time.

3.2 Passage Manipulation

Once the passage boundaries are defined, the next step is to utilize evidence from these passages to improve different tasks in information retrieval. Researches have used passages not just only for ad-hoc retrieval task, but also considered passage evidence for tasks like document summarizing[107, 130], question answering[131, 235, 210], named entity recognition [50]. Moreover, researchers employed several ways to manipulate passage evidence to improve the performance of an IR system.

3.2.1 Ad-hoc Retrieval using Passage Evidences:

In IR, the traditional approaches consider the document as a single entity. However, splitting the document into passages can be beneficial. For example, It is possible that a lengthy document contains a passage that is very relevant to the query, but that passage loses its significance when seen in the context of the entire document. Therefore, by splitting the document into chunks, we will be able to retrieve the passage and its original document easily. TREC's relevance judgment authorities consider a document to be relevant even if a chunk of text (passage) in that document encapsulates relevant information [217]. Therefore, it is useful to utilize the retrieval techniques to a chunk of a document text compared to the whole document. Retrieving passages and not only the documents can also be viewed as Ad-hoc Information Retrieval [150].

Callan [32, 75] demonstrated different ways of defining and ranking passages and more importantly how passages can be used in the retrieval process. He showed that ordering documents based on the score of the best passage may be up to 20% more effective than standard document ranking. Similarly, for certain test collections, it was concluded that combining the document score with the best passage score gives improved results. This outcome confirms that adding passage-based evidence upgrades the effectiveness of the inference network, which reflects the overall strength of the answer set. Similarly, Buckley et al. [30] also combined the global similarities (document retrieval) and local similarities (passage retrieval also discussed by Ren et al. [167]) to improve the performance of ad-hoc retrieval. Both similarities were combined in a more complex manner, to generate scores for ranking. Their ranking formula not only considers the best passage at a document level but also considered the overall highest ranked passage from the answer set and use that to normalize the overall score. Their approach was illustrated as more of a precision-oriented approach because the results presented for the top few documents (P@K) were significantly better than the normal document level approach. Use of the best passage from the document was not the only way to rank the documents. As it is often the case that a document is divided into multiple subtopics. If certain concepts in the text occur close to each other then it can be a good indicator of topicality. Hearst et al. [92] used a text tiling approach to find the passage boundaries based on those sub-topics in the full-length document and showed that instead of only using the best passage with the maximum score, summing the top segments (passages) for each document gives overall a better ranking as compared to the ad-hoc document ranking approach.

Salton [182] discussed another idea to calculate the similarity of the passage to the query. They re-ranked and filtered out the documents that have a low passage score associated with it. They included all the passages that have a higher score than its overall document score, and then used these scores to boost or penalize the final document rank. In this way, the document that has a lower score to the document level score but a higher score at passage level for certain passages will get a better ranking score in the end. In addition to that, Albahem et al. [3] utilized the passage evidence to improve document retrieval in TREC (Text Retrieval Conference) Dynamic Domain Track[227]. In this track, the user interacts with the search system and supplies feedback and as a result, the system decides either to terminate the search or return extra documents to the user. They hypothesized that with the help of interactive relevance feedback, use of passage evidence would be effective. To score the document at a passage level, they used the maximum passage relevance score for the document. The results with the passage evidence were more effective compared to the document-level score. However, the results showed variations across different queries and the improvement was not significant. They also utilized passages for query expansion by using the Rocchio algorithm[175]. For relevant documents, they concatenated the relevant passage from the relevant documents into a pseudo-relevant passage. They concluded that using the passage for query expansion resulted in a slight improvement compare to the baseline. However, the difference wasn't significant.

Similarly, Bendersky et al. [13] used the measure of document homogeneity and heterogeneity to combine the document and passage similarity with the query to retrieve the best documents. They assumed that a document is more homogeneous if the passages of those documents are more similar to each other. They observed that long documents tend to be more heterogeneous (diverse in topics) than the shorter ones. To use the passage level evidence, their scoring method used the maximum query-similarity score that is assigned to any passage in the document ranking. As for their passage based language model, they used the simple unigram based standard to estimate the probabilities at passage and document level. Moreover, Krikon and Kurland [115, 12] used a different language modeling approach where they tried to improve the initial ranking of the documents by considering the centrality of the documents and the passages by building their respective graphs. They took the initial answer set of documents and generated a bi-partite graph between the documents and the passages similar to that document. All those passages come from the initial answer set of documents. The edges denote the inter-term similarities and the centrality is computed using the hubs and authorities (HITS) and an influx approach[127, 120, 111]. A passage connected to more documents from the initial list was considered to be more central. They reported that their approach performed better than the normal maximum passage approach and some variation of interpolation score of maximum passage score with document score.

It is important to understand that there is no universal way to define a passage. The passage that is most suitable for the query is highly dependent on the characteristics of the query and documents in the test collection. Due to the recent improvements in the learning-based models, researchers are using neural networks for passage evidence that could improve the Ad-hoc Document retrieval. Qingyao and Croft [2] developed a passage-based neural model that uses the evidence given from the passages for the document retrieval. They used a learning-based approach to weigh the passages of different sizes and granularity and did not adopt the usual single window size approach for passage extraction. They introduced a fusion framework that aggregates the passage score based on its document properties and relation with the query characteristics. They compared their results with the work done by Liu et al. and Ponte et al. [135, 163] and showed that their neural passage model outperformed the previous passage-based retrieval models. Moreover, Kurland et al. [198] introduced a learning-to-rank (LTR) [133, 134] based document retrieval method by exploiting the high ranking of passages against the given query. To identify the correct ranking for passages, they used the learning-to-rank approach as well. They noticed that most of the passage-based document retrieval approaches utilize the passage-query similarity which is either in the form considering the highest query similarity of passage within the document or by integrating the passage similarity with the document-query similarity. Hence, they also considered LTR passage features that were query independent like stop words, entropy (passage with smaller entropy will tend to be more cohesive and more focused on a single topic) inspired by Bendersky and Croft's work [11] which employs multiple content-based features that can reflect the quality of a document. They first created the initial document list by using the standard language model approach and then passed that list to the LTR method to obtain the initial ranking. This ranking list was used as their baseline. Later on, they compared this variation of passage ranking models that were either based on LTR or by using the passage-based feature vector. They concluded that combing the LTR score of passage and the document gives the best results. Furthermore, recently contextual embeddings [48, 149, 143, 156] are also becoming popular to re-rank the documents by using passage retrieval.

Jong [102] proposed an approach which involves considering the score of passages generated from an evaluation function to effectively retrieve documents in a Question Answering system. They consider passages of different length and neglects the conventional way of fixed length passage approach. The passages are set based on the occurrence of each query term in the document. Their ranking function determines the score of all possible passages by calculating the proximity of the different terms used in the query with different passages and takes the maximum proximity score (most appropriate passage) for the document ranking.

3.2.2 Passage based Language Models

While searching for any document, it is the user's responsibility to formulate the query with the words that would likely appear in the returned document. The language modeling approach to IR directly models that idea. It is defined as a probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability $P(w_1, \ldots, w_m)$ to the whole sequence, that specifies the probability of that document in the language (as opposed to all the other documents that one might encounter).

Question Answering Task

The most commonly used LM is the n-gram model[211].TREC's Question answering (QA) task is used to support the evaluation of newly build QA systems[219, 218]. In QA track a passage task is used to locate the segment of text that matches the answer to the asked question. Using the exact phrase answer lacks context and therefore small segment of texts are more suitable to the user's need[131]. Dell and Wee [235] used the n-gram approach to passage question answering. They divided their QA system into two major modules. First was the question classification (from which class the question belongs to) and the second one was passage retrieval. For passage retrieval, they first retrieved the top relevant passages from the document collection by using external software (MG software [224]). To rank them as per their likelihood of having the right answer, the context of each question's answer was identified based on the question-topic language models that were generated by the web search results. For every question, they search the web (Google was used) to build the question-topic model and augment it with the set of probabilistic constraints. Then they use the same question to get the relevant passages from the local collection and rank it by using the question-topic language model. Passages were defined as half-overlapped window passages of size 30. For every result they get from the web, they build a smoothed uni-gram model. They defined certain constraints on the answer type and its context as well. This helped them to include some previous knowledge into the question-topic model as well. Once the question-topic model is generated, they also generated a smooth unigram model foe each passage and then took both models (passage and question-topic) and applied Kullback–Leibler (KL) divergence [117] between them to measure the similarity between the question and passages. Smaller KL divergences reflect the higher relevance of a passage to the topic of a question. Similarly, Galko et al. [72] used the neural network-based approach to improve the passage retrieval for the Question Answering(QA) task in the Biomedical Domain. They used the weighted combination of word embedding terms by using word2vec[142] and measured the cosine distances between the query terms and the passages. One issue with using only vectors generated from approaches like word2vec is that the weights assigned to words are generated via uniformed weighting (term frequency) and because of that, the abundance of stop words penalizes the importance of other significant words that reflects the true semantic meaning of the text. To tackle this problem, they used non-uniformed weights (idf) to capture the cosine distance. They combined the words from the biomedical corpus and a general questions set[166] to avoid giving higher weights to words like '*what*', '*when*', etc. and boost weights to the more effective words like '*doctor*', '*veins*', '*disease*', etc. Finally, they compare their results with the previous neural-net based models [81, 158, 123] and reported improvements with their approach. In a general QA task or for simple passage retrieval, queries related to certain topics can be answered by either simple facts (like one-word answers or entities) or it might take several sentences or segments of texts to fulfill the user's need [235]. Semantic approaches have been adopted in the past to capture the sentence based similarities between the text [154, 228, 37] as well.

Query Likelihood Model

A basic approach for using language models in Information Retrieval is the query likelihood model where the document should be ranked in decreasing order of their probability of relevance to the query. It considers the fact that a document is independent of its relevance to the query and its prior probability is considered as uniformed across all the documents. So, documents are just ranked based on Language Modeling i.e., the probability of a query being generated by a document. However, Ganesh et al. [73] relaxed that assumption for passage retrieval scenarios by calculating the prior probabilities of a passage being relevant or not to the question and used them for ranking the passages afterward. For relevant and non-relevant text, they created different uni-gram language models and used KL divergence to measure the distance between the passage uni-gram model and the relevant or non-relevant model. Then, they combined these calculated probabilities with the query likelihood of the relevant document to calculate the total rank. They used a simple yahoo search engine to find the answers and then they manually analyzed the results/snippets of 50 questions from TREC 2006 [93] to extract the best results. However, finding the non-relevant text was a bit tricky. They parsed the question to extract the parts of speech (POS) tags from which they separated the topic (proper nouns or nouns) words from normal keywords. They used coverage (amount of questions for which the relevant answer was found in top K results) and redundancy (average number of answer within passages found amongst the top k passages retrieved per question) [170] as their evaluation metrics and found that the use of prior probabilities improves the performance of passage retrieval. Similarly, Keikha and Sanderson [108] employed language modeling approaches like query likelihood (QL) language model[220], Sequence Dependence Model(SDM) [139], Positional Model(PM)[35] and the interpolation of QL (combining score of passage with the score of the document) where each passage was treated as a document against the long informational queries(non-factoid) for passage retrieval. For passage retrieval all employed language models performed closer to each other, however, interpolation performed a bit better in terms of MAP and P@10 measure but it has no significant difference in the higher ranks. To evaluate the returned passage retrieval system against the long informational queries, they used a character-level (Passage2MAP) measure from TREC Genomics passage retrieval [93] task similar to R-Precision (inspect relevant characters from the passage answer-set) measure used in passage retrieval task in HARD track [4]. To measure the Mean Average Precision (MAP), the amount of total relevant characters from the retrieved answer set was used whereas they considered each character in respective passages as ranked documents. The limitation of these character-based measures is that for a document to be considered relevant, it requires annotation of exact characters, which is time-consuming and not feasible. Keikha and Croft [107] used the concept of a document summarizing as a measure to evaluate the passage level retrieval. As it has been shown previously [130] that summarization metrics perform well against human judgments. For evaluation, they used the summarizing evaluation metrics which measures the similarity between the retrieved/candidate passage and the ideal passage. With that way, they filtered the candidate passages that can be used for evaluation without doing any passage level annotation. They compared the results with both, the existing measures [4, 93], and also with human judgments. They used ROUGE [129] package for evaluating automatic summarization. For summarization, ROUGE has different measures. They used N-grams, skip bi-grams (pair of words in the same order and the distance between them is limited), and the combination of both. To compare the results with existing measures, they used character level MAP(Mean Average Precision) measure to find the correlation with ROUGE measure and observed that ROUGE-2 (i.e. bi-grams) is highly correlated with character-based measures. They also did another experiment where they annotated the passages manually in different categories and tried to find the correlation between the results from ROUGE measures and the ideal answers. The idea here was to see if ROUGE measures can differentiate the passages of different levels and the results showed that to a certain extent it can. For this experiment annotators used GOV2 collection that correlates with TREC queries.

In traditional document or passage retrieval systems, the performance is judged based on their likelihood of relevance to the query. It does not consider the diversity of the answer set while performing the ranking task. This creates problems in situations where certain aspects of the answers are missing. Retrieved passages even relevant to the query could have become outdated or multiple top answers could cover only a single aspect of the required answer. This problem is more visible in the medical science domain due to the diversification of aspects that could be related to a single concept e.g. disease, and operational procedures. In TREC Genomics track [93] Aspect Mean Average Precision was introduced to measure the relevance and diversity of ranked answer set. In that track, the diversity element of the answer set was judged manually and the top results were filtered accordingly. An automatic way of identifying the ranking diversity was introduced by Carbonell et al. [33] called Maximum Marginal Relevance (MMR). It measures relevance by considering the document (or passage) similarity with the query and dissimilarity of documents within the returned answer set.To measure the aspect, word co-occurrences are used. One drawback of measuring aspect by using the MMR approach was that it may be the case the same word is used in two different passages but different contexts. But since we are considering the word co-occurrence, the second document will be penalized because the system would consider two documents to be similar to each other, which is not true in this case. Another way to measure the aspect is to look at the topic drift of retrieved passages.

Latent Dirichlet Allocation

An LDA (Latent Dirichlet Allocation) based approach was used by Yan Chen et al. [39], where they calculated the topic distribution of the passages retrieved and the word distributions of every topic by using LDA and used it to re-rank the passages based on the topic distribution similarity among different passages of N size sliding window. While ranking the passages, they consider both the relevance and the diversity of a passage to reduce the redundancy of information provided in the collection of passages retrieved by the system. From the answer set, they first take out a passage with the largest topic coverage compared to the other ones and put it in an observed list(new ranked list). Then they choose the next passage from the remaining list based on the diversity of them to compare to the passages from the observed list and update the list one by one and repeat this process. They used TREC 2007 Genomics collection as their test data set and to calculate mean average precision

(MAP) for evaluation and showed that topic modeling approach achieved better performance than the highest aspect MAP outlined in TREC Genomics track.

3.2.3 Passage Retrieval Using Semantic Knowledge and Query Augmentation

The way documents and queries are written and expressed in natural language makes it difficult for the retrieval system to interpret the query accurately due to its dependency on the effective communication by the user for the intended purpose [193]. Similarly, there are words in different domains that might have a different meaning in a different scenario, e.g. the word 'crane'can be used for a bird or the machine. Similarly, while retrieving a document from a specific domain may require certain domain knowledge in advance to understand the query and their relation with the document in a better way. Therefore, external domain knowledge-based resources have proven useful to augment the information that is already provided via text from the document [132, 31]. The original HAL algorithm was proposed by Lund and Burgess in a theoretical analysis on the concept of capturing interdependencies in terms [136]. In this work they applied a window of 10 to their document corpus and measured the co-occurrence of terms. The value of this weight decrementing as it moved away from the target value. Their three applications of the method were: One) they took the nearest neighbours and determined euclidean distance between them to evaluate how correct their intuition was. Two) they ascertained that categorical terms can be ascertained from the method. Three) the got related terms and non related ones. Jumbled them up and see if people would manually give the same distance from a fixed point that they had automatically mapped out. Similarly, Song et al. implemented HAL to capture three level cognitive model [29]. They use a fixed window size of 6 in their experiments. They propose an augmentation of HAL that normalises the values. They apply the following formula to each word $w_{c_i p_j} = \frac{w_{c_i p_j}}{\sqrt{\sum_k w_{c_i p_k}}^2}$ and use it to determine the most appropriate term. We implemented this approach (discussed in Chapter 6) to further investigate their findings and test its applicability to query expansion.

For passage level retrieval, Wei Zhou et al. [240] used the domain-specific knowledge in the biomedical literature (information about concepts and their relationships in a certain domain) to improve the effectiveness of an IR system. They used the Genomics Track of 2006 TREC document collection that contains 162,529 full-text docs in HTML format [93]. The passages were described as any text that does not contain the <p> </p> tag. So it can be a sentence, multiple sentences, or a paragraph itself. To extract the medical terms from the query, they used PubMed query translation functionality. To Collect Synonyms, Hypernyms, and Hyponyms as a part of domain-specific knowledge, they used MeSH(Medical Subject Headings) [1] and Entrez Gene database [67] to find the information and relationships between genes and different concepts used in the query. They also collected the lexical variants (different words, same concept i.e abbreviations) by using a heuristic approach (expanding the word by putting spaces, and changing numbers format) or by using ADAM(abbreviation database) [239]. For passage extraction, their approach assumes that the optimal passage(s) could be the one that has all the query concepts in it and it should have minimal sentences from that paragraph. Also, compared to the segments of texts, the selected passages have a higher density of query concepts. Additionally, they used a variation of pseudo-feedback approach given by Yates et al. [9] to add new terms in the query. The results show that utilizing the information from the domain knowledge leads to significant improvements.

Similarly, VinhTuan Thai et al. [85] used ontology-based information extraction (OBIE) techniques to include domain-specific knowledge in Business Intelligence area. At first, they modeled the domain-specific knowledge by producing the conceptual model (defining and identifying the key concepts and their relationships and finding the unambiguous terms) and then created the ontology (OWL) by using a tool called SWOOP [103]. To index the document, Apache Lucene [19] was used; where they divided the information into two attributes. In the 'content'attribute, they indexed the passage text. Since the report/document text is in HTML format, they used the HTML markups along with some language engineering approach (which is not explained in the paper) to identify the passage boundaries. Once the passage boundaries are identified, they used Gazetteer[2] list from the GATE library to find and annotate the entities to their associated concepts. Similarly, for complex phrases where the gazetteers list does not work, they used JAPE (java annotation patterns engine) [207] rules. These associated annotations were indexed along with the passage in 'Annotations 'field.

In natural language, the same word will often be used to confer different meanings, and when presented in different contexts can embody different concepts. Term mismatch between the query and relevant documents tend to reduce the performance of an IR system. Query Expansion (QE) approaches are useful to fill the term mismatch gap by adding new terms to the original query. Blind relevance feedback (BRF) [145, 184] is an approach that is effective for automatic query expansion at the document and passage level. The motivation behind relevant feedback is to utilize the explicit relevance judgments provided by users to refine the query model and further retrieve more relevant results. Previously, it is shown

[1]https://www.ncbi.nlm.nih.gov/mesh
[2]https://gate.ac.uk/sale/tao/splitch13.html

that taking passages from the relevant documents for query expansion can be more effective than the document itself [80, 144, 135, 62]. The relevance feedback information needs to be interpolated with the original query to improve retrieval performance, such as the well-known Rocchio algorithm [175]. In some situations, BRF considers the top-ranked documents to be relevant but it may be the case that there is a topic drift in those ranked documents which could make few of them non-relevant in true sense. To avoid this topic shift passages instead of the whole document can be used which could avoid the addition of noisy terms to the original query [74].

Different query expansion approaches have been used in the past where the information extracted from passages was employed. Ferhat et al. [8] used two different methods to expand their queries for passage retrieval in the biomedical domain that can help to identify the protein interaction (PPI). At first, they used a supervised approach which uses the combination of term frequency-relevance frequency to identify the added terms. They subsequently used an unsupervised approach where they used a medical ontology to get the expanded terms. Similarly, Ganguly et al. [74] took passages in the form of a sentence and use them to expand the original query. They picked sentences from the relevant segment of text and ranked them based on their similarity with the query. Once ranked, they took out the top-ranked sentence and use it as a whole to expand the original query to broaden the context around the query. They showed that their sentenced based query expansion approach for language models outperform the previously used language models relevance feedback approaches[137, 205, 30]. Zhang et al. [234] endeavour to improve upon search results using meta data found within the corpus. Similar to our approach, they chose to use a graph based approach, however their work focused on using metadata found within the word groupings of the text. Specifically they designate two features found within the documents as indicators of how to rank the documents; *information richness* and *diversity*. Information richness is the extent in which a document relates to a particular topic, and diversity is the sum of topics found within the corpus. In addition to determining these scores for the documents, they assign each document to a graph where the node represents the document and the surrounding nodes is determined by the inter similarity of the documents. The overall score of the document is determined by combining the information richness with the diversity. The diversity is calculated in relation to the query input. They use this final score to inform how the documents should be ranked. Experiments were performed on three datasets and the top 50 best results were extracted. The datasests consisted of Yahoo! directory, ODP data and Newsgroup posts. Through this approach they improved the overall ranking of information gain and diversity by 12% and 31% respectively. Additional examples of using information found in metadata would be the use of part of speech tags in selecting the appropriate terms.

Recently Croft [18] used incremental relevant feedback (IRF) [1] where in contrast to normal relevant feedback, after collecting user feedback on a small number of interactions, documents are re-ranked iteratively rather than gathering a lot of feedback and perform the retrieval refinement in one go. They compared their model with the standard top-k relevance feedback framework and evaluate their system on two search tasks. One was ad-hoc document retrieval and the other was to find the relevant passage to the given query. They stated that with the few iterations better results were found due to iterative re-ranking. However, they experience the missing of a few relevant topics from the results due to less feedback provided in each iteration. The results for passage retrieval were more convincing than the document retrieval. Since the passages are smaller than the documents, it reduced the topic drift issue that occurred in the document retrieval task. To represent document vocabulary and capture context of a word in a document, Word embedding approach has been adopted in the past.It has become very popular lately for various passage retrieval tasks [71, 232, 43, 181, 49]. P. Arora et al. [6] presented a query expansion approach for sentence (passage) retrieval on the answer passage retrieval task [108, 228]. Similar to our work (described in section 6.3) they also utilized word embeddings (word2vec ([141])) and Pseudo Relevance Feedback (PRF) as stand alone approach and also as combined to expand the query terms. For PRF they used the Roberstson's standard Okapi QE relevance feedback approach [172] where as we employed Rochio Algorithm [175] for query expansion task. For word2vec they generated the embedding by combining all the documents as a corpus and also utilized the externally trained embedding on Google news. They have shown that the word2vec approach increased the performance over the baseline but it wasn't significant. The best results were formed when the PRF and word2vec approach was combined together.

3.2.4 Document-Passage Graphs for Ad-hoc Retrieval

Previously, graphs have been used to represent text for ad-hoc information retrieval [20, 152, 65, 208]. The formulation of the weighting schemes to rank documents and summarize text by using graphs has also been studied lately [20, 212, 180, 61, 206, 138]. Graph-based approaches like *PageRank* [157] and *HITS* [111] have been widely employed for ranking the top web pages, analysis of social networks, as well as for ad-hoc document retrieval [120, 119] purposes. For passage retrieval, different graph-based approaches have been used previously. Li et al. [128] proposed a graph-based ranking model that measures the relationship between passages and uses it to re-rank the passage results in question answering (QA) task [52]. They constructed the graph after the initial standard retrieval against a query, and then they re-ranked the returned passages based on a relationship of different passages/vertices. The strength of each edge between vertices relies on the keyword matching distance and the success rate of that matching which is inspired by the work of Monz [146]. Otterbacher et al. [153] used a variation of a graph-based ranking model called LexRank [61] to rank the set of sentences for the generation of a document summary. They applied this approach in the context of passage retrieval for the Question Answering task. They calculated the tf-idf of all the sentences in the documents and use that to build the graph. Then by looking at the high scoring sentences, they looked at their neighbor sentences with the assumption that it should also have a higher score. And finally, they combined the high scoring sentences together to form a summary. Similarly, Dkaki [58] presented a model based on graph comparison for passage retrieval task. Their graph model considered the sentence dependencies by following the HITS algorithm [111] or PageRank [23]. However, they did not consider the explicit links between the documents by using hyperlinks or citations. Instead of using the implicit inter-document relationship [96] based on the cosine similarity, they have utilized the approach the identify the linkage between units/sentences based on related terms that are shared among themselves. Although their model helped in improving the precision of the system, they highlighted some drawbacks in terms of the computational complexity of generating their recursive graph. They proposed MAC/FAC [68] architecture, which can use to reduce the complexity by providing restricted selection and filtration to eliminate the unnecessary document. Another passage-graph approach was employed by Bendersky et al. [12] to improve the document ranking. While most work on passage-based document retrieval ranks a document based on the query similarity of its constituent passages, their approach leveraged information about the centrality of the document passages concerning the initial document list. They generated the initial document list by identifying the top 50 relevant documents to each query based on the similarity score $sim(q,d)$. They hypothesized that the passages similar to many documents in the initial list contain information that pertains to the query due to the virtue by which the list was created. In contrast to our approach mentioned in section 5.1 where we build the graph from all passages within the documents, they utilized the passage evidences from the answer set. They introduced a one-way bipartite graph G in which an edge with a non-zero weight connects document d in the initial list with the passages that are most similar to d in the initial list with the passages that are most similar to d. Once the graph is generated, they measured the centrality of a passage by simply adding the edge weights of all documents that are connected to a respective passage, or they used the HITS score for each passage. Their approach outperformed the baseline and other commonly passage-based approaches like Max passage and interpolation technique [32, 135]. Similarly, Krikon et al. [120] used the same graph approach and presented a language-model that can be used to re-rank the answer set. Their model considers the inter-passage similarity (central passage) based on the initial document list and also evaluates inter-document (central document) for a given query. The results showed that their graph-based approach beats the other approaches that use PageRank or HITS methodology. By taking only the passages from the initial list of documents and using it as a bipartite graph, a relevant passage could be penalized more if an off-topic document is a list that could have a relevant passage pertains to the query. However, by considering the inter-passage similarity graph (same as used by us in section 5.1), that passage will still be related to other related passages from the graph and will get a higher boost and can go up in ranking compared to the bipartite document passage graph approach.

More recently, Sheetrit et al. tested the cluster hypothesis [118, 217] by using the documents as well as inter-passage similarities [199]. They used the nearest neighbour (k) similar to the approach we used in this thesis to find the most similar passage and documents. They have shown that the cluster hypothesis not only holds for documents but also for passage, which supports our motivation to utilize the inter-passage similarity in a graph space. Later on, Sheetrit et al. introduced a clustering-based approach that also uses inter-passage similarity for focused retrieval [197]. Unlike using the passages to improve the document ranking (which we are proposing in this thesis), they used learning to rank [198] approach to rank the passages from each document based on their relevance to the query for passage retrieval task.

3.2.5 Passage Utilization for Query Performance Prediction (QPP)

For the Query Performance Prediction task, the main objective is to estimate the effectiveness of retrieved results against a given query without any relevance judgements [36, 165]. The main body of work in QPP is divided into two common areas. Pre-retrieval predictors consider query and the corpus information before the retrieval process [45, 88, 89, 147, 236]. Post-retrieval predictors also include the ranked result list of retrieval process [64, 241, 201, 46, 200, 233, 216]. Our approach falls under the post-retrieval predictors. Major work in QPP post-retrieval work has been more focused on using the document retrieval evidence and very little work has been done to use the passage based information for the QPP task.

Roitman [177] used passage information for QPP task. Similar to our work, they also hypothesized that the passage evidence from a retrieved results can provide a valuable evidence with respect to the effectiveness of the retrieval. They used the max passage approach [32] to obtain the best passages from the retrieved document list. However, rather than retrieving documents (and extracting passage information from them), we index and retrieved passages as answer-set to measure the effectiveness of retrieval because the top ranked results in the answer-set provide better contextual information around the topic of the query. Moreover, A. Khwileh et al. [110] utilized QPP to select the suitable passage type against each query. They also introduced a new QPP approach called Weighed Expansion Gain (WEG) and compared it with existing approaches like WIG [241] and NQC [201] to select the best passage based evidence approach [135, 32, 91] for a given query. Once the most suited passage is selected, they formulated an adaptive Query Expansion (QE) technique to improve the performance of the system. Semantic-based post retrieval predictors have also been presented in the past for QPP task [40, 109, 101]. Recently, Jafarzadeh et al. [101] proposed a semantic-based approach that utilizes the graph model which exploits the topological features (represented by document entities) to capture the semantic similarities between a given query and top-returned documents. We used cohesion of the answer-set based on the retrieved passages. They employed the similar approach to measure the cohesion of the answer-set but they considered documents and not the passages. Also, their approach to measure the similarity between the top returned documents was build upon a graph-based semantic model and not a traditional approach like cosine similarity. Other than cohesion, they also introduced a semantic based query-drift predictor (similar to the model presented by Shtok et al. [201]). They have shown that their proposed predictors were effective compared to other existing QPP approaches. For question answer (QA) task, Li et al. [128] introduced a graph-based model to re-rank the retrieved results (passages) by using their similarity with each other. We generated the passage graph in a similar way and then apply edge count threshold on it to measure different features of the graph for QPP task. As our intuition of utilizing the similarity between passage is motivated by the cluster hypothesis, a learning to rank approach based on a similar intuition has recently been employed by Eilon et al. [197] for passage retrieval task. They have also shown that cluster hypothesis holds for returned documents as well as for returned passages against a given query.

3.3 Summary

This chapter presented an overview of work relevant to ours within the passage retrieval domain. Both passage selection and passage manipulation approach to developing a passage-based Information Retrieval system have been discussed. At first, we presented relevant work done in the past related to the three main categories of choosing the passage boundaries like discourse (bounded) passages, window passages, and semantic passages. We then presented a summary of research efforts that attempt to use the passage evidence to improve the ad-hoc retrieval, question answering tasks, or text summarizing based systems. Furthermore, intending to explore further how passages can be used to identify the context around the text, we presented details of research related to the use of passages for semantic knowledge domain, query expansion tasks, and finding the similarity between passages via graph-based

Table 3.1 Summary of Literature in Passage Retrieval

Methodology	Application Domain	IR Models and Techniques	Type	References
Discourse Passages	Ad Hoc Retrieval Passage Retrieval Question Answering	Vector Space Query Likelihood	Pre-Processing	[32, 230, 135] .
Window Passages	Ad Hoc Retrieval Passage Retrieval Question Answering Document Segmentation Element Retrieval	Vector Space Kullback-Leibler Query Likelihood	Pre-Processing	[135, 106, 42, 210, 99] .
Semantic Passages	Ad Hoc Retrieval Passage Retrieval Question Answering Graph Generation	Vector Space LDA Query Likelihood	Pre-Processing	[91, 221, 169, 21] .
Passage for Re-rankings	Ad Hoc Retrieval Passage Retrieval Question Answering Query Expansion WebSearch Learning to Rank QPP Text Summerization	Vector Space LDA Query Likelihood Contextual Embedding (BERT) HITS	Post- Processing	[107, 130, 235, 210, 150, 32, 75, 30] [167, 3, 13, 115, 120, 127, 2, 198, 133, 48, 149, 102] [235, 72, 154, 37, 108, 93, 39]
QE with Passages	Ad Hoc Retrieval Passage Retrieval Question Answering	Vector Space Pseudo Relevance Feedback Incremental Relevance Feedback (IRP) Contextual Embedding (BERT) Word-Embeddings Ontology based Information Extraction (OBIE)	Post- Processing	[29, 240, 9, 85, 135, 62, 8, 74, 234, 18, 6, 49]
Passage Graph in IR	Ad Hoc Retrieval Passage Retrieval Question Answering Text Summerization Clustering	Vector Space LexRank HITS PageRank	Post- Processing	[128, 146, 153, 58, 12, 120, 199, 197]
Passages for QPP	Ad Hoc Retrieval Passage Retrieval Question Answering	Vector Space WIG NQC Standard Deviation WEG	Post- Processing	[177, 110, 101, 201, 128, 197]

approaches that were used to improve the performance of an IR system. Lastly, we discussed the state of the art work done for Query Performance Prediction task and the different approaches that researchers have developed by utilizing the passage based evidence as the query performance predictors. Table 3.1 shows the summary of literature discussed in this thesis in the domain of Passage Level Retrieval. As shown in the table, the re-ranking of passages has been performed for many IR tasks. It is a well-studied area with extensive work done so far. However, if we consider the *QE* task, we did not find any noticeable work that utilizes passages-based *QE* for tasks like Text Summarization and Web Search. Similarly, using passages to visualize topically similar or different texts is another future avenue that is less explored so far. It would be interesting to study the existing clustering approach and form a system that can aid the user in finding relevant and diverse answers. We discussed one example of it in Section 8.2.5.

Chapter 4

Passage Based Evidence for Document Retrieval

This chapter presents the proposed framework that focuses on improving document ranking by using different passage based evidence. Each document is indexed as a series of unique passages. We explore and analyse a number of similarity measures which take into account the similarity between a query and a document at passage level with the aim of improving the quality of the answer set. We have also explored the notion of query difficulty to understand whether the best performing passage-based approach helps to improve the performance of certain queries. This work was published by Sarwar et al. [190, 191] which details the experiments conducted on two test collections (WebAp, Ohsumed) against different passage-based similarity measures.

The retrieval of relevant information from large collections is a difficult problem; search queries and documents are typically expressed in natural language which introduces many problems such as ambiguity caused by the presence of synonyms and abbreviations, and issues arising from the *vocabulary difference problem* which occurs when the user expresses their information need with terms different to those used to express the same concept in the document collection. In IR, traditional approaches consider the document as a single entity. However, we choose to split the document into separate passages given the intuition that a highly relevant passage may exist in a larger document which itself will be considered as non relevant [32]. Indexing passages individually as *pseudo-documents* has its limitations as well. For instance, when passages are considered (instead of a complete document), the number of documents stored and indexed will increase significantly and as a result, it will affect the speed and cost of retrieval [170]. Though improving the cost of retrieval is not the emphasis of our work, we used multiple nodes of Solr[1] as a baseline system, which is a high performance search server build using Apache Lucene core to reduce the retrieval cost.

In the following section, we first present passage evidence-based retrieval framework. This is followed by the experimental setup employed for the selected test collections and the evaluations performed to analyse the notion of query difficulty and also to measure the performance of the retrieval framework.

Passage Evidence Retrieval Framework

In this section, we will start with describing the notational conventions used in our framework, followed by the several passage based similarity functions that we used to attempt to improve the performance of the retrieval system. We then present different passage boundaries that we used for the tests collections.

4.1 Notational conventions

We used d,q,and p to denote a document, query, and passage. In this work, we view every document as being represented as passages or 'pseudo-documents' i.e. $d^{'} = \{p_1, p_2, \ldots p_n\}$. We attempt to better estimate $sim(d,q)$ by estimating $sim(d^{'},q)$. Different similarity functions are designed in a way that different characteristics of the passage level results can be used alone, or in combination with the document level results. We represent $sim(d^{'},q)$ as $f(sim(p_i,q), sim(d,q))$, where f is a mathematical function applied to the similarity scores, n denotes to the number of passages in d and N represents total number of documents. The function *sim* here denotes to the lucene similarity function (LVSM) between the query and a document (or passage) as specified in section 2.10.1.

4.2 Similarity Functions:

Following is a brief description of these similarity functions in which different characteristics were computed from the passage level evidence:

- $\{$**SF1**$\}$ Max Passage: One way to estimate the $sim(d^{'},q)$ is to consider the similarity and ranking of the passage that has the highest similarity score to the query as a

[1]http://lucene.apache.org/solr/5_2_1/index.html

representative of the similarity of the document.

$$sim(d^{'},q) = \max_{1 \leq i \leq n}(sim(p_i,q)) \tag{4.1}$$

- {**SF2**} Sum of passages: It is similar to the max passage approach, but instead of taking only the top passage, the top *n* passages are taken and their similarity scores are combined by adding them together.

$$sim(d^{'},q) = \sum_{i=1}^{n} sim(p_i,q) \tag{4.2}$$

- {**SF3**} Combination of document and passage similarity scores: In this case, the passage and document scores are combined and then the results are re-ranked based on the new score. For example, in the equation below, the passage score (max passage) is combined with a document score. An α and β scores are applied to incorporate a certain weight on the document and passage score.

$$sim(d^{'},q) = \alpha(\max_{1 \leq i \leq n}(sim(p_i,q))) + \beta(sim(d,q)) \tag{4.3}$$

- {**SF4**} Inverse of rank: Rather than using the document or passage scores, the rank at which these passages are returned can also be used to estimate the similarity between the passages and the query. This can be calculated as follows:

$$sim(d^{'},q) = \frac{\sum_{i=1}^{n} \frac{1}{rankP_i}}{n} \tag{4.4}$$

- {**SF5**} Weighted Inverse of Rank: Another way to take the rank of these passages into account is to take the sum of the weighted inverse ranks. By applying weighed inverse rank, we penalize the passages with low ranks and elevate those with high ranks. Therefore, passages with higher ranks will have a greater impact on the outcomes than lower ranks and will alter the overall ranking. The value of the weight(γ) used for the experiment is specified in section 4.4

$$sim(d^{'},q) = \sum_{i=1}^{n} \frac{1}{(rankP_i)^{\gamma}} \quad |p_i \in d^{'}, \gamma > 1 \tag{4.5}$$

4.3 Passage Boundaries

We use two different test collections in the experiments. The characteristics of both the test collections and the overlapping window size (for passage boundary) is specified in table 2.2. We choose the half overlapping, fixed length window-size to index the documents, because these passages are more suitable computationally, convenient to use, and were proved to be very effective for document retrieval [32, 135]. At indexing level, we maintained other attributes with passages to keep the track of its original document. Once the documents are returned in the form of passages, as shown in figure 4.1, we transform them at the document level by combining all the passages to its parent document and pass this representation to the similarity function to re-rank the documents and compare the results with the normal document level representation. Figure 4.2 shows how passages are represented at the document level and re-ranked using any of the similarity functions specified in the previous section. In step one, passages are returned in a ranked list along with their similarity score to the query. In step two, all passages are represented on their parent document level where passages belonging to the same document will be joined together to form a new document. Finally, a ranking function is applied to the set of these new documents to assign a new similarity score (by using a score of all or some of the passages in that document) to each document and return the sorted list based on their final similarity score. Furthermore, Algorithm 1 shown below represent a pseducode of passage retirval, conversion of retrieved passages at a document level and how passage evidence function like MaxPassage is applied for re-ranking and evaluation of the returned results against a given query. For p passages retrieved from the d documents, time complexity of converting the passages into document level evidence can be denoted as $O(p \times d)$.Sorting is applied afterwards to re-rank the documents based on passage evidence score that has a time complexity of $(O(d \log d))$. So the overall time complexity of the aforementioned approach is $O(p \times d)$

4.4 Experimental Setup and Evaluation

In this section, we present details of different parameters that we consider in our experiments. Lastly we will describe the brief overview of the evaluation measures that we used in the experiments.

4.4.1 Parameters

For different similarity measure functions, we used different parameters. For sum of passages (SF2) and inverse rank (SF4) function, we set the n value to be equal to 5 and the results were

Algorithm 1 Retrieve and Evaluate Passage Evidence (Max Passage)

```
 1: Array Passages ← GETPASSAGESFROMSEARCHENGINE(Query)
 2: Array ReturnPassageDocuments ← []
 3: for Passage in Passages do
 4:     DocId ← Get Parent Document Id of a Passage
 5:     if ReturnPassageDocuments Contain DocId then
 6:         Doc ← ReturnPassageDocuments[DocId]
 7:         Include Passage in Doc
 8:         ReturnPassageDocuments[DocId] ← Doc
 9:     else
10:         Create new Doc
11:         Include Passage in Doc
        ReturnPassageDocuments[DocId] ← Doc
12: index=0
13: for Doc in ReturnPassageDocuments do
14:     for Passage in AllPassages ← Doc do
15:         Find Passage with Max Score
16:     Set Doc Score ← Max Score
17:     ReturnPassageDocuments[index] ← Doc
18:     index++
19: Sort ReturnPassageDocuments In Desc Order
20: Array OriginalDocuments ← GETDOCSFROMSEARCHENGINE(Query)
21: Array RelevantDocuments ← GETRELEVANTDOCS(Query)
22: EVALUATE(ReturnPassageDocuments,ReturnDocuments,RelevantDocuments)
```

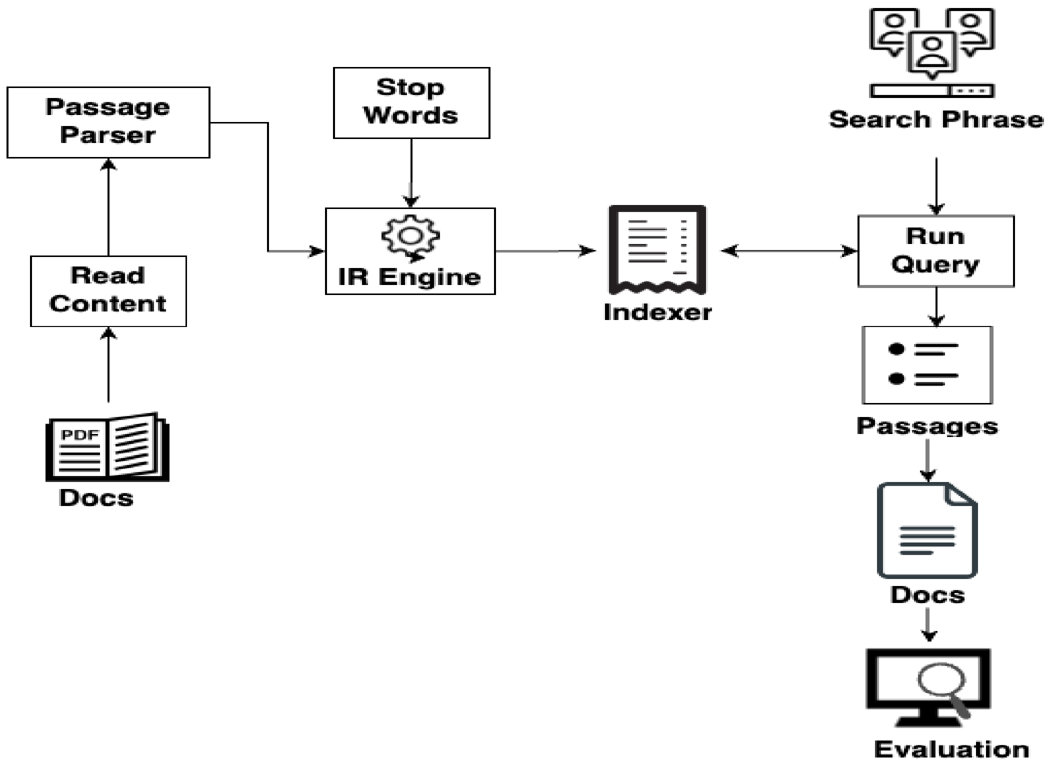


Fig. 4.1 Architectural Diagram.

normalized having received the final score. Furthermore, for $SF5$, we use the weight(γ) $\gamma = 2,3,4,5$ and reported the results with $\gamma = 2$ in this chapter, because of its high performance results compared to other variants. Similarly, we gave twice the boost to the passage level score as compared to the document level score while combining the results together i.e $\alpha = 2, \beta = 1$. For $SF3$, we did not perform any normalisation because we did not want to normalise the document level score with the α value. Also, the similarity scores were normalised already (i.e. less than 1) when the α and β score was applied. Therefore, applying extra normalisation based on boosting parameters would reduce the significance of the boost value. Also, it is to note that we are not using the actual similarity score in our evaluation. We use the ranking value based on the similarity score, and for ranking, the normalization is already implicitly applied. Our motivation for applying a higher value to α was to improve the ranking of a passage within a less relevant document. This ties back to our initial statement in this thesis that by using passages, we can boost a ranking of an irrelevant document that has a small relevant text to answer the query. Furthermore, giving the higher boost to passage level gives better performance to the inverse ranking functions in the WebAp collection, whereas the same settings against the Ohsumed Collection gave better results for Max passage and Sum of passage. One main reason for this difference is the nature

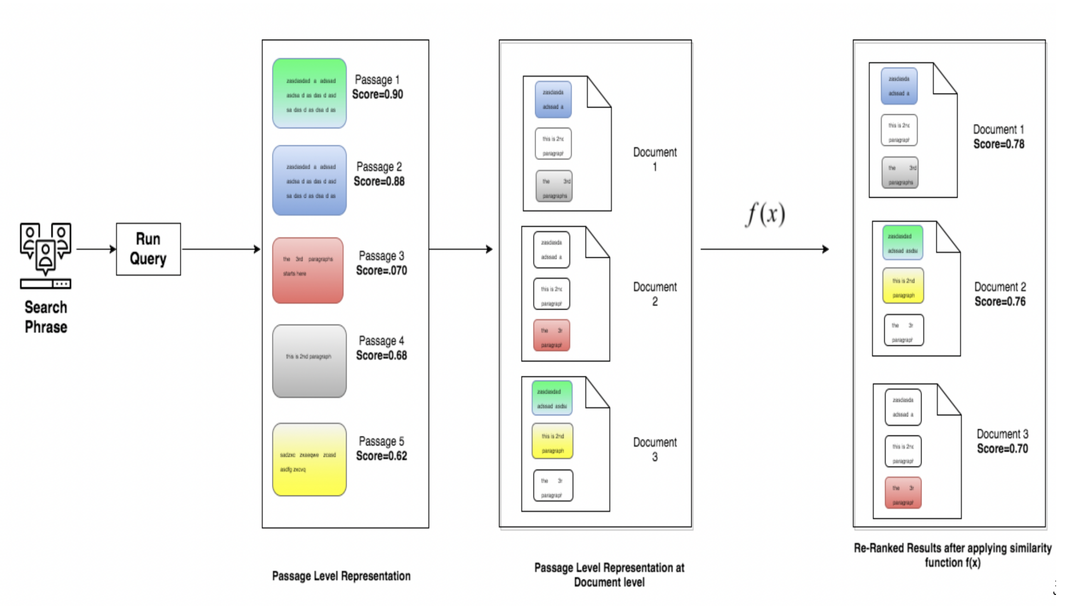


Fig. 4.2 Passage Level Representation at Document Level.

of both test collections. As the document length is small in the Ohsumed collection and the size of the test collection is large, the Max passage approach did not perform well on its own. Therefore, combining the passage evidence with the document level gave better results for the Ohsumed collection. The sensitivity analysis was also performed on a variation of α and β values as shown in Table 4.1 and Table 4.2 for the WebAp and Ohsumed collection.

4.4.2 Evaluation Measures

In IR, different evaluation measures are used to measure how well the system is performing to satisfy the user's need in returning the relevant documents to a given query. In our case, to measure the quality and performance of our approach, we used Mean Average Precision (MAP) and precision@k. MAP value is used to give an overall performance overview of the system and different similarity functions across both test collections. On the other hand, precision@k was helpful in illustrating the user's experience and the behavior of relevant documents returned in terms of their ranking frequency with the different threshold values. We evaluated the precision value for top 40 unique documents, both at passage level and at document level.

Table 4.1 Sensitivity Analysis on α and β value WebAp Collection

Similarity Functions	D+SF1	D+SF2	D+SF4	D+SF5
α=2, β=1	72.0	71.3	72.9	72.8
α=1, β=2	72.3	71.4	72.1	72.0
α=1, β=1	72.3	71.9	72.4	72.1
α=5, β=1	71.6	71.2	72.5	72.1
α=1, β=5	72.7	77.6	72.2	72.2

Table 4.2 Sensitivity Analysis on α and β value Ohsumed Collection

Similarity Functions	D+SF1	D+SF2	D+SF4	D+SF5
α=2, β=1	13.7	13.5	13.4	13.2
α=1, β=2	13.6	13.2	13.5	13.2
α=1, β=1	13.8	13.6	13.4	13.2
α=5, β=1	13.4	13.5	13.4	13.3
α=1, β=5	13.5	13.8	13.4	13.3

4.5 Results

In this section we present the experimental results to show the performance of the different similarity functions at passage level and at document level for both the WebAP and Ohsumed datasets. In Figure 4.3a and 4.3b, a bar chart is used depict the comparison of the document-level score with the different similarity functions of passage level scores for WebAP and Ohsumed test collections.

Using the WebAP collection, the results show that combining the document level score with inverse-rank based passage level score ($SF4$ and $SF5$) gives a performance improvement. The results show that, considering the rank of the documents instead of the similarity score gives better performance when document ranking is combined with the passage level evidence. For the sum of passages (SF2) approach, only the top 5 (i.e. n=5) results were considered in calculating the query similarity score.

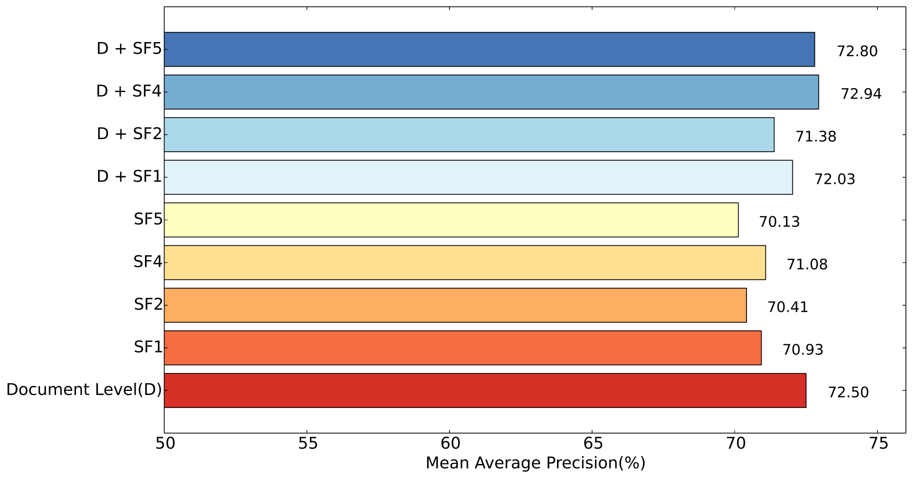
In contrast to WebAP, for the Oushmed collection the combination of document score with the max passage score performed better than the combination of inverse passage rank with document score. However, for functions not including the document level similarity, inverse rank by gamma (SF5) performed better than the other passage level similarity functions and give approximately similar performance in comparison to document level. Furthermore, the sum of passages (SF2) performed better here than the Max passage (SF1) score. The best results were observed for $n=2$. We have observed that the MAP values decrease as the k value increases, hence max passage similarity function performs better than the sum of passages function for WebAP test collection. However, in Ohsumed SF2 performed better than SF1 for $n = \{2,3,4\}$, which is due to the fact that a single passage size in Ohsumed collection is quite small. Therefore, combining passages together gives better representation of the the answer-set with respect to a given query.

It is to note that the $SF4$ gave the lowest performance (MAP) for the Ohsumed collection. It is because for most of the queries, very few relevant passages are returned in the top results; and when the inverse rank is applied, passages from irrelevant documents form a higher ranking score than the relevant documents. We saw that penalizing the lower rank passages and boosting the higher rank passages by using the $SF5$ resolved that issue to some extent and hence the performance was improved with $SF5$. However, the WebAp $SF5$ gave the lowest performance because unlike the Ohsumed collection (where only a few relevant passages are returned), the top passages were mostly from the relevant documents and the boosting or penalizing did not help much in increasing the performance. As a result, the performance reduced a bit because a few highly ranked irrelevant passages received a boost, which in result lowered the performance.

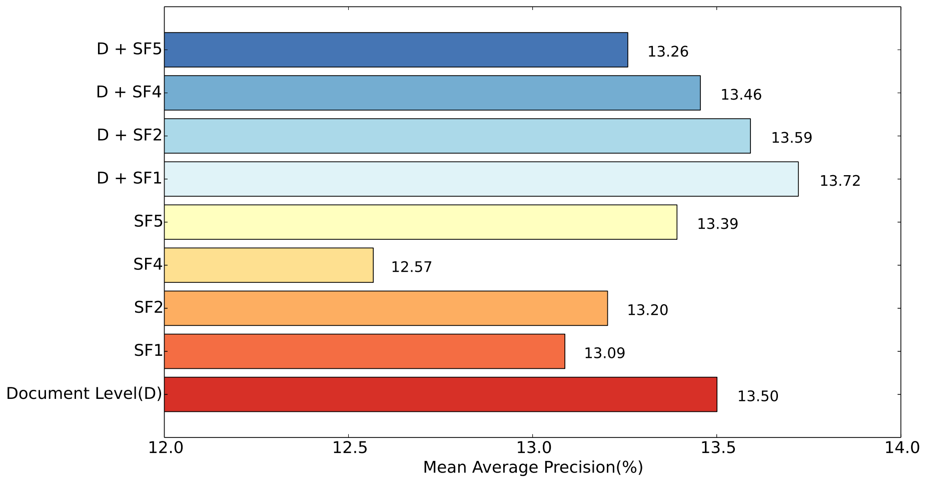
We also used precision@k as a different evaluation metric. The objective of this experiment was to check how well the documents are returned at the top k ranks at the document and passage level, and to measure on average how many relevant documents are returned at the different k values. Figure 4.4a and figure 4.4b illustrate the calculated precision values for WebAP test collection and the Ohsumed collection at document level as well as at passage level. At passage level we used SF4 and SF5 to measure the average precision for the WebAP and the Ohsumed, as when we considered it separately (without in conjunction with the document score), their performance was better than SF1 and SF2.

For the WebAP, the results show that the document level achieved better $p@k$ in comparison to SF4, and out of 40 documents, 33 of them are relevant in document level and 31 of them are relevant at the passage level when SF4 was used. On average, the precision value for document level and passage level was 90% and 86%. This indicates that the correct documents for all queries are clustered together or are closely related to each other and therefore, most of them are returned in top results, hence the high results.

For the Ohsumed collection, SF5 clearly outperformed the document level results and gave marginally better precision from the start to top 20 results (p@20) compared to the document level. However, for the higher values i.e. $k{>}20$, the document level and SF5 gave almost the similar performance. Out of 40 documents approximately 9 are relevant in document retrieval and 10 of them are relevant in passage retrieval by using the inverse rank by alpha function(SF5). The overall performance for the Ohsumed collection is fairly low and this could be partially due to the large size of the test collection, small document length



(a) WebAp Collection



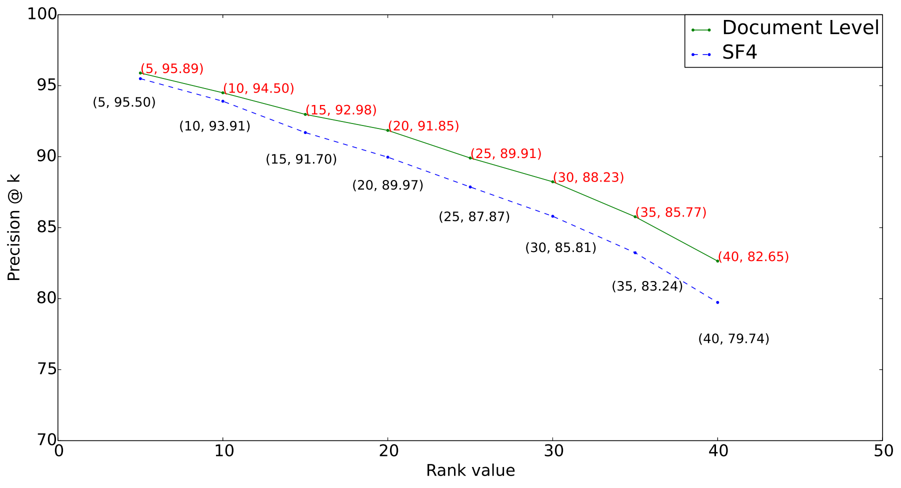
(b) Ohsumed Collection

Fig. 4.3 Mean Average Precision for Different Similarity Functions

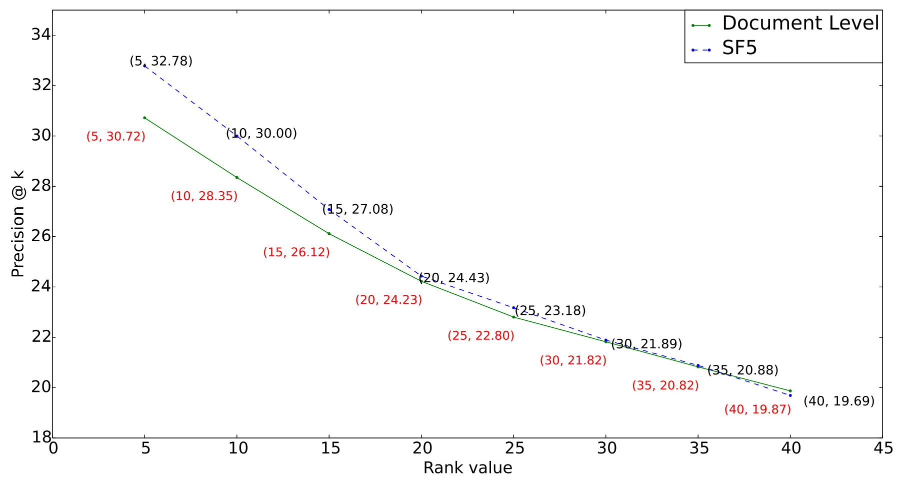
and the variation of relevant document information in relevance judgment file. On average, precision value for the document level and passage level was 24% and 25%.

Table 4.3 illustrates the mean average precision at top 5 (MAP@5) and at top 10 (MAP@10) for both test collections and as the results were discussed before, in the WebAP the combination of document level with passage level scores with different similarity functions give better results. The best results were obtained when the document score is combined with SF5. Whereas, for the Ohsumed, the functions that do not involve combining passage level and document level evidence gives better performance in both cases.

To get a better understanding on the statistical significance of the differences shown in the Table 4.3 for the test collections, we used the Student's t-test on paired samples for the



(a) WebAp Collection



(b) Ohsumed Collection

Fig. 4.4 Precision at K for Different Test Collections

top 50 MAP values with the difference of 5 (i.e. top 5, top 10, top 15, till top 50). For the WebAP, we compared the document level results with the D+SF5 similarity function as it gave an overall better performance on the top results. The average MAP difference between both experiments was 0.18 with the standard deviation of 0.09 and the calculated p-value was 0.00024. Therefore, the performance shown by D+SF5 is statistically significant as compared to the normal document level results. Similarly, we performed the same t-test on the Ohsumed collection by comparing the document level results with D+SF4 due to its advantage over the performance on normal document level results. For the Ohsumed, the average difference and standard deviation were 0.07 and 0.13 with the p-value of 0.069. Hence, for the Ohsumed, the results were not improved very significantly.

It is also seen that the value of α and β affects the overall results when the document level is combined with the passage level evidence (SF3). For both collections, giving the higher boost to passage level i.e. $\alpha > \beta$, gave a better performance for the inverse ranking functions, whereas a higher boost at document level i.e. $\alpha < \beta$ improves the results for SF1 and SF2. We chose $\alpha = 2$ and $\beta = 1$ for the results shown in this paper because it gives an overall better performance for all the passage level similarity functions when combined with the document score.

Table 4.3 MAP(%) For WebAp and Ohsumed Collection at n=5 and n=10

Similarity Functions	MAP@5(WebAP)	MAP@10(WebAP)	MAP@5(Ohsumed)	Map@10(Ohsumed)
Document Level(D)	9.52	18.60	2.97	4.75
Max Passage(SF1)	9.43	18.56	3.23	4.96
Sum of Passages(SF2)	9.42	18.54	3.19	**4.99**
Inverse of Rank(SF4)	9.42	18.56	**3.27**	4.89
Weighted Inverse of Rank(SF5)	9.43	18.58	3.20	4.98
D+SF1	9.53	18.65	3.01	4.90
D+SF2	9.53	**18.67**	2.82	4.74
D+SF4	9.54	18.66	2.88	4.80
D+SF5	**9.55**	**18.67**	2.80	4.60

4.5.1 Further Analysis of SF2

In the previous section, we used $n = 2$ to report the results for SF2 using the WebAp and the Oshumed test collection. We have seen that by varying the value of n, the average precision changes, which leads us to highlight the effect of changing the value of n in SF2 against the test collections used in this paper. To understand, and to determine how well the addition of passages performed in terms of improving the document ranking, we illustrate the behavior of SF2 at different n values. We report the results for $n = 1,2,3,4,5$, because considering the average size of the number of passages per document in both test collections, bigger k values i.e >5 did not demonstrate any improvement in performance. Figure 4.5a and 4.5b shows how the Mean Average Precision changes for different k values in both the test collections.

For the WebAP collection we have seen that the precision decreases with increasing k values. However, for the Ohsumed collections the best value is obtained for $n = 2$. This could be due to the number of passages per document in both collections. The WebAp has documents with the bigger document length, having more than 10 passages per document on average.

In the Ohsumed collection the document length is quite small with around 3-4 passages per document. Moreover, it is worth noting that by adding more passages together i.e., with the increase in the k value, we are losing the accuracy and adding more noise in the result set, which could be a cause of decrease in the MAP value. As shown in Figure 4.5a, the higher values of k are giving lower precision in both collections that supports our argument regarding the decrease in efficacy and the increase in noise.

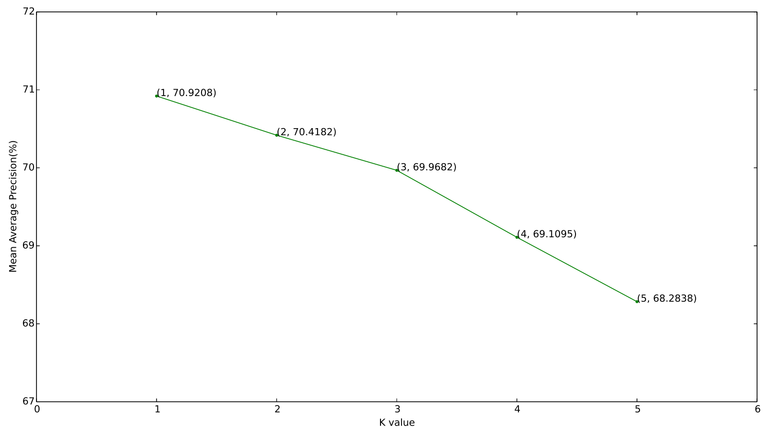
We have also performed the one sample t-test to check if the difference between the MAP at various k values is significant or not. For the WebAp collection, the p value<0.0001 and therefore, this difference is considered to be extremely statistically significant. Similarly, for the Ohsumed collection, the p values is also <0.0001, which makes the difference significant as well.

4.5.2 Query Difficulty

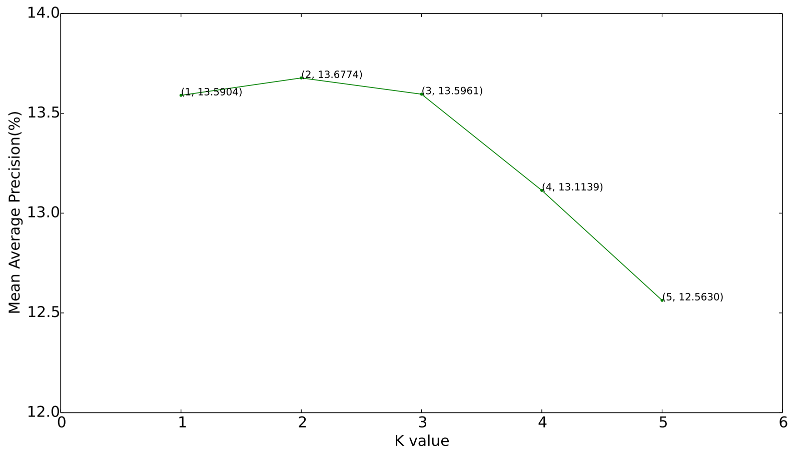
In this work, we have explored a number of passage level approaches and demonstrated in some cases, a modest, yet statistically significant improvement over the baseline adopting a classical document-level approach. However, it is not known if this improvement is due to a large number of slight improvements over a large range of queries or due to larger improvements over a small set of queries. Moreover, it is worth exploring if the best performing passage level approaches actually damage the performance for certain queries.

In this section, we present a query by query overview of the performance of the baseline and the best passage level ranking function in both the collections. In the WebAP collection, the SF4 approach performed best and in the Ohsumed collection, the SF5 approach gave the better performance when passage evidence is not combined with a document similarity score ($SF3$). We use these similarity functions to compare their impact on all the queries in their respective test collections. We identify the queries for which there is a substantial change in performance between the two and attempt to provide an explanation for this change.

Figure 4.6a and 4.6b illustrate the average precision across all queries. As shown in Figure 4.3a and 4.3b, we compare the baseline results with the similarity functions that was giving the better performance at passage level i.e. SF4 in the WebAp and the SF5 for the



(a) WebAp Collection



(b) Ohsumed Collection

Fig. 4.5 SF2 results for top 5 *n* values

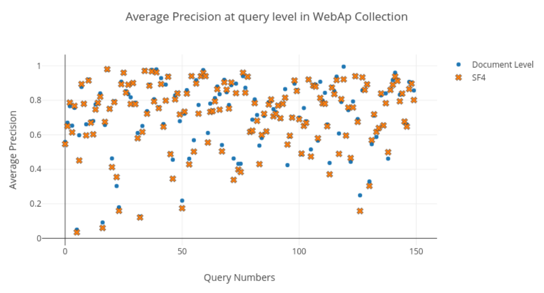
Ohsumed collection. In the WebAP collection, we saw that for the difficult queries (queries which performed worst), the document level was giving better performance than the SF4 approach. We also measured the query length of bottom 10 queries against document level and SF4 and didn't find any significant difference between them. On average the query length of the document level results and SF4 was 3.3 and 3.1 words per query. Moreover, for the WebAP collection, we have seen that the SF4 performed better because of the substantial improvements in a small subset of the queries (easy and difficult ones) and not due to large number of slight improvements over a set of queries. Of the 150 queries in total, in SF4 performed better than the baseline for 38 of them. It appears that, for the passage level technique (SF4), the worst-performing queries the performance was damaged slightly and the those that perform well (easy queries), the performance was boosted. For the poorly performing queries, the IR system has difficulty distinguishing the relevant from non-relevant documents (similar term frequency distributions). In incorporating passage level evidence we are possibly including evidence from weakly related passages. Rather than improving performance, we are merely hampering performance by incorporating information that does not improve our ability to make a useful similarity estimate.

For the Ohsumed collection, due to very low values of average precision, we considered the bottom 20 queries in order to get the better understanding of how well the difficult queries are performing against the document level and SF5. For difficult queries, SF5 gave better performance against the document level results and the average number of words per query noted for SF5 and document level was 6.5 and 6.4 words. Though the difference between them is not significant but SF5 slightly boosted the results for worst-performing queries. Among the total 96 queries in the Ohsumed collection, SF5 gave better accuracy for the 42 queries (including 25 difficult queries) compare to the document level results. Hence, we can say that the overall performance is increased due the small improvements in the large set of difficult queries.

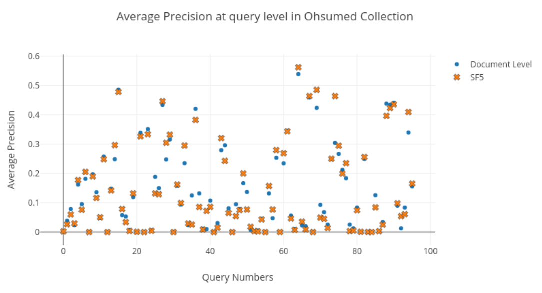
An overall better approach would be to attempt to identify in advance which queries are likely to be improved by the passage level augmentation. To this end, we attempt to identify differences between those queries that are benefited by the passage level and those whose performance is damaged.

4.6 Summary

The main focus of our work in this chapter was to describe how effectively the passage level evidence affected the document retrieval. We explored several similarity measures that can be used to improve the document ranking. Although we saw that the rank of a passage is



(a) WebAp Collection



(b) Ohsumed Collection

Fig. 4.6 Average Precision of each Query for Different Test Collections

an effective measure, it is not sufficient enough to improve the performance significantly without the interpolation of document level evidence ($SF3$). In addition to that, we undertook the detailed analysis of SF2 to understand its behavior on different n values. SF2 performed best when the value of n is smaller. For the WebAP collection, we notice that the precision decreases with increasing n values. However, for the Ohsumed collection, the best value is obtained for n=2. Moreover, we investigated the idea of query difficulty with regards to its impact on our rank-based passage functions. For the WebAp, we compared the baseline results with $SF4$, and $SF5$ was used in the Ohsumed collection due to its higher performance. Final results reveal that for the passage level technique, the difficult queries are damaged slightly and the those that perform well are boosted for the WebAp collection. However, for the Ohsumed collection, SF5 promoted the performance of the worst-performing queries. But the increase in performance was not statistically significant compared to the document level results.

Chapter 5

Passage Graph to Improve Document Ranking

5.1 Introduction

Finding the relevant information for a user's query is a difficult task and is considered complex because contextual information may be spread across the document. Approaches, where documents are represented as a series of passages, have been used in the past to attempt to improve the performance of ad-hoc retrieval systems. The intuition behind all these approaches is to present a document to the user that might seem to be off-topic, but a part of that document (passage) answers the user's query. One problem with these approaches is that since the amount of text is small and mostly comes from long documents, it is common to lose the context and the relationships between passages (of documents), which could potentially be used as evidence for re-ranking. In IR, different inter-document similarity measures [120, 15, 115, 7] were presented which are fundamentally based on the concept of cluster hypothesis [118, 217]. In the cluster hypothesis: "documents in the same cluster behave similarly with respect to relevance to information needs" [213]. Though inter-document similarities are useful, due to the occurrences of irrelevant snippets (passages) in a relevant document, it could affect the evaluation of similarity measure [199]. Therefore, rather than using the relevant document with irrelevant snippets, passages can be utilized [32, 108].

In this chapter, we represent the relationship between passages as a graph. The motivation behind using a graph as a construct to identify the relation between passages is to get a better notion of context around the text within the document and understand the structural and semantic information more effectively. Moreover, we adopted this approach because the strength of an edge can be defined flexibly within the graph. In this way, we can use the same graph but define the relationship between the nodes in several ways. Here we begin with an introduction of a novel graph-based approach that employs cohesion as a graph measure to understand how passages are linked to each other by using inter-passage similarities. Our approach correlates with the cluster hypothesis as we aim to check whether the passages from the relevant documents are connected closely to each other (hence more cohesive) than the non-relevant documents. We use cohesion as a metric to compare the relevant documents from the non-relevant ones. We then compared the cohesion based document ranking function with some of the passage-based ranking function described in chapter 4. We showed that document cohesion is an effective measure to improve the performance of an IR system. The discussion in this chapter is divided into two sections where at first, we took only the passage graph without considering the importance of each passage with the query. Later, we discussed some limitation of our cohesion approach and introduced a query-passage graph along with the previously employed cohesion graph to reflect the importance of each node in the graph and discuss the variation of results. Lastly, we will conclude the chapter with the discussion serving as proof into the usefulness of our proposed cohesion approach for document retrieval. The methodologies, experiments, and findings of this chapter are taken from the published work [192].

Graph Approach

In this work, we represent every document as a set of passages or 'pseudo-documents' i.e. $d^{'} = \{p_1, p_2, \dots p_n\}$. We use that representation to generate a weighted directed graph $G = (V, E)$ where each vertex p_i represents a passage that corresponds to $v_i \in V$ and E is a set of edges where the weight of an edge corresponds to an edge-weight function that is based on the similarity between passage nodes p_i, p_j. Figure 5.1 illustrates a high-level structure in which different passages from each document are connected to other passages. Below we describe our approach to use that graph model for calculating the cohesion score.

5.2 Measure of Document Cohesion Independent of Query

In this section, we will start with dividing the methodology in three different phases.

1. The graph approach at passage level and the definition of cohesion.

2. Introduction of cohesion based similarity functions.

3. Passage level division used to generate the graphs.

5.2.1 Cohesion Score:

To measure the cohesion score of each document, we consider the following two parameters.

1. Inter-connectivity of each passage i.e. passages connected to each other from the same document. For example, as shown in Figure 5.1, $p1$ from the document $D1$ is linked to $p3$ (denoted with plain arrow), and $p2$ from the document $D2$ (denoted with dotted arrow). To measure the inter-connectivity for $p1$ we only consider its linkage to the passages that belong to $D1$ i.e. $p2, p3, p4$.

2. Strength of edges between them i.e. the similarity score between each pair of passages p_i, p_j is denoted as $sim(p_i, p_j)$ in figure 5.1. In section 5.2.4, we will explain how the similarity is computed for the passages in the graph.

Let's assume that every vertex is connected to k neighbouring vertices, i.e. every vertex has an outer degree of size k in a graph, N is the total number of passages (from the same document), p_{j_i} corresponds to the passage j from the document i and n_{j_i} is a neighbouring node (inter-connected passages) for passage j in the graph from the same document d_i. C(d_i) denotes the value of the cohesiveness of document i. We use the following equation to calculate the cohesion score.

$$C(d_i) = \frac{1}{N(N-1)} \sum_{j:p_{j_i} \in d_i} \sum_{n_{j_i}} sim(p_{j_i}, n_{j_i}) \tag{5.1}$$

In this formula, we not only consider the inter-connectivity of passages but also take their positions/rank into account by adding the similarity score of the neighbour's nodes that belong to the same document. Therefore, the higher the rank, the higher the similarity score. Besides that, by taking the similarity score into account, we are considering the strength of the vertex/passage with its neighbours. The above formula first calculates the cumulative similarity score of a passage p_j in document D_i with the other passages based on their inter-connectivity and then sums up this score for all the passages in D_i to calculate the final score. The score is normalized by the inter-connected graph size, i.e. the maximum number of edges/relationships each passage can have with the other passages from the same document, i.e. $(N(N-1))$. Figure 5.2 illustrate the cohesion score of two documents $d1$ and $d2$. It is to note that the dotted edge between the nodes of $d1$ and $d2$ has a weight of zero because to calculate cohesion, only the nodes from the same documents were considered.

5.2.2 Similarity functions

To boost the ranking of documents, we created similarity functions based on the cohesion score. The motivation behind this is to investigate whether by boosting the cohesive documents, does the more relevant documents end up higher in the ranking. Our hypothesis here is that the relevant documents tend to be more cohesive than the non-relevant documents. If that is true, then by boosting a cohesive document in ranking, we could improve the performance. The following is a brief description of cohesion based similarity functions.

- {**CSF1**} One way to compute the $sim(d_i, q)$ is to multiply the similarity score of the document and the query by the cohesion score of the same document. By using multiplication, we enforce a document to be up in ranking if it is highly similar to the query and the document is cohesive at the same time.

$$sim(d_i, q) = sim(d_i, q) \times C(d_i) \tag{5.2}$$

- {**CSF2**} Instead of using the multiplicative expression, we can also use the additive way of boosting; by simply adding the cohesion score with the normal similarity score.

$$sim(d_i, q) = sim(d_i, q) + C(d_i) \tag{5.3}$$

- {**CSF3**} One limitation of $CSF2$ and $CSF1$ is that simple addition or multiplication may downgrade the overall similarity score because in some situations the cohesion score is higher for a document compared to its similarity score with the query q. By using the conventional boosting, a non-relevant document with higher cohesion score will get boosted higher in raking, which will lead toward the lower performance of the system. Therefore, instead of using the simple addition, a better way is to add only a X ratio (e.g. 10%, 20%) of cohesion score with the document similarity score. To report our results in this paper, we used X=0.1 as the best results were produced with this value. We did not perform normalization because the $sim(d_i, q)$ score and $C(d_i)$ score is already normalized and ranges between 0-1. Furthermore, as we only applied a ratio of the cohesion score to the document-query similarity score, normalization will reduce the effect of the cohesion score even more.

$$sim(d_i, q) = sim(d_i, q) + (C(d_i) \times X) \mid X = 0.1, 0.2, 0.3 \tag{5.4}$$

- To compare the performance, we also considered a Max passage approach [32, 13, 190] that has been commonly used to re-rank the document based on passage base evidence. Below equation is the same $SF1$ used in the previous chapter. We used $SF1$ as one of the baseline result in this chapter.

$$sim(d^{'},q) = \max_{1 \leq i \leq n}(sim(p_i,q))$$

5.2.3 Passage Level Division

In order to subdivide the documents into passages, we adopted the half overlapping, fixed-length window-size to index the documents, because in the literature these passages are found to be more suitable computationally, easier to use, and have been shown to be very effective for document retrieval [32, 135].

The characteristics of the employed test collections (Webap, Cranfield, and Ohsumed) in our work is specified in Table 2.2. Furthermore, only queries that have relevant documents associated with them were used to measure the performance, e.g. in Ohsumed collection, out of 106 queries in total, 97 of them were found to have the relevant document(s) associated with it.

5.2.4 Assumptions and experimental parameters

To measure the similarity between passages $sim(p_i, p_j)$ in our graph, we sent each passage p_i as a query to SOLR index for their respective test collection and retrieved the top k results. SOLR uses something similar to cosine similarity, but not quite the same. Therefore, the similarity $sim(p_i, p_j)$ and $sim(p_j, p_i)$ are slightly different (shown in Figure 5.2) due to other boosting mechanisms used in Lucene score e.g document normalization, and ‘coord boost’ which is based on query terms. However, the difference between similarity scores has no serious effect on our cohesion similarity score. Because as shown in Equation 5.1 we calculate the similarity $sim(pj_i, nj_i)$ and $sim(nj_i, pj_i)$ (provided p and n belong to same document) and we take average of it by dividing it with N which produce a normalized score. Furthermore, even if the similarity $sim(pj_i, nj_i)$ and $sim(nj_i, pj_i)$ are the same (e.g. by using standard cosine similarity), the normalization done in our proposed cohesion equation is generic and valid for any directed weighted graph with more than one vertex (passage in our case). For a document d with p passages, time complexity of calculating the cohesion score (C(d)) can be denoted as $O(p^2)$). Additionally, for documents that have only one passage, the cohesion score is not computed and therefore, the score of that document was not boosted

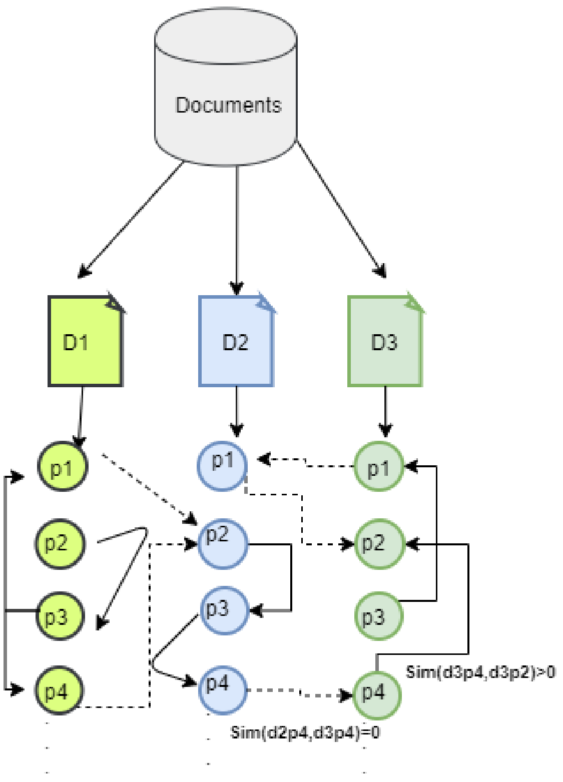


Fig. 5.1 High Level Graph Structure of Passages Nodes at Document Level

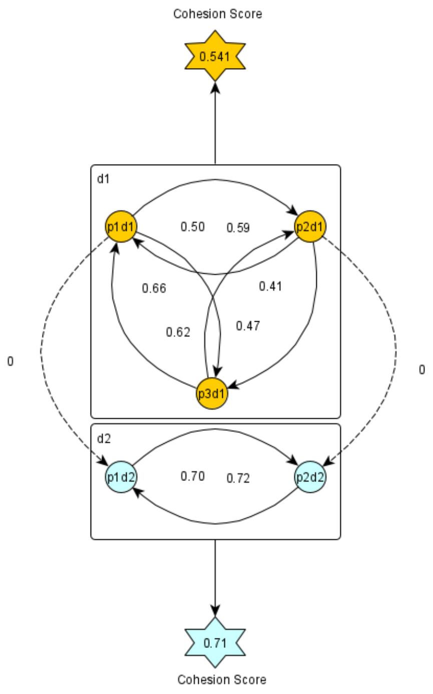


Fig. 5.2 Cohesion Graph of two documents

by the cohesion-based similarity functions. Furthermore, to generate graphs, we choose a different neighbour size k as the length of each document varies in all the collections; We choose $k = 30$ for the WebAp and $k = 10$ for the Ohsumed and the Cranfield collection. On average, a document contains only 6-7 passages in the small collections; therefore, we have chosen a smaller number for the graph neighbour size. Similarly, for WebAp each document contained between 25-30 passages and therefore we choose a higher number for the WebAp.

5.2.5 Experiments and Results

We have two major hypotheses for the experiments.

1. We hypothesized that there is a significant difference in the cohesiveness scores of R and NR sets for any given query. Here we wanted to investigate whether or not the relevant documents are more cohesive.

2. If the relevant documents are more cohesive, we suspect that the cohesion score can be an effective measure to improve the performance of the system.

We divide our discussion of experiments by explaining the results of both hypotheses in the following subsections.

Cohesion Score for relevant and non-relevant Documents

In this section, we present the experimental results to illustrate the difference between R and NR documents based on their cohesion score. We use Equation 5.1 to calculate the cohesion score for each document. We have calculated the cohesion score for R and NR set against each query separately to understand how differently the score behaves and to check if there is a significant difference between the cohesion scores of both sets for the given test collections. The overall results from this experiment will give us a better indication of distribution of the cohesive documents in the result set against a given query, and is useful to compare the R and NR set when passage evidence is applied. Figure 5.3,5.4, and 5.5 illustrate the average cohesion score of both relevant and non-relevant sets of each query in the form of a line plot. As seen in figure 5.3, 5.4, the relevant documents exhibit higher cohesion on average for most of the queries. For the Ohsumed collection (figure 5.5), NR has slightly better cohesion on average. Similarly, we used two-tailed student t-test at a confidence level of 95% to determine whether the difference between relevant and non-relevant set is significant or not. As shown in Table 5.1, at the query level for all test collections, there was a significant difference between R and NR documents, which support our first hypothesis that there is a significant difference between both sets. For the WebAp, and the Cranfield, relevant documents were

Table 5.1 Cohesion Score Statistics for R and NR

	Avg Cohesion for R	Avg Cohesion for NR	T-Val	P-Val
WebAp	**0.19**	0.14	8.74	< 0.05
Cranfield	**0.27**	0.24	4.23	< 0.05
Ohsumed	0.230	**0.242**	-2.3	< 0.05

found to be more cohesive and vice versa for the Ohsumed. As the length of each document in Ohsumed is small, the cohesion graph doesn't provide much evidence to differentiate the R and NR set and gave better cohesion indication for the collections that were bigger in length (WebAP, cranfield).

Table 5.2 Comparison of Similarity Functions and Baseline($LVSM$)

	Cranfield			WebAP			Ohsumed		
	MRR	P@5	p@10	MRR	P@5	p@10	MRR	P@5	p@10
$LVSM$	0.75	40.0	27.8	0.97	95.0	93.8	**0.49**	30.7	**28.3**
$CSF1$	0.46	22.2	16.7	0.72	73.2	77.6	0.30	18.5	16.4
$CSF2$	0.74	34.0	23.4	0.93	89.5	87.5	0.40	24.9	21.7
$CSF3$	**0.77**	**40.2**	**28.0**	**0.97**	**95.3**	**93.8**	0.48	**30.9**	27.4
$SF1$	0.70	36.2	25.5	0.96	94.9	92.0	0.48	30.5	27.5

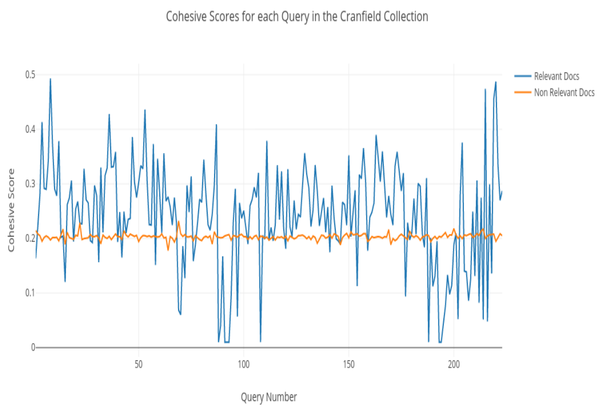


Fig. 5.3 Query Level Cohesion Score for Cranfield Collection

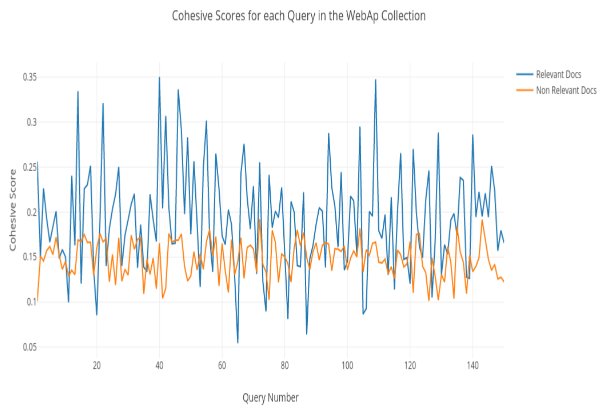


Fig. 5.4 Query Level Cohesion Score for WebAp Collection

Effects of Cohesion Score on the Document Ranking

In section 5.2.5, we discussed the experiments based on the cohesion score for the relevant and non-relevant documents. We also explored how average cohesion for each query differs

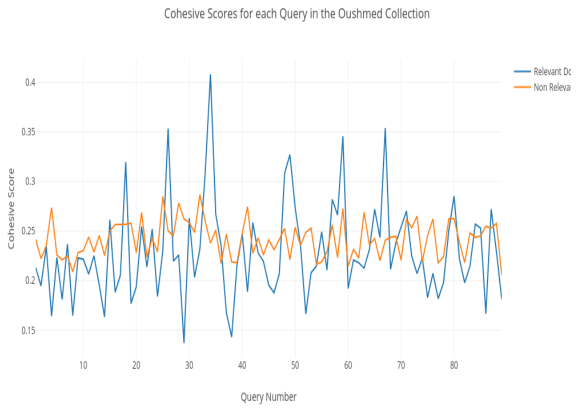


Fig. 5.5 Query Level Cohesion Score for Ohsumed Collection

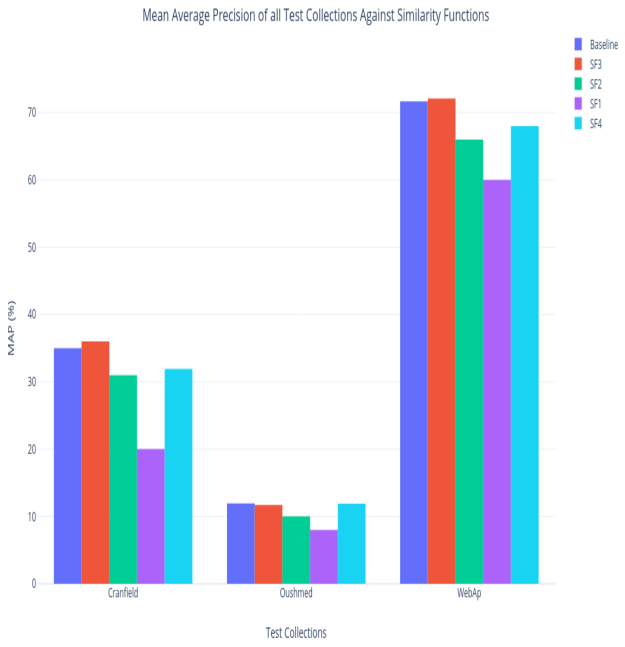


Fig. 5.6 MAP@100 of Similarity Functions for all Test Collections

in different test collections. Taking our hypothesis further, in this section, we will discuss the impact of the cohesion score in the ranking function can have on the performance of the system. Figure 5.6 illustrates the comparison of Mean Average Precision of different similarity functions ($CSF1$, $CSF2$, $CSF3$, and $SF1$) against the baseline ($LVSM$ specified in section 2.10.1). For the WebAP and the Cranfield collection, the MAP for the $CSF3$ is slightly better than the $LVSM$. This supports our hypothesis that if relevant documents are more cohesive, a certain boost based on the cohesion score can improve the performance. Moreover, as the NR documents have slightly higher cohesion for the Ohsumed collection; therefore, the $CSF3$ and all other similarity functions reduced the performance of the system, which is expected. We also used precision at the top 5 and 10 documents (p@5 and p@10) as well as the Mean Reciprocal Rank of the first relevant document (MRR) to access the different re-ranking methods for top results [195]. Table 5.2 shows that for the WebAP, and Cranfield, $CSF3$ outperforms the $LVSM$ as well as Max passage approach ($SF1$). For Ohsumed, the $LVSM$ gives better results. However, the cohesion score improved the p@5 that reflects that for some queries as the relevant set had higher cohesion (spikes shows in figure 5.5), it helped to improve the top rank documents. The best performing results were highlighted in bold in Table 5.2.

We also did a comparison of the average precision query by query for $LVSM$ and $CSF3$ at top 100 results to check whether the increase or decrease in performance is distributed across all queries or the boosting approach penalized the top queries (easy queries) significantly. The motivation behind this analysis was that if a relevant document is more cohesive, then due to cluster hypothesis [213] the relevant (and cohesive) documents will be closer in graph space, and therefore, by boosting the document score with cohesion, a difficult query would give better performance as compared to the $LVSM$.

We took the 20 worst performing queries (difficult queries), and the top 20 best performing (easy queries) queries and compared the performance of $LVSM$ with $CSF3$ query by query. For WebAP, in 16 of the 20 difficult queries, the boost given by cohesion score gave slightly better average precision to $LVSM$. Overall on average, precision was boosted by 2% for the difficult queries. In contrast to that, for the top 20 queries, the cohesion score was either equal or slightly reduced the overall precision for each query, and that is due to the fact that for WebAp, the average precision for easy queries was already very high, i.e. $\approx 95\%$ and since the documents are already clustered together in the result set, the passage graph does not affect much and could introduce some noise which causes the little dip in performance for $CSF3$ overall. Similarly for Ohsumed, the boosting did not help much in improving the overall precision for both easy and difficult queries. Rather the precision on average for NR was higher for the 14 difficult queries out of 20. Furthermore, the correlation was seen even with a decline in performance. As the non-relevant documents had higher cohesion values, therefore, the boosting reduced the performance, which still reflects that cohesion is a useful measure to the document relevance. Lastly, for the Cranfield, we see a stronger correlation between the cohesion score and the average precision. For all the difficult and easy queries, where the cohesion of R was higher than the NR set, the performance was improved, which supports our intuition of boosting the difficult queries with the cohesion score. Though the average number of passages per document is similar in Cranfield and Ohsumed (6-7 passages), the overall difference between the size of the test collection is huge. Therefore, due to the small graph size $k = 10$ it is hard to get the correct contextual notion of the document from a test collection that is larger, which can be one the reason for the low MAP for $CSF3$ for the Ohsumed collection. Increasing the graph size for Ohsumed could cover more contextual notion in the graph, but it would be computationally expensive. One way to reduce the complexity is to generate the graph only from the answer set and not from all documents in the test collection.

5.3 Measure of Document Cohesion Dependent on Query

In the previous section, we measured cohesion by looking at the similarity between passages from the same document. We considered the inter-connectivity of passages with in the document. Later we introduced a similarity measure that incorporates the cohesion score along with the document similarity score. Then we used that similarity score to re-rank the answer-set and measured the performance. One limitation of only using the inter-connectivity score is that each document will have the same cohesion score, regardless of the query information. One way to consider the importance of neighbour nodes is to check their similarity with the given query q. We want to improve our definition of cohesion by including a query passage graph. The motivation behind it is that, by including the relation of each passage with the query in graph, every document will have a different cohesion against a given query.

Previously, we noticed that by performing the ranking with only consideration of the inter-connectivity, the $CSF1$(multiply) and $CSF2$(Addition) did not perform well. This is due to the fact that a highly cohesive document will only be relevant for a certain query. Therefore, when we apply a boost to that document for all queries, most cohesive documents will end up higher in ranking compare to the expected relevant documents. Hence, performance of the system decreases. In this section, we investigate whether including the query information within the passage graph would help overcome the problem of the same cohesion score for all queries.

5.3.1 Methodology

Let's consider a bipartite graph G(U,V, E) where in the set U, each vertex p_i represents a passage and in set V each vertex represent a query q_i. The edge represents a similarity weight w between two nodes. By considering the same approach used in equation 5.1, we are now extending our graph to include the query-passage similarity. The cohesion of a document D_i is derived from the following equation

$$C(q,d_i) = \left(\frac{1}{N(N-1)} \sum_{j:p_{j_i} \in d_i} \sum_{n_{j_i}} sim(p_{j_i}, n_{j_i}) \right) \times \left(\frac{\sum_{p_{ji}} sim(p_{ji}, q)}{N} \right) \mid n_{ji} \in d_i \qquad (5.5)$$

In Equation 5.5, we consider the inter-connectivity of passages by adding the similarity score of all the neighbour passages that are from the same document. In addition to that, strength of passage p_{ji} is based on how strongly they are connected to query and its neighbours from the same document n_{ji}. The motivation behind this is that by calculating the cohesion of each document that way, we are still considering the relationship of passages within each document. However, we are eliminating the limitation of using the same cohesion score of a document for each query. In equation 5.5, the first part of the numerator illustrate the contextual notion by measuring the relationship between passages, and the second part demonstrate the importance of a document to a query by calculating how strongly all passages p_{ji} of a document d_i are connected to the given query q. In Figure 5.7, we can see that a document $d2$ has different cohesion score for $q1$ and $q2$. We re-defined cohesion of a document by not only considering the relationship between passages from the same document but also considering the relation of each passage with the given query.

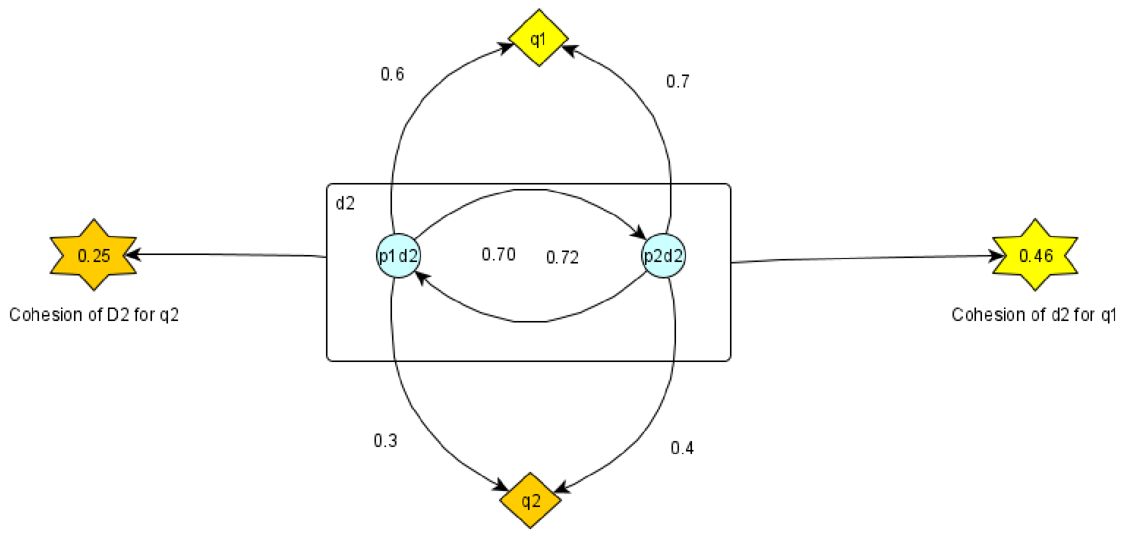


Fig. 5.7 Query Dependant Cohesion Graph of a document

5.3.2 Experiments and Results

Taking into account the experiments in the previous section, following are the extending hypothesis:

1. We hypothesized that by including the query information, a document that is strongly connected within itself (i.e.shown less topic drift in document content) and has higher similarity with the query, is considered to be more cohesive (and relevant) to the query.Therefore, a significant difference in the cohesiveness scores of R and NR sets for any given query should be found.

2. By including the query-passage relatedness in graph, cohesion score would be more effective as a performance measure even for smaller size test collections (e.g Oushmed collection).

Similar to previous section, we divide our discussion of experiments by explaining the results of both hypotheses in the following subsections.

Query Cohesion Score for relevant and non-relevant Documents

In this section, we present the experimental results to illustrate whether by including the query graph, *R* and *NR* are still comparable based on their Query Cohesion score. We use Equation 5.5 to calculate the cohesion score for each document. We have calculated the cohesion score for *R* and *NR* set against each query separately measured their statistical significance. Figure 5.8, 5.9, and 5.10 illustrate the average cohesion score of both relevant and non-relevant sets of each query in the form of a line plot. As seen in all figures (Figure 5.8, Figure 5.9,Figure 5.10) the relevant documents has shown higher cohesion on average for most of the queries. In previous section, we saw that for the Ohsumed collection (figure 5.5), *NR* has on average slightly better cohesion. However this has changed after we include the query graph. We have seen that now by including the query level graph, the cohesion score difference between relevant and non-relevant set is significant even for a small size test collection, which supports our hypothesis. Similarly, we used two-tailed Student t-test at a confidence level of 95% to determine whether the difference between relevant and non-relevant set is significant or not. As shown in Table 5.3, at the query level for all test collections, there was a significant difference between *R* and *NR* documents, which support our first hypothesis that there is a significant difference between both sets.

Effects of Query Cohesion Score on the Document Ranking

In section 5.3.2, we discussed the experiments based on the query cohesion score for the relevant and non-relevant documents. We also explored how average cohesion for each query differs in different test collections. Taking our hypothesis further, in this section, we will discuss the impact the cohesion score in the ranking function can have on the performance of the system. Figure 5.6 illustrates the comparison of Mean Average Precision of different similarity functions ($CSF1$, $CSF2$, $CSF3$, and $SF1$) against the $LVSM$ (Vector Space Model). For the WebAP, the MAP for the $CSF3$ is slightly better than the $LVSM$. Previously, by using cohesion (equation5.1) the performance for Ohsumed collection by using $CSF1$, $CSF2$, and $CSF3$ was not compareable to $LVSM$. However, after including the query graph, $CSF3$ and $LVSM$ produced similar MAP. Furthermore, the MRR and P@5 for $CSF3$ outperformed the baseline (*LVSM*). This supports our hypothesis that if relevant documents are more cohesive, a certain boost based on the cohesion score can improve the performance. And since by including the query graph, *R* documents were significantly more cohesive than the *NR* documents in Ohsumed collection, we saw the improvement in performance. By using only cohesion without the query graph, we saw in Table 5.1 that *CSF*3 outperformed the *LVSM*. However, after including the query graph, other cohesion based similarity functions like *CSF*2 and *CSF*1 has also produced better performance results. Hence, our initial motivation to generate a unique cohesion score for each document (against a given query) has exhibited promising results. Table 5.4 shows that for the WebAP, and Cranfield, *CSF*3 and *CSF*2 outperforms the *LVSM* as well as Max passage approach (*SF*1). Previously, we have seen in Figure 5.6 that for the Ohsumed collection, the *LVSM* gave better results. However, as shown in Figure 5.11, the new query based cohesion score *CSF*3 performed better than the *LVSM* for MRR, MAP, and also produced better precision scores (shown in Table 5.4. It reflects that for most queries as the relevant documents had higher cohesion (figure 5.10), our query based cohesion score improved the ranking and the the overall system's performance. The best performing results are highlighted in bold in Table 5.4.

Table 5.3 Cohesion Score Statistics for *R* and *NR*

	Avg Cohesion for *R*	Avg Cohesion for *NR*	T-Val	P-Val
WebAp	**0.10**	0.073	4.71	< 0.05
Cranfield	**0.19**	0.16	5.65	< 0.05
Ohsumed	**0.073**	0.0433	5.20	< 0.05

Table 5.4 Comparison of Similarity Functions and Baseline (*LVSM*)

	Cranfield			WebAP			Ohsumed		
	MRR	P@5	p@10	MRR	P@5	p@10	MRR	P@5	p@10
LVSM	0.75	40.0	**27.8**	0.97	95.0	93.8	0.49	30.7	**28.3**
*CSF*1	0.77	38.2	26.9	0.71	72.1	76.0	0.37	22.0	19.8
*CSF*2	**0.78**	39.5	27.1	0.97	**95.6**	**94.3**	0.48	29.5	26.1
*CSF*3	0.75	**40.0**	27.7	**0.97**	95.2	93.9	**0.50**	**30.7**	28.2
*SF*1	0.70	36.2	25.5	0.96	94.9	92.0	0.48	30.5	27.5

5.4 Summary

In this chapter, the main emphasis of our work was to explore the potential difference between relevant and non-relevant documents based on their cohesion scores. We presented a graph-based approach to calculate the cohesiveness of documents by using their passage based relation. We used two variations of the graph-based cohesion score. First, without including the query-passage graph, and secondly the one with the inclusion of query-passage graph. The results show that the cohesion score we introduced in this paper can be a useful measure for the document relevance. Moreover, we calculated the average cohesion scores of R and NR sets at a query level. The experiment showed a statistically significant difference between both sets. The relevant documents are more cohesive for all test collections except the Ohsumed, in which case it was the opposite when we used the cohesion graph without the query-passage graph. However, after recalculating the cohesion score with the query-passage graph (i.e. with Query Cohesion Score), the difference in R and NR set (in terms of their cohesion score) was statistically significant for all test collections. This shows that our extended approach with query cohesion score gave better performance results. Moreover, we also explored the use of cohesion score to re-rank documents, and we used different evaluation measures like MAP, MRR, p@5, and p@10 to capture the performance. We noticed that using both cohesion graphs (with and without query-passage graph) $CSF3$ outperformed the $LVSM$. We also saw a considerable improvement in performance with $CSF1$ and $CSF2$ after including the query-passage graph. This performance improvement substantiates our hypothesis that if the relevant documents have higher cohesion, the graph measure can be used as a indication to determine the to document relevance against a given query. Lastly, we also investigated the behaviour of easy and difficult queries against all test collections. We noticed that the cohesion score helped improve the performance for the worst functioning queries more than the easy queries.

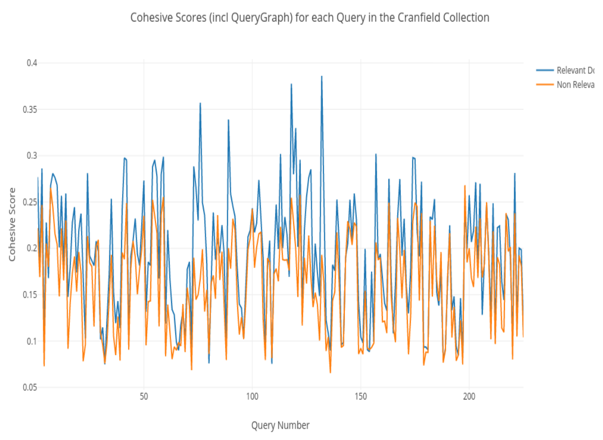


Fig. 5.8 Query Level Cohesion Score for Cranfield Collection

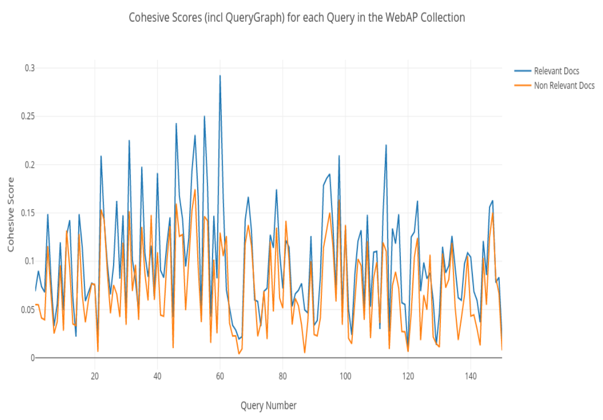


Fig. 5.9 Query Level Cohesion Score for WebAp Collection

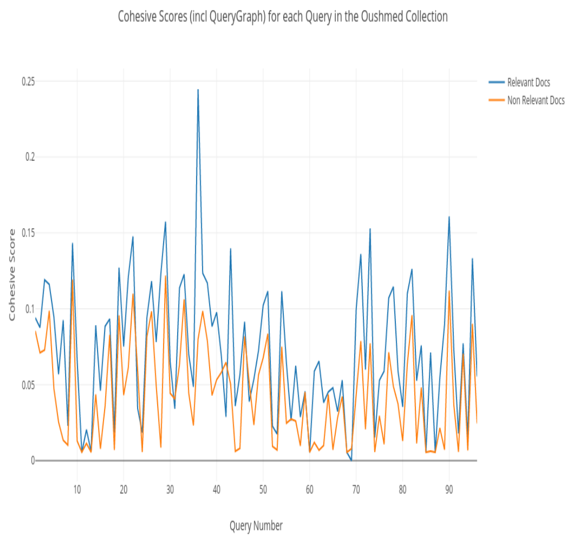


Fig. 5.10 Query Level Cohesion Score for Ohsumed Collection

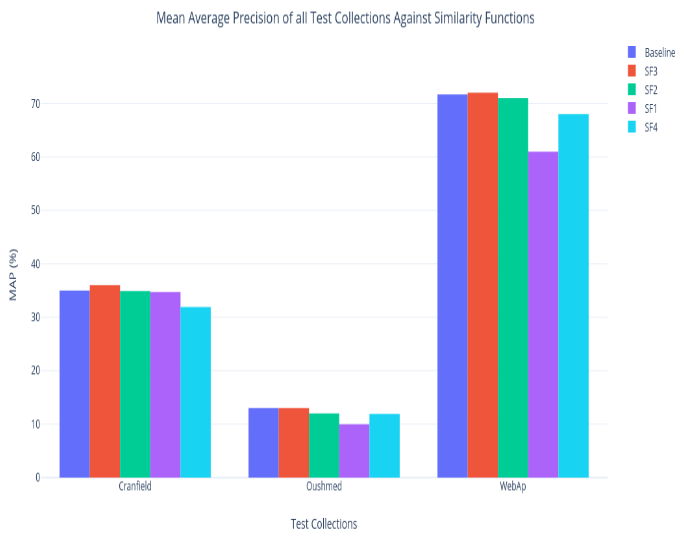


Fig. 5.11 MAP@100 of Similarity Functions for all Test Collections

Chapter 6

Query Expansion with Passage Level Evidence

6.1 Introduction

As with many areas in computer science, the overwhelming amount of data available is a major challenge for developing effective information retrieval procedures. Employing additional processes can produce better results, but at a computational cost. The fundamental issue is that text documents are represented by large vectors of terms and queries are predominantly short key terms aimed at retrieving a few select items amongst many. Standard approaches involve assigning a value to each term and returning a cache of documents that best match the inputted terms. In natural language, the same word will often be used to confer different meanings, and when presented in different contexts can embody different concepts. Furthermore, querying domain specific documents may require the user to refrain or adopt certain query terms in order to achieve best results. A successful system is one that is able to accurately interpret queries and return pertinent documents ranked by relevance. In many information retrieval systems, queries tend to be short and comprise a few indicative terms. Employing additional processes can improve results but at a computational cost. Standard approaches involve assigning a value to each term and returning documents that score highly in relation to the query terms submitted [186, 124]. The most informative terms are ones that feature highly in a document but not generally over the corpus [183]. Subsequent approaches attempt to incorporate additional inputs, such as part of speech tagging (POS) [28] probabilistic frequency [22] and semantic dependencies [136] to name a few. Additional issues are found with the polysemy nature of terms. A user who is not looking for the most popular instance of a term will find poorer results. Similarly, if a query contains a word that has many different occurrences in the corpus, identifying the instance that relates to their information need can be difficult.

In this chapter, we will discuss the different graph based approaches to augment the query with new words that could help in improving the overall performance for an Information Retrieval system. Firstly, we will discuss a graph approach to map out the semantic dependencies of terms and use those findings to reformulate the query. The terms to generate the graph are extracted from a passage level retrieval system. We also compared the results by taking the terms from an ad-hoc document retrieval system as well. As a baseline, we used Rocchio's Algorithm based on a relevance feedback approach. Later we employed the commonly used word embeddings approach i.e. *word2vec* to find the most appropriate terms to expand the query terms with more words. We also applied the query augmentation only on the difficult queries to see how result changes. Furthermore, we performed a comparison of expansion terms extracted from Rocchio's algorithm and *word2vec* to explore the similarities and differences in the generated output from both approaches. Lastly, we conclude the chapter by discussing the limitation of each approach and highlighting the significance of passage level retrieval for query augmentation task.

6.2 Use of Semantic Relatedness at Document and Passage Level for Query Expansion

In this section, we investigate the use of semantic dependencies to see if appropriate additional terms can be identified and used in query augmentation. To ensure that our approach is robust, we will investigate varying a number of parameters in our graph model; as well as by exploring different document pre-processing steps. To generate the graph model we also exploited passages instead of the whole documents. The motivation behind the adoption of passages for query expansion was to hopefully reduce noise that could occur and lead to the topic drift in the resulting query. Moreover, retrieving the indexed passages over documents from the IR engine shortened the amount of text to be processed in our graph approach. In this section, the described work discussed was taken from our published paper [188].

In the following sections we will first start with the explanation of our methodology to generate the graph model to encapsulate the semantic relatedness between the terms. We used the same passage retrieval model that we used in Chapter 4 for the Oushmed collection. We took the top retrieved document from our inverse rank approach ($SF5$ described in chapter 4) and generated the graph from it. We then discuss the experiments and the generated results by comparison our approach with Rocchio and Song's normalised HAL approach (explained in Chapter 3).

6.2.1 Graph Model

Consider a graph G(V,E) where *V* is a set of vertices where each vertex *v* represents a term and *E* represents the set of edges between terms. An edge exists between the term based on the sliding window. A sliding window of varying length was run over the text and the co-occurrences were observed by incriminating the strength of the edge weight between the target term node and the every proceeding and preceding term within the range of the window size. The window size *w* varied from one to ten to generate difference graphs. Our

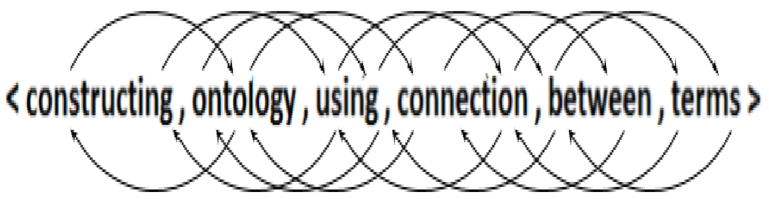

< constructing , ontology , using , connection , between , terms >

Fig. 6.1 Graph of terms with window size 2

data-set consists of a query and a number of labelled documents. Taking the original query, we labelled all of the returned documents as relevant. We limited the number of returned documents and passages to 2000. The number of relevant documents (and passages) for each query varies in number. While converting these documents to a vector space model, stop-words were removed. Taking all of the documents labelled as relevant, the terms were graphed as nodes and the co-occurrences of terms marked with a weighted edge. The edge weight signify the frequency of the neighbouring terms appeared across the set of relevant documents. Using this graph we augmented the original query from the words found there. In the first instance the highest unweighted co-term was selected to perform the augmentation. For the implementation of Song's normalised version [29], we divided the weight of the edge values with the degree of that term. We augmented the queries starting from level 1 to level 10. The level reflects the size of the co-occurring term window used on the returned passages. Depending on the approach used, a single term is selected and used to augment each word of the query. We explored different levels to determine what the optimal size in the sliding window should be. Figure 6.2 illustrates the basic flow of the complete system. Furthermore, Algorithm 2 specify pseudocode of how query expansion is performed for a given query.

To consider documents at passage level, different passage representation functions can be used to re-rank the results as well as to filter the returned text for query augmentation. We used conventional passage level retrieval methods i.e. Max passages and sum of passages discussed in chapter 4 for passage retrieval task. For sum of passages, k=2 was found to be the best and it was used to report the results.

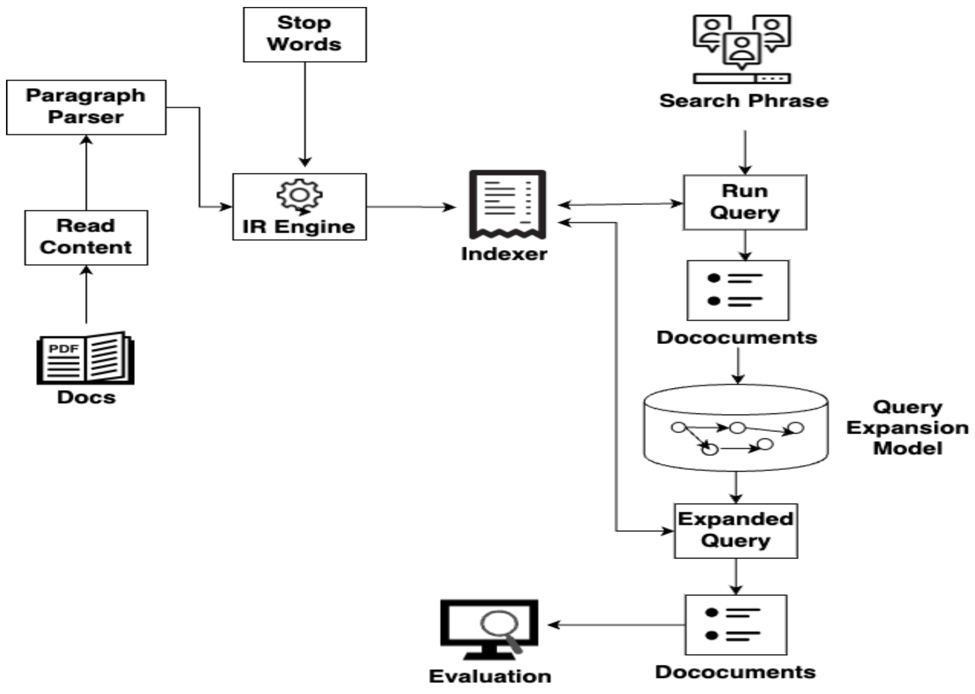


Fig. 6.2 Query Expansion Architectural Diagram.

6.2.2 Results and Discussions

In this section, we present the results of two query expansion algorithms and compare them against the baseline approach. Scenarios using documents as the basic unit and scenarios using passage level evidence are considered. We will explain how the performance is changed when we perform query augmentation.

Four different representations of the data were used in the query expansion process:

1. Document Level (*DL*): For each query, the original text of retrieved documents was used.

2. Passage Level (*PL*): Instead of taking the documents, passages are used to expand queries without any further processing on them.

Algorithm 2 Query Expansion With Documents

```
1: Array OriginalDocs ← GETDOCSFROMSEARCHENGINE(Query)
2: Array RelevantDocs ← GETRELEVANTDOCS(Query)
3: Array TrainingDocs ← []
4: i=0
5: for Doc in OriginalDocs do
6:     if RelevantDocs Contain Doc then
7:         TrainingDocs[i] ← Doc
8:         i++
9: MODEL ← TRAINQEMODEL(TrainingDocs)
10: ExpandedQuery ← MODEL.EXPANDQUERY(Query)
11: Array NewDocs ← GETDOCSFROMSEARCHENGINE(ExpandedQuery)
12: EVALUATE(NewDocs,RelevantDocs)
```

3. Max Passage Level (*MPL*): Once the passages are retrieved (just like in *PL*), the similarity function $SF1$ (discussed in chapter 4) is applied to extract the text of highest scoring passage of each document for the *QE*.

4. Sum of Passages Level (*SOPL*): Instead of taking the highest scoring passage, $SF2$ is applied with $k = 2$ as a parameter so that each document has a combined text of two top passages.

For the graph model, we used two approaches. In the first approach, we changed the window size to include more preceding and following terms for each given term to broaden the context. Then, for each query word, we chose the top term that was most pertinent. In the 2nd approach, we increased the number of terms added against each term in the query to check how the results change. Following are the results and discussion around the selected graph approaches. For the baseline, without using any expansion approach the recorded performance for a normal document level (DL) MAP was **13.50** and for the Max passage (*MP*) approach the MAP achieved was **13.09**. To calculate MAP we used the top 1000 documents i.e MAP@1000. We used the Max passage similarity function to report our passage level results as it was giving overall the best results after we applied the query expansion in comparison to document level results. The *MP* approach performed well compared to the sum of passages (*SP*) because the *MP* approach focused more on retrieving the most similar passage from an IR system against a given query. By most suitable we mean that the similarity between the top passage and query is high due to the increase in the term frequency compared to the passages ranked below.

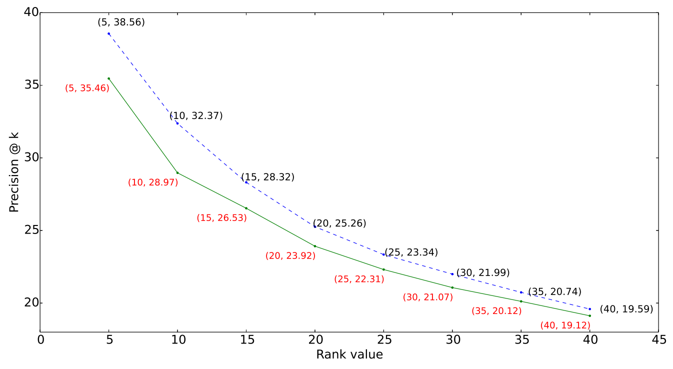
We used the aforementioned MAP results (for *DL* and *MP*) as a baseline against the varying sized window and the increased number of terms approach. The results are also compared to classic Rocchio expansion (explained in Chapter 2). For Rocchio, the parameters β, α, and λ can be tuned before the process begins; we used 16, 8 respectively as it is commonly used in the literature [30, 140] for α, β. Lambda(λ) was set to 0 as query syntax for Solr in Lucene does not support negative weights, and the assigned alpha and beta values were shown to produce the best results. The query will expand to a length equal to the number of all unique words present. The MAP value recorded for Rocchio at passage level was **31.45**%. This MAP was relatively high compared to other approaches discussed here due to the large size of the query after the expansion process. In these experiments, we refer to our edge count graph approach as (*EC*).

Varying sized windows

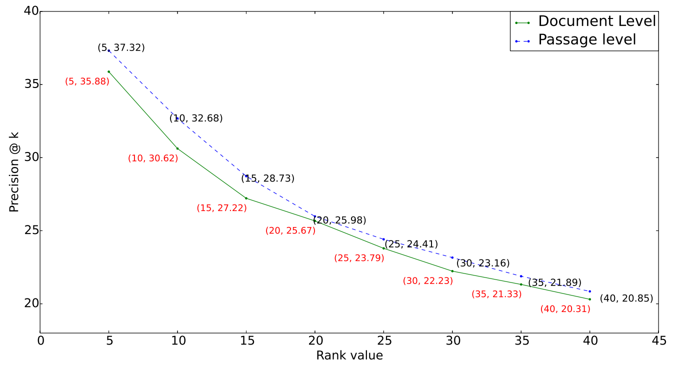
Table 6.1 illustrate the results with the variation of window size applied by using documents or passages. In Table 6.1, the *Level* represents window size. The *DL* results give slightly better MAP as compared to the *PL* approach. However, for the top results, passage level evidence was giving a better performance during our analysis (shown in figure 6.3a and 6.3b) when compared with the document level results, which reflects the significance of passages over the documents. In addition to that, by considering the variation of results at different levels, we took the best value in each query expansion approach. For example, for the graph approach (*EC*) it was 18.16 and for Song's *NEC* it is 18.74. We compared the results at this position with baseline at different MAP levels to check the significance of the improved results. To do that, we performed the paired student's t-test for the MAP at 5 to 50 with the difference of 5 in each iteration and calculated the p-value. Considering the paired sample t-test, the calculated p-value was less than 0.05 for both *EC* and *NEC*. It reflects that both graph-based algorithms significantly outperformed the baseline.

Increased Number of Terms

In Table 6.2, we show the results for the graph approach for all four representations of the document sets. There is a marked difference between the results for including terms to augment the query over the increase in the sliding window. The results improve and continue to improve in a linear fashion as the number of augmented terms is increased. This indicates that every newly added term had a positive influence on the overall results. Table 6.3 contains the results for Song's normalized HAL approach. Interestingly the graph approach makes noticeable improvements for the first three iterations, before increasing at a much slower rate. Song's approach only does so on the first two iterations.



(a) Graph Approach - Edge Count (EC)



(b) Song's Normalised Edge Count (NEC)

Fig. 6.3 Precision at K for Document and Max passage level.

The normal document length shows the strongest results for both approaches. Presumably, because there is more evidence from which to capture the semantic relatedness in the terms. MPL, SOPL, and PL all show results that are very near one another for both approaches. This indicates that both algorithms perform similarly when applied to smaller bodies of text. However, SOPL outperformed MPL and PL nearly at all levels, which supports our intuition behind using the passage representation function to isolate significant tracts of text.

Song's approach, however, does not show the same level of improvement as graph approach on the larger datasets, suggesting that it does not capitalize on the extra information. We believe that the reason for this is that the normalization smooths out some of the distinctions between terms. While this is a positive characteristic when grouping documents, it shows to return poor results when determining the highest distinguishing associative term. Normalising the term value has a similar affect as applying idf in the TF-IDF approach. It is designed with the goal of smoothing out the weights applied to terms that have a high instance in all documents, like stopwords. However, this effect might not be the most useful when dealing with words that have no particular advantage for query augmentation. In a scenario where we are looking for a word to add context to a query term, the common most co-occurring term will be sufficient for this purpose. In the case of normalising the values, if this same term has two common meanings, the overall impact of the term will be lessened in the normalisation process. Therefore our graph approach performs better across all levels. In future work, we aim to confirm this hypothesis by applying the same procedures to group documents and seeing if this maxim holds true.

Table 6.1 MAP(%) of SF1 for the Ohsumed Collection at Different Query Expansion Approaches Using the **Varying Sized Windows** Approach

Window Size	PL Edge Count	PL Normalised Edge Count	PL Rocchio	DL Rocchio	DL Edge Count	DL Normalised Edge count
Level 0	N/A	N/A	**31.45**	**31.54**	N/A	N/A
Level 1	17.01	17.15	N/A	N/A	17.12	17.62
Level 2	18.13	17.10	N/A	N/A	17.00	18.17
Level 3	17.85	16.98	N/A	N/A	17.50	18.44
Level 4	17.50	16.64	N/A	N/A	18.27	17.44
Level 5	**18.16**	17.12	N/A	N/A	**18.71**	**18.74**
Level 6	18.07	17.33	N/A	N/A	18.48	17.47
Level 7	17.42	**17.55**	N/A	N/A	18.47	17.33
Level 8	17.32	17.36	N/A	N/A	17.27	17.24
Level 9	18.00	17.07	N/A	N/A	17.49	17.28
Level 10	17.22	17.10	N/A	N/A	17.46	17.53

Table 6.2 MAP(%) of SF1 for Graph Approach at Different Query Expansion Approaches Using the **Increased Number of Terms** Approach Per Query Word

Additional terms	DL	MPL	SOPL	PL
Level 1	17.17	18.41	17.36	18.85
Level 2	21.22	20.62	21.06	21.03
Level 3	24.23	21.92	22.49	21.13
Level 4	25.64	22.56	23.58	21.80
Level 5	26.56	22.90	24.36	21.77
Level 6	26.51	23.20	24.86	22.85
Level 7	27.25	23.59	25.18	23.12
Level 8	27.27	23.90	25.33	22.95
Level 9	27.59	23.92	25.80	22.96
Level 10	**27.97**	**24.25**	**25.86**	**23.22**

Table 6.3 MAP(%) of SF1 for Song's Normalised HAL approach at Different Query Expansion Approaches Using The **Increased Number of Terms** Approach Per Query Word

Additional Terms	DL	MPL	SOPL	PL
Level 1	18.80	18.03	18.00	18.02
Level 2	21.14	20.76	20.59	20.59
Level 3	22.61	21.48	22.29	22.05
Level 4	23.54	22.30	23.65	22.29
Level 5	24.72	22.55	24.08	22.34
Level 6	25.00	22.97	24.22	22.83
Level 7	25.26	23.43	24.93	23.09
Level 8	25.17	23.50	25.16	23.26
Level 9	25.46	23.97	25.42	23.22
Level 10	**25.80**	**24.27**	**25.58**	**23.45**

6.2.3 Summary

In this section, we have undertaken an analysis of approaches to capturing semantic relatedness between terms in the text. While the approach fell short of the baseline used (Rocchio), we did make significant improvements over the basic retrieval performance. With regards to setting the window size, our results are closest to Song's setting of six; we found that five provided better results. The difference might be explained by using different datasets. These figures differ from Burgess's [136] and Kotov's [113] assertions that 8 and 20 respectively were optimal window sizes. Secondly, we found that increasing the terms added to the query produces better results. It is important to note that the number of additional terms used was only 10 per query term. This is dramatically less than used in the Rocchio approach, which uses every term in documents for which feedback was given. Moreover, we are taking evidence from relevant documents only; the Rocchio method also takes evidence from unrelated documents, which can help generate a very suitable query.

A third feature of note was the use of passages as pseudo-documents over entire documents. Our intuition was that the use of passages would aid the graphing of concepts because it would remove elements of noise found in a text document, which contains a number of topics. To a degree, this intuition proved feasible, as the results at passage level were quite competitive. The advantage here is that by applying this pre-processing step it reduces the amount of text needing to be processed. Although the document level results outperformed the passage level approaches (*MPL*, *SOPL*), we observed that the passage level outperformed consistently for precision@k metrics. Therefore, for a system where the priority is to ensure high relevance at top *k* results, passages evidence is useful for the query expansion task.

6.3 Utilizing Passages for Query Expansion using Pseudo Relevance Feedback and Word Embedding

In recent times deep learning techniques have become more and more prevalent in NLP tasks. Word embedding (WE) is a representation of a word, using real-valued numbers. They are a configuration of numbers that conveys the semantic and syntactic details of words and their context in a language that computers can comprehend. In this section, we consider the use of the word embedding approach *Word2vec* to generate a model and use that model to perform the query expansion task. *Word2vec* produces vector representations of words from a corpus of text. It has been claimed that the vector similarity between the matching embeddings created by this method often accurately captures the semantic relatedness between words[142]. As a result, this technique offers a practical means of locating words that are semantically connected to any certain word. Because the goal of query expansion (QE) is to discover words that are semantically related to a given query, word embedding techniques like *Word2vec* can be employed to find relevant expansion terms and improve the effectiveness of QE. Another popular WE approach is called Global Vectors for Words Representation (GloVe) [141]. It is an unsupervised learning technique that creates word embeddings by combining the global word-to-word co-occurrence matrix from a given corpus. These two approaches are successful in a variety of natural language processing applications [141, 226, 56], but as can be seen from the literature, *word2vec* is the approach that is most frequently used for word embedding for QE due to its strong performance over Glove. Therefore, for the QE task, we compared a state-of-the-art Pseudo Relevance Feedback (PRF) approach with the *word2vec* approach for the Ohsumed collection.

The *Word2vec* algorithm creates a vocabulary from the corpus and then learns how to represent words in vector form. We took the top *k* (where k=100,200,1000,2000) documents and passages against each query and formed a corpus *C* that contains all these return document as one. *Word2vec* model was trained using that corpus. We also build smaller models by creating a new model for each query by using only the top *k* documents or passages against each query. The motivation of creating these small models for each query was to compare it side by side against the standard pseudo-relevance-feedback approach (like Rocchio) as it takes the top documents against each query to produce the expansion results. We compared our results against the Rocchio and *word2vec* approach by using documents as well as passages. We also analysed the frequency of similar terms derived from each compared approach and performed an analysis of how well each model is different from the other by measuring the Jaccard coefficient [84] amongst the discussed approaches. This gives us an extra insight into how different semantics and ad-hoc approaches process the same amount of text (which in our case are documents and passages) and produce comparable results.

6.3.1 Experimental Methodology

We used a similar experimental setup that was employed in the previous section 6.2. We used Rocchio Algorithm as a baseline and compared it with the *word2vec* approach. Our benchmark was the normal document level MAP@1000 result i.e. 13.50 to evaluate the performance improvement. Details of the semantic expansion by using word embeddings i.e. word2vec are as follows:

Word Embeddings via Word2vec

We computed the embedding of each word by using the *word2Vec* [141, 142] approach. *Word2Vec* is a popular approach that transforms the given text into vector representation. The algorithm learns the vector representation from the constructed vocabulary by using a neural network. More detail on *Word2vec* is described in the Chapter 2. We employed a *QueryWord* approach that was presented by Arora et al. [6] to construct a matrix of all the top words for each query. Instead of using the CBOW model [142] (used by Arora et al.), we opted for the skip-gram model [142] as it is more useful on a small corpus [97]. For each term q_i in given query Q, We used the Apache spark ML Library (MLLib)[1] implementation to capture the top *k* similar words. Words are extracted by capturing the cosine similarity

[1]https://spark.apache.org/docs/1.2.0/mllib-guide.html

between the query term's vector representation and the vector of all the terms in the corpus. Once we generated the pool of top n words, we sorted the list and picked the highest similarity score terms. This way, we are extracting the expansion terms that are strongly related to the terms in the query.

Pseudo Relevance Feedback (PRF) -Rocchio Algorithm

Similar to the *word2vec* approach, we used the top k documents and passages to extract the expansion terms. The top k documents were assumed to be relevant, and no negative weights were applied to the expanded query terms as we did not provide non-relevant document or passage information. Similar to *word2vec* we used the top n terms to expand the given query Q.

Term Similarity Between Word2vec and Rocchio

One motivation of this work was to check the frequency of similar terms against each query for both the word2vec and PSF approaches. *Word2vec* takes into account the contextual notation of terms, however, the Rocchio Algorithm is purely derived from the bag of words model with the tf-idf approach. As per our knowledge, we haven't seen any thorough comparison of both of these common approaches in terms of their generated output i.e. expansion terms for the query. We used Jaccard similar coefficient to measure the similarity and the diversity of both sets against each query.

6.3.2 Experimental Setup and Notations

We used only the Ohsumed test collection for this experiment as it already had a lower performance (MAP) for top k results (amongst the test collections employed in this thesis). The details of the test collection and the passage size are specified in section 2.2. We used different measures to evaluate the performance of both approaches like MRR, MAP, P@5 and P@10. To generate the word2vec model we use the standard settings similar to Arora et. al [6] local embedding settings. The only difference in our model was that we set the embedding size = 100 (instead of 200) and window size =8 (instead of 10) due to the smaller document length. Iteration set for learning =20 was applied along with the rest of the default parameters of the Apache Spark MLLib Implementation. For Rocchio, we kept the same parameters (as specified in section6.2) i.e. α, β and λ were set to 8, 16 and 0 respectively. Lambda was set to 0 as the query syntax for Solr in Lucene does not support negative weights, and the assigned alpha and beta values were shown to produce the best results.

The used the following representation to classify our experiments.

- Document Level Rocchio (DLR): Here, we passed the top *k* documents to Rocchio to generate the expansion terms.
- Passage Level Rocchio (PLR): Instead of documents, Top *k* passages were used to generate the expansion terms from Rocchio.
- Document Level Word2vec - Different build model for each query (DWVQ): Top *k* documents were used to generate a word2vec model and extract the top expansion terms. Here against each query, we build a new word2vec model by using only the top *k* documents.
- Passage Level word2vec - Different build model for each query (PWVQ): Top *k* passages were passed to the word2vec model and generated the top expansion terms. Similar to DWV, we used the top *k* passages to build a new model against each query to extract the expansion terms.
- Document Level Word2vec - One model for all queries (DWVC): Here, we combined all top *k* documents against each query. Build a corpus and use that to generate the word2vec model. Once the model was generated, we used that model for all queries to retrieve the expansion terms.
- Passage Level Word2vec - One model for all queries (PWVC): Similar to DWVC, instead of using the documents, top *k* passages against each query were used for a corpus and build the model.

6.3.3 Experimental Results

We have categorized our experiments into two different categories.

- First we wanted to check whether the passage information give better expansion terms for the Rocchio and Word2vec compared to the document based counterpart.
- Investigate whether there are any commonalities between the expansion terms that are derived from the pseudo relevance feedback approach i.e. Rocchio and the word2vec approaches.

We split our discussion on experiment results by presenting the results pertaining to these categories below.

Performance Comparision between PRF and word2vec

In this section, we use different evaluation measures like MAP and Precision @10 to compare the PRF and word2vec based approaches. Table 6.4 illustrates Mean Average Precision (MAP@1000) for different methodologies at document as well as at the passage level. In the table, the *Level* corresponds to the number of documents or passages used for the PRF or word2vec model generation. We can see that the *PLR* approach performed the best till the top 1000 passage level and also produced better performance compared to the document level performance benchmark (i.e. 13.50). It signifies the importance of using the passages over the documents and support our intuition that passages can be better than documents for query expansion task. We also noticed that at the 2000 passage level, *PWVC* performed the best and that is because the word2vec models rely more on the provided data. The algorithm learns and improves the embeddings based on the data provided. Where for PRF approaches whereby providing more data can cause query drift due to additional noise coming from the extra training set, word2vec does the opposite of that. We have also noticed that our both word2vec based approaches at passage level i.e. *PWVQ* and *PWVC* overall performed better than the document level measures i.e *DWVQ* and *DWVC*. To evaluate the performance we added 10 additional expansion terms against each query because our experimental results in section 6.2 performed best when 10 extra terms were added (Level 10) for query expansion.

Furthermore, we also measured the Precision@10 to check how different approaches are generating results. Precision@10 is important in a situation where the main focus of the application is to produce better results at the top level in the ranked list. Figure 6.4 illustrates the bar charts that represent the discussed approaches at different levels of documents and passages. Similar to the results for MAP, we noticed that *PLR* gave the best results till level 1000. At Level 100, both passage-based approaches for word2vec gave better results than their document level counterparts. However, for other levels (200,1000,2000), document-based word2vec gave better results. We suspect this is because the most dominating terms in the query are derived more from the top documents due to the higher size of training data. Adding more passages in the training set are useful but not at the top-level results.

Common Terms Analysis between PRF and Word2vec

Previously, we outlined that passage-based approaches like *PWVC* gave the overall best performance for the word2vec, and *PLR* outperformed other approaches in the Pseudo Relevance Feedback category. In this section, we took both these approaches and performed a common terms frequency analysis on them against all the queries. We used the Jaccard coefficient (*JC*) to gauge the occurrences of common expansion terms through both query

Table 6.4 MAP(%) for the Ohsumed Collection at Different Query Expansion Approaches Using the Fixed Expansion Terms = 10

Training Set Level	DLR	PLR	DWVQ	PWVQ	DWVC	PWVC
Level 100	15.0	**15.1**	7.95	8.6	11.10	12.0
Level 200	13.0	**14.1**	12.05	8.0	12.1	10.1
Level 1000	9.97	**10.24**	10.0	9.1	9.68	9.93
Level 2000	9.17	9.44	9.92	10.0	9.92	**10.8**

expansion approaches (*PLR* and *PWVC*). We used level 100 and level 2000 to perform the comparison as we wanted to observe the effect of training data size via both approaches. Out of the total of 106 queries in the Ohsumed collection, 42 queries had a $JC > 0$ between the expansion terms at Level 100. However, for level 2000, 50 queries had $JC > 0$ between the expansion terms. Only one query had 4 words in common with a $JC = 0.2$ at Level 100 and the highest *JC* noted at Level 2000 was 0.15 having 3 words in common. The overall results have shown that adding more training data to build the word2vec model improves the chances of finding more words between the PRF and word embedding approach.

6.3.4 Summary

In this sub-chapter, we explored a neural network-based word embedding approach (word2vec) and compared it with the traditional pseudo relevance feedback approach (PRF) i.e. Rocchio Algorithm to extract the terms for the Query Expansion (QE) task. We have shown that the word2vec model built by using a variation of a passage-based training set outperformed the document-based counterparts. We divide our experiments into two categories. At first, we outlined the different passage and document-based PRF and word2vec approaches. We trained our word2vec model at different levels to observe the change in result. We have shown that the passage-based PRF and word2vec approach i.e. *PRL*, *PWVQ*, and *PWVC* outperforms their document-based counterparts i.e. *DLR*, *DWVQ*, and *DWVC*. This shows the significance of passages over documents and supports our intuition that by reducing noise from the data using passages, better performing models can be produced. We further extend our work to capture the common expansion terms derived from the best-performing PRF and word2vec approaches. We have observed that almost half of the query terms had a $JC > 0$ through both approaches (*PRL* and *PWVC*). In addition to that, the word2vec approach gives better results when larger training data is provided especially when the focus is to produce highly relevant top-ranked results (i.e. through P@5 and p@10).

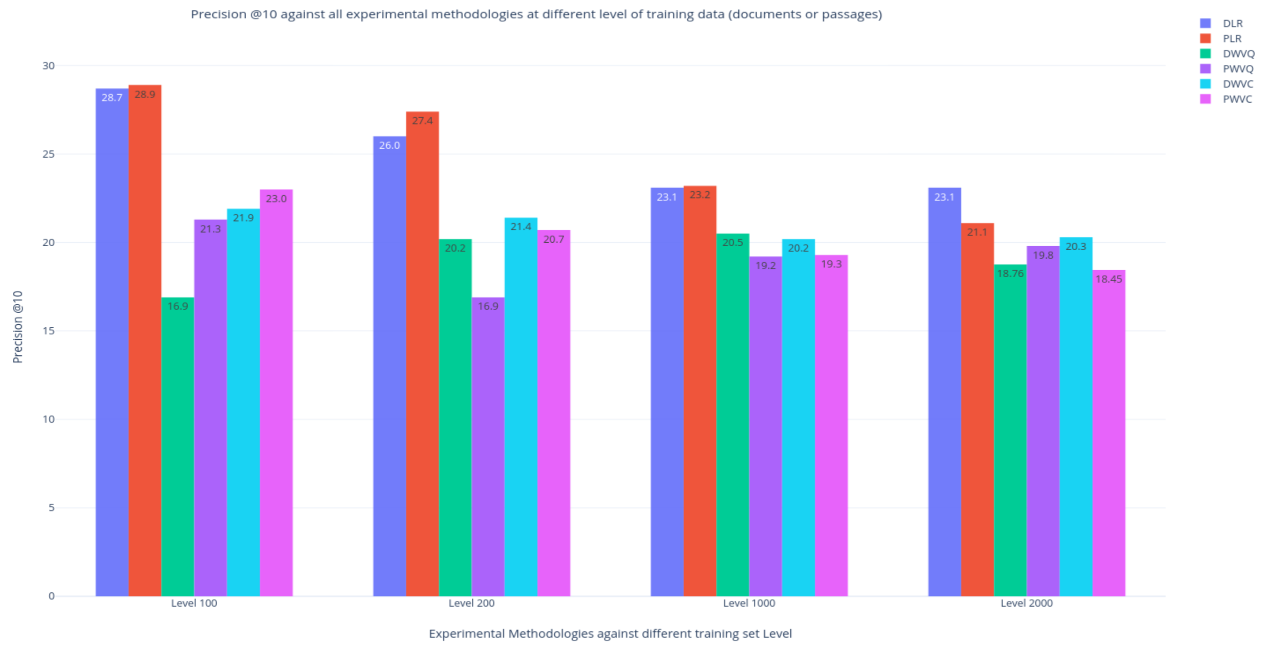


Fig. 6.4 Precision@10 for Different Query Expansion Approaches Using the Fixed Expansion Terms = 10.

Chapter 7

Graph of Answer Set Passages For Query Performance Prediction

Finding a way to distinguish between an easy and a difficult query is a challenging task for Information Retrieval (IR) systems. Several indicators have been developed in the past which are categorised into two main approaches: pre-retrieval and post-retrieval. Pre-retrieval indicators focus on the features of the query terms [45], whereas the post-retrieval indicators take the answer set for a given query into account [64]. The main purpose of all QPP indicators is to differentiate between an easy and a difficult query and formulate strategies to improve the performance of the IR system with the gained knowledge of a given query. To check the effectiveness of a given predictor, a correlation of average precision (which depicts the performance of the system) is used against the resulting score of the predictor.

In general, the QPP task is studied by extracting the information from a document retrieval system. However, there is very little work done so far where researchers have utilized passage information as a measure for predicting query performance. The passage based QPP is either used in QA tasks [114, 42] or a passage evidence approach like MaxPassage [32, 190, 177] is combined with document similarity score to formulate a query performance predictor. Utilizing passages rather than documents is beneficial especially where the documents in the collection are diverse in nature [210]. Similarly, as shown by Kaszkiel et al. [106] that a single passage can be far more effective for ad-hoc retrieval task than whole-document ranking with the cosine measure. They emphasized that because cosine measure has shown to be biased towards short documents, using passages can overcome this issue due to their pre-defined length. Also, passages are effective in reducing the additional noise caused by the topic drift within the document as well as computationally less expensive to compare.

The stated work in this chapter is taken mostly from our published work [189]. We used a completely novel approach by composing an answer-set graph where each node represents a passage and their similarity score with each other forms an edge between them. The structure of this graph captures valuable information about the nature of the answer-set. Our approach is based on a cluster hypothesis [118], which states that similar documents within a cluster represent the same relevance to information need [217]. Let us consider two extreme scenarios where an easy query returns a cohesive answer-set or a difficult query exhibiting ambiguity results in a diverse answer-set covering a large number of potentially unrelated topics. The motivation behind considering the graph of the answer-set is that by considering the cluster hypothesis, we would see a significant difference in the graphs of the aforementioned scenarios; in the first case, we would expect to see one large cluster of related topics with edges with a relatively high similarity score. For the second scenario, we would expect to see several small clusters (highly connected within) that are loosely connected to each other with edges of lower similarity weight. In order to check how well connected are these answer sets, different graph features can be used that determines the quality of the graph and the relatedness of its nodes. Furthermore, if a query is 'difficult' due to certain terms dominating the query, or certain terms having several meanings, then we would expect different documents to be returned by virtue of different aspects in the query, and hence a graph of these documents would exhibit substantially different features. As we use the relation between passages within the answer-set (top k passages), our approach is computationally economical compared to applying the similar approach and generating graph by using the documents (due to the small text size i.e. passages compare to full-length documents), and provide flexibility in terms of formulating the strength of edges within the graph by utilizing numerous similarity measures (term frequency, semantic approaches etc.) as per the need.

7.1 Answer-set Graph

Consider an undirected weighted graph $G(V,E)$ where each node $n_i \in V$ represents a passage p_x. An edge $e_{x,y} \in E$ where the weight of an edge $e_{x,y}$ illustrate the similarity between vertices x and y. The weight of an edge represents the strength of similarity between passage nodes p_x and p_y. We used cosine similarity as a measure of similarity between passages. Initially every node in the answer-set graph is connected to all other nodes, forming a complete graph. In this chapter, we used an edge count threshold ec as the top number of edges (with highest weight) to form a sub graph where we discard all the remaining edges below the ec. We represent each document as pseudo-documents (passages) i.e. $d^{'} = \{p_1, p_2, \dots p_n\}$. We assume that the set of all documents (corpus) C, a query q and retrieval approach (RA) remain constant. The main purpose of our work is to utilize passage evidence from the answer-set and formulate a predictor to evaluate the quality of the ranked results given by the retrieval approach. Here we use the term 'answer-set' that represents the top ranked k passages as a retrieved list (by using RA) for a given query q where k is a parameter that can be changed as per the need. As mentioned previously that at first, the undirected graph is constructed by taking all the passages within the answer-set as nodes and calculating the similarity between them as edges; formulating a complete graph and then an edge count (ec) threshold is applied to generate a subgraph. An undirected complete graph of 100 nodes will have a maximum of $n(n-1)/2$ i.e. 4950 edges. So if $ec = 250$, the remaining edges after the top 250 edges from the aforementioned graph will be discarded. We used $ec = \{100, 250, 500, 1000\}$ and the motivation of setting different ec value is to remove the noise from the answer-set graph and only consider the relationship between passages that are forming close clusters based on cluster hypothesis. Figures 7.1a and 7.1b illustrate the initial answer-set graph and the state of the graph after the edge count ec parameter is applied.

Below we explain two different graph measures that we adopted with the passage answer-set graph for the QPP task.

Cohesion

In graph structure, cohesion is used to measure the interconnection between the nodes of the graph. One way to capture cohesion in a given text is to look at the term distribution [168, 215] or by analysing different clusters generated from a returned documents graphs [104, 160]. Recently, Sarwar et al. [192] defined cohesion by considering the similarity of passage within the same document. Rather than taking only the passages in a returned list from the same document [192], we applied a similar approach on the passage answer-set that measured the inter-connectivity of the answer-set. We take cohesion as a divergence of topics within the answer-set. If the returned answer-set of passages discuss a similar topic, the cohesion of the answer-set will be high and vice-versa. As per the cluster hypothesis, the nodes that belong to the same cluster convey similar relevance to the information need [213]. For an easy query, as the returned documents will largely be relevant, therefore, according to the cluster hypothesis these documents will be similar to each other. Hence, the cohesion score can be an effective measure to distinguish between the easy and hard query and can be used as a predictor of query performance.

For an undirected graph, the size of a graph is its number of edges $|E|$ where it contains n(n-1)/2 maximum number of edges. We denote the cumulative weight of the graph as $W = \sum_e wt$ where wt is an edge weight and total number of edges are defined as $|E|$. The cohesion of an answer-set (AC) graph is denoted as follows:

$$AC = \frac{W}{|E|} \tag{7.1}$$

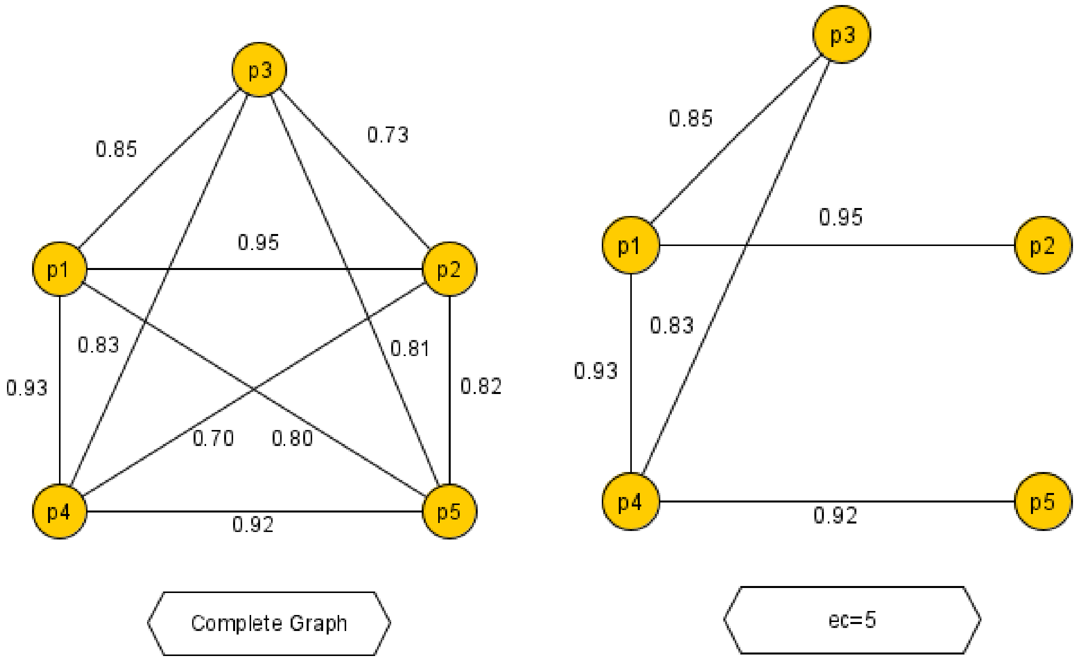
For P passages retrieved as an answer-set, the time complexity of generating the cohesion can be denoted as $O(P \times |E|)$.

Minimum Spanning Tree

The graph of the answer-set contains a node for every passage returned and an edge between each pair represents the similarity (above the *ec* threshold). The structure of this graph should capture some valuable information regarding the nature of the answer set.

One measure often used in graph theory is to find the minimal network of a given network. The resulting graph should be connected i.e. involve all nodes of the original network. This is been explored in traffic networks [57], and communication networks [25] to find the fundamental 'cheapest' graph with the costly redundant edges removed. Prim's [164] and Kruskal's [116] are two well-known algorithms that achieve this. The algorithms produce what is known as a Minimal Spanning Tree (MST). We can apply this idea to the graph representing the answer set. We are interested in finding the minimum spanning trees where the lower the edge weight between two nodes, the lower the similarity.

For a set of loosely related topics, we will have low similarity between many of the nodes in the answer set; this will lead to a low-weighted minimum spanning tree. One could argue that for easy queries we would expect larger clusters of related topics with high similarity scores. If that is the case, the nodes with a relatively lower similarity score in the graph are expected to have a higher edge weight score. Hence, the MST score will be relatively high. Whereas, a graph formed with a hard query is expected to form loosely connected clusters in the answer-set graph. Therefore, the MST score of a hard query graph will be distinguishable compared to an easy query graph. MST illustrates the overall connectedness of a graph, which is why it can exhibit different characteristics of the answer set. By taking the lower score edges to generate MST, we are measuring a summary statistic of the answer-set graph which can capture the nature of the retrieved responses for a given q. Thus, we used MST as a predictor for the query performance task. We used Kruskal's implementation of MST and calculated the MST score by adding all the weights assigned to each edge of the generated minimum spanning tree from the answer-set graph. For P passages retrieved as an answer-set, the time complexity of generating the MST can be denoted as $O(E \times logP)$ where E is the number of edges in the answer-set graph.



(a) Passage Answer-Set Complete Graph (b) Answer-Set Graph; Edge Count (ec) =5

Fig. 7.1 Answer-Set Graph Before and After Applying Edge Count Threshold

7.2 Evaluation

7.2.1 Experimental Setup

We performed our experiments using three different test collections i.e. WebAp, Cranfield, and the Ohsumed collection. The characteristics of these test collections is specified in table 2.2. To identify the passage boundaries, we employed a half overlapping fixed-length window size approach [32]. For each test collection, we predicted the performance of each query based on its top 100 highly ranked documents and passages.

To extract the graph features specified in section 7.1, we used a sub-graph based on the top edge count *ec* value. We used $ec = \{100, 250, 500, 1000\}$ to create four different graphs. One drawback of keeping more edges in the graph is to increase more noise and similarly by lowering the amount of edges, we may lose the context and important relations within the answer-set. Moreover, for an easy query, increasing the edge count may decrease the MST score of the graph due to the selection of low weighted edges. Hence, we will lose the information about the highest correlated nodes with in the graph. Therefore, setting the higher threshold for edge count will overall reduce the correlation performance. We present our correlation results with $ec = 250$ as we found it to be the optimal one. We will further discuss the results on all four graphs with different edge count size in Experimental Results section. We employed Pearson's correlation as an evaluation measure since it is commonly used to evaluate the query performance prediction. It is computed by considering the correlation between the actual document level retrieval performance i.e. average precision (AP) for a given query q and the output generated for q by the specified query performance predictor.

7.2.2 Baselines

Oftentimes in IR, it is a common scenario where the retrieved documents are not relevant to the information required for a given query q but the IR system still shows a high similarity to q due to the reasons like term distribution, and query word dominance. One way to quantify the variation of this similarity for q in the ranked list is to use the standard deviation of the query-document similarity scores [46, 178]. We choose our first baseline as a standard deviation on a fixed cut-off point i.e. at top 100 document σ_{100} as it has been shown as an effective predictor [46, 161]. We also considered other popular post-retrieval QPP approaches like Normalized Query Commitment(NQC) and Weighed Information Gain(WIG). Similar to the normal standard deviation approach like σ_{100}, NQC also utilise standard deviation by further normalizing it with the corpus (all documents in the collection) score. The motivation in NQC is to estimate the query-drift [201] and use that as a predictor for query performance. Furthermore, WIG measures the difference between the average score of top-ranked documents and of the corpus. For WIG and NQC, we generated the corpus by concatenating the content of all the documents (except any tags and boundary information). So far, all these QPP approaches used document retrieval information. Approaches like $NQC(psg)$ and $WIG(psg)$ have been explored lately that utilize the same fundamental principles while using passage information [177]. Here we also include the σ_{100}(psg) to show a clear comparison of existing document vs passage-based approaches that have been explored in the past. To calculate the AP for the passage based approaches, we used the standard Max passage [32] estimation approach with top 100 ranked documents.

7.2.3 Experimental Results

In this section we present the experimental results to illustrate the evaluation of different QPP approaches. Table 7.1 summarize the results of existing approaches against our passage answer-set graph methods (Cohesion and MST). The results are reported against three different test collections. First, we observe that the document level QPP approaches along with their passage based equivalent methods exhibit differing performance results. For example, we can see that for the WebAp, the passage-based counterparts of WIG i.e $WIG(psg)$ give a much better correlation result compared to its document level equivalents. Similarly, for all the test collections, the passage-based σ_{100}(psg) performed better than the document level NQC and WIG methods, which shows the significance of the use of passage based evidence in QPP tasks.

Furthermore, we can see that our newly proposed graph-based approaches give a positive correlation for all the test collections, which supports our intuition of exploiting the answer-set of passage graph for predicting the query performance. Our MST approach outperformed all the other document and passage-based baselines for the Cranfield and the Ohsumed collection. The standard deviation σ_{100} approach at document level performed the best for the WebAp, yet our MST and cohesion approach outperformed the NQC and WIG baselines. This indicates the significance of our answer-set passage graph approaches. The standard deviation is calculated based on the spread of query document similarity scores on a given ranked list. For the webAp collection, we noticed that there was a significant difference (calculated via student's t-test) between the Sd scores of the top 5 hard queries and easy queries. For MST we saw the difference as well, however it wasn't as strongly significant as the SD scores. One possible reason for it is that in our MST approach we are focused more on the relevance of the returned passages with themselves and not the query itself. Our MST work is based on the cluster hypothesis where we hope to find different MST score for easy and hard queries. One limitation here for MST can be that in some situations even if the returned passages are irrelevant, their answer-set graph can still exhibit a high MST score (similar to easy query) because those irrelevant passages are strongly connected to each other. And we saw that this was the case for WebAp collection; Therefore, SD approach σ_{100} gave better results compared to the MST approach.

In Table 7.1 we reported our results for edge count *ec* set to 250 top-scored edges in the answer-set graph. Adding more edges i.e. passage information can cause extra noise, which could result in the reduction of prediction performance. In Table 7.2 we show the results for different edge count to illustrate the implication of reducing or increasing more passage information on the overall performance. We can see that for MST when the *ec* was increased from 100 to 250, the performance is always improved against all test collections. Similarly, when we go above 250 edges in our graph, the performance tends to decrease gradually for 500 and 1000 edges. This shows that adding more passage information causes noise and as a result, reduce the predictor's performance.

Table 7.1 Prediction via Pearson Correlation for document and passage QPP baselines against Cohesion and MST. Boldface: best result per test collection.

	Cranfield	WebAp	Ohsumed
σ_{100}	-0.07	**0.63**	0.48
NQC	-0.113	0.17	-0.09
WIG	-0.32	-0.13	-0.07
$\sigma_{100}(psg)$	0.082	0.30	0.46
$NQC(psg)$	-0.074	0.062	-0.09
$WIG(psg)$	-0.30	0.19	-0.08
Cohesion	0.03	0.19	0.24
MST	**0.25**	0.24	**0.48**

Table 7.2 Pearson Correlation Comparison of Passage Graph Approaches at Different Edge Count (ec). Boldface: Best Result Per test collection against multiple ec.

	Cranfield				WebAP				Ohsumed			
	100	250	500	1000	100	250	500	1000	100	250	500	1000
Cohesion	**0.09**	0.03	0.007	-0.005	0.14	**0.19**	0.17	0.15	**0.34**	0.24	0.18	0.13
MST	0.21	**0.25**	0.18	0.10	0.15	**0.24**	0.22	0.18	0.45	**0.48**	0.41	0.20

7.3 Summary

In this chapter, we introduced a novel approach to predict the query performance by using the features of the answer-set passage graph. We explained two graph features i.e. Cohesion and Minimum Spanning Tree (MST) and calculated their respective scores and used them as a measure to find its correlation against the retrieval performance (AP) for QPP task. We also compared different baselines that utilize not just the document relevance but also the passage information for QPP. The results show that our MST approach outperforms the existing QPP approaches against the Cranfield and the Ohsumed test collection. The MST approach also outperforms other popular methods like WIG and NQC against WebAp test collection as well. Our Cohesion and MST approach shows a positive correlation for all test collections whereas other document and passage-based counterpart exhibits mixed performance results. It signifies that graph features of the passage answer-set can be a useful measure for Query performance prediction. Moreover, features like cohesion and MST exhibit the connectedness of the answer-set. In a scenario where the user wants to visualize the differentiation between the answer-set in terms of topic drift or uniqueness, our approach can be useful there as well.

Chapter 8

Conclusion

In this chapter, we present a broad summary of all the contributions made in this dissertation. This is followed by a discussion on the significance of our research outcomes. Lastly, we outline several limitations of our work and present future avenues of research.

8.1 Summary of Contributions

The main goal of the research in this thesis is to examine the use of passage level evidence in multiple Information Retrieval tasks. We explore some of the major IR tasks such as Retrieval Models and Ranking, Query Expansion and Query Performance Prediction tasks by utilizing the passage information. To perform these tasks, we developed a system that implements the state of the art passage retrieval approaches particularly specified by Callan et al. [32] and compares them to document level retrieval. We integrated our system with applications like SOLR (to index passages) [196], mallet (for topic modelling) [78], deepLearning4J + Apache Spark (for *word2vec*) [105]. We extend this system to incorporate the graph-based structure of passages and integrate our passage-based retrieval system with a few baseline algorithms for query expansion (e.g. Rocchio, *word2vec*, HAL) and QPP (e.g. WIG, NQC, Standard Deviation) task. In our work, we explored numerous experimental settings and applied a combination of parameters to explore the deviation of the results and to find the optimal solutions to improve the performance of the system. Table 8.1 enlist the summary of research outcomes along with their corresponding hypotheses, research questions and the future work. The following contributions are made in this thesis which reflects the effectiveness of passage-based evidence in Information Retrieval tasks.

8.1.1 Passage based Ranking Models to Improve Document Level Retrieval

In this work, we focus on improving the document ranking by using different passage-based evidence. Several evaluation measures were employed and more in-depth analysis was undertaken to identify the most suitable passage-based evidence approach. We also explored the notion of query difficulty to understand whether the best performing passage-based approach helps to improve, or not, the performance of certain queries. This work addresses our first research question, Q1:

Q1:Are there any other useful measures that can be used as passage evidence for ad-hoc retrieval?

Chapter 4, provides a detail of the commonly used existing similarity measures that utilize passage evidences, such as Max passage, and the sum of passages. Rather than utilize the passage similarity score (traditional approach), we introduced two new similarity measures that utilize the rank of the passages. We have shown that the rank of a passage is a useful measure, and when used separately, or in conjunction with, the document score can give better results as compared to other passage or document level similarity approaches. Our first hypothesis $[H1]$ has been proven to be valid as we observed that the rank of a passage is an effective measure and produced efficient performance results compared to previously used passage based similarity measures. This work was published by Sarwar et al. [190, 191] which details the experiments conducted on two test collections (WebAp, Ohsumed) against different passage-based similarity measures.

8.1.2 A Graph-based Passage approach to distinguish between the Relevant and Non-relevant Documents

This work presents a graph-based approach at the passage level to calculate a cohesion score of each document. Here, we represent the top returned passages as a graph with each passage corresponding to a vertex. We connected the vertices (passages) that belongs to the same document to form a graph. The aim was to identify the difference between the cohesion score of relevant and non-relevant documents for a given query. This work addresses our 2nd research question, Q2:

Q2: How do relevant document compare to non-relevant document when using passage evidence?

Chapter 5 presents a new approach to calculate the cohesion of a document by utilizing the graph-based passage approach. We discussed two kinds of graph approaches. To generate the first graph (discussed in section 5.2), we considered only the top returned passages within the document to calculate the cohesion. Though there was a clear distinction between the cohesion score of the relevant and non-relevant documents; we noticed one limitation that by only using the inter-connectivity score (graph-passage approach), each document will have the same cohesion score regardless of the query information. We improve our definition of cohesion (discussed in section 5.3) by including a query passage graph (bipartite graph), which in a result assign every document with different cohesion against a given query. The overall results showed improvement compared to the standard cohesion approach based on the first graph. Against all the tested collections, we have shown that our graph-based passage approach is an effective measure to differentiate between the relevant (R) and non-relevant (NR) documents. Which shows that our hypothesis [$H2$] is true and our introduced cohesion measure that leverages the connection between passages within the same document from a relevant set is stronger than the document from the non-relevant. We have also shown that the retrieval performance (MAP, MRR, and P@10) was improved when the cohesion score is combined with the original document similarity score. This work was published by Sarwar et al. [192].

8.1.3 Use of Semantic approaches at Passage Level for Query Expansion Task

We explored the use of exploiting semantic relationships within the text. We used passages as well as documents to investigate whether by reducing some noise (with the use of passages) more useful query expansion terms are generated. We compared the traditional relevance feedback approach with the word embedding approaches and discussed the variation in the overall performance of the system. This work addresses our third research question Q3:

Q3: Can combining such passage evidence with different word embedding approaches (such as word2vec, term-occurrences matrix, HAL) produce better knowledge for query expansion and improve the performance of an IR system?

In chapter (6.1) we presented a graph model that is built upon the sliding window of varying length against the provided text (document or passages). The purpose of this graph approach was to capture the semantic relatedness between terms in the text. Our main focus of this work was to use passages, to record the degree of connection between terms, and use this to suggest possible additional words for improving the precision of a query. The described work was published by Sarwar et al. [188]. Furthermore, we also explored the neural network-based word embedding approach (*word2vec*) and compared it with the traditional pseudo relevance feedback approach (PRF) i.e. Rocchio Algorithm in section 6.2. We have built the *word2vec* model with different data-set and has shown that corpus build based on the passages outperformed the document level training set. The main contribution of our work in chapter 6 was to utilise passages for query expansion, which in result can improve the system, and also perform a comparison between the *PRF* and *word2vec* approach in terms of findings the common expansion terms that both approaches provide against a given query. Overall, we observed that the different AI based or Pseudo Relevance Feedback approaches we employed produce more relevant query expansion terms when using the documents instead of the passages. In this case, our hypothesis [$H3$] has not been proven to be entirely true as per the employed test collection. Though we did see minor dominance of passage based evaluation results (like p@5 or the *PLR* approach) over the document, the results did not remain consistent against other evaluation measures. As the Ohsumed collection is comprised of small length documents, utilizing passages may not be effective to develop models for term-selection.

8.1.4 Passage based Answer-Set Graph for Query Performance Prediction

We present a novel approach of representing the top returned passages (answer-set) as a graph where each node represent a passage and an edge weight indicate the similarity score between these passages. By examining the answer-set graph we developed new predictors that utilize graph features such as cohesion and minimum spanning tree. This work addresses our fourth research question Q4:

Q4: Given the passages returned (answer-set) for the respective query (q), could combining a relationship between the passage answer-set be used as an effective measure for the Query Performance Prediction task?

Chapter 7 outline the detailed explanation of our graph-based approach and the performance of the experiments against three test collections (WebAp, Ohsumed, Cranfield). Based on the empirical evaluation, we show that our answer-set graph predictors i.e. MST and $Answer - set\ Cohesion$ are very effective and perform even better (for Cranfield and Ohsumed Collection) than the current state-of-the-art QPP approaches. We compared different baselines (NQC, WIG, Standard deviation) that utilize not just the document relevance but also the passage information for QPP. The baseline of standard deviation approach at passage level $\sigma_{100}(psg)$ was presented for the first time through our work as it was not discussed previously in the literature. The hypothesis [$H4$] has proven to be true as we have shown that our proposed graph measures i.e. $Answer - set\ Cohesion$ and MST produce a positive correlation to the average precision, and the MST approach outperforms other passage-based

Table 8.1 Summary of Research

Research Question	**Hypothesis**	**Research Outcome**	**Future work**
Q1	H1	Hypothesis H1 was valid and we have seen that rank of a passage is an effective measure	Future work involve experimenting on Large test collections, applying learning approaches to optimize parameters and utilization of passage annotations in WebAp to improve ranking.
Q2	H2	Hypothesis H2 was valid and we have seen that *R* have a higher cohesion score than *NR* documents	In future, other graphs features and different semantic based similarity measures can be applied on the graph to include contextual information between passage similarity.
Q3	H3	Hypothesis H3 was not proven to be entirely true for the selected test collection	Future work involves extending the current approach with other variants like LDA, BERT, and FastText.
Q4	H4	Hypothesis H4 was proven to be true and we demonstrated the significance of utilizing passages to generate a positive correlation with *AP* with the answer-set Cohesion and MST approach.	Answer-set passage graph can be utilized in future for topic visualization and generating clusters to show diversity of answer-set.

baselines against all the test collections. Hence, the graph features can be used as an effective measure for the QPP task. This work was published by Sarwar et al. [189].

8.2 Future Work

The potential research avenues of future work originating from this thesis are suggested as follows:

8.2.1 Large Size Test Collections and Learning Parameters

As this thesis presents the use of passages across multiple test collections like Cranfield, Ohsumed and the WebAp. We have observed that for a document collection with relatively larger document length i.e. WebAp, approaches with passage-based evidence (Max Passage, Inverse Rank) exhibit better precision. Therefore, one possible avenue to further explore the passage evidences is to test them on big test collections like TIPSTER, GOV2, and ACQUAINT. WebAp is a subset of the GOV2 collection and it would be interesting to examine how the overall results would vary when our newly proposed ranked based passage evidence approach is applied to it. Moreover, as per our results, it is shown that the passage level evidence on its own is not sufficient to improve the document ranking significantly for the selected test collections. There are multiple passage-based evidence parameters (passage score, rank, α, β) that affect the overall results. During the literature review, we haven't come across any paper that tries to optimize these parameters. It can potentially be an interesting area to explore to utilize machine learning or genetic algorithms to learn the most suitable parameters to combine the passage evidence to improve the performance and the ranking of the result set.

8.2.2 Different Similarity Measures as an edge weight for the Graph

Our work discussed in chapter 5 and chapter 7 outlines a weighted graph G= (V,E) where each vertex *pi* represents a passage and *E* is a set of edges where the weight of an edge corresponds to an edge-weight function that is based on the similarity between passage nodes p_i, p_j. We employed a standard cosine similarity to capture the strength of the relationship between the two provided texts (passages in our case). Though the cosine similarity is a useful measure, it does not capture the semantic relatedness between the text. Also, in a given text, there might be other features (synonyms and entities) that could provide a better sense of similarity between the text. By taking the same graph approach that we proposed, an interesting future avenue could be to define the edge weight based other semantic based similarity measures such as, entity-based, word embeddings (BERT, Word2vec), or topic modelling (LDA).

8.2.3 Other graph features to distinguish between relevant and non-relevant documents

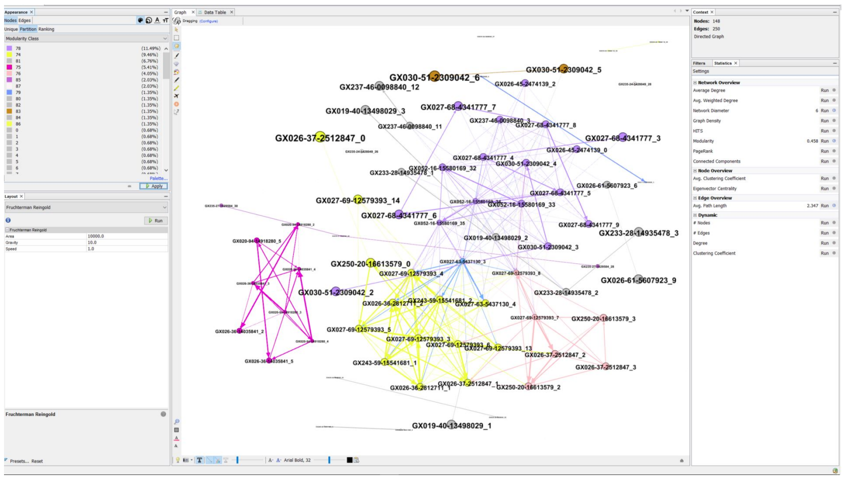
In graph theory, an edge between two nodes represents a connection between them. Often times finding the strength of that connection is not enough. In social networks, it is important to capture how important that node is based on multiple features of that connected graph. Approaches like Page Rank [157, 120] or HITS [111] are commonly adopted to identify the significance of connected nodes. Also, graph measures like centrality, clustering coefficient, and modularity signify the importance of neighbouring connections within the graph. Our work in chapter 5 discusses the cohesion feature of the generated passage graph. Other graph features like centrality, and community detection can also be used where instead of just looking at the relationship between passages from the same document, we can consider the edges from other documents as well. One possible future direction would be to extend our approach to define passage centrality and compare it with the centrality graph model proposed by Bendersky et al. [12, 115].

8.2.4 Utilization of more knowledge to improve the quality of expansion terms

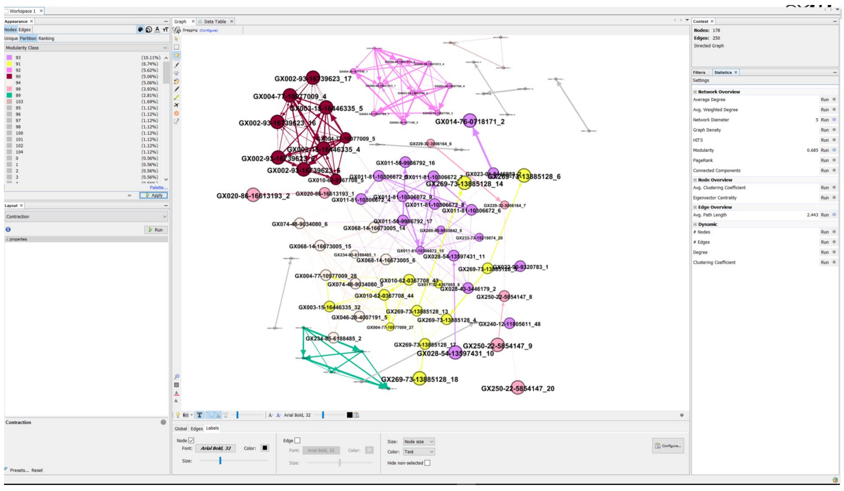
The aim of our work in chapter 6 was that the use of passages would aid the graphing of concepts because it would remove elements of noise found in a text document. To a degree, this intuition proved feasible, as the results at the passage level were quite competitive. The advantage here is that applying this pre-processing step reduces the amount of text needing to be processed. In future, this pre-processing step can be further improved. One interesting avenue would be to apply the Latent Dirichlet Allocation (LDA) [22] to the corpus and use the results to inform on where best to segment the documents into passages. Furthermore, one thing we have noticed through our experiments is that by increasing the number of expansion terms, the overall performance increased. Based on that, further future extension would be to find the optimal number of expansion terms. As the focus of our work was to take only the top returned documents (with the assumption that top documents tend to be more relevant) to train the word embedding models (graph approach and word2vec) and also due to the computational constraint, we noticed that providing more text can form better embeddings and relationship amongst the words. Therefore, one possible future direction of this work would be to train the model by using the complete corpus (local embedding) or with the extension via external source (global embedding) as specified by Arora et al. work [6]. On the other hand, rather than using the static word embedding approach like *word2vec* where given a word, its embeddings is always the same in whichever sentence it occurs. Another interesting area would be to use contextual embedding like BERT [54] or ELMo [162] where the embeddings are obtained from a model by passing the entire sentence, which will provide a better and contextually more relevant expansion terms for a given query. Recent work undertaken by Zheng et al. [238] and Padaki et al. [155] can to be considered as a baseline with future extension of applying passage evidence to the generated models and compare the results. Similarly, the inability of *Word2vec* to handle unfamiliar or out-of-vocabulary (OOV) words is one of its major flaw. The generated model won't be able to understand a word or create a vector for it if it hasn't seen it before. The only option left is to use a random vector, which is not optimal. A word embedding approach named FastText can handle OOV problems by paying attention to subword information using n-gram [26]. FastText renders each word as an n-gram of characters rather than immediately learning word vectors. For instance, the fastText representation of the word "artificial" with n=3 is $< ar, art, rti, tif, ifi, fic, ici, ial, al >$, where the angular brackets denote the word's beginning and end. Which enables the embeddings to comprehend suffixes and prefixes and helps to grasp the meaning of shorter words. Once character n-grams have been used to represent the word, a skip-gram model is trained to learn the embeddings. Rare words perform well with fastText. Therefore, even if a word wasn't observed during training, its embeddings may still be obtained by splitting it up into n-grams. FastText is also shown to be effective on small training datasets [60]. As a result, another future work would be to use the Ohsumed Collection to compare the provided results against the FastText method for the Query Expansion task.

8.2.5 A topical visualization of answer-set to improve the user experience

In chapter 7 we discussed a novel passage-based graph approach that focuses on the relationship within the answer-set. We applied that relationship to a Query Performance Prediction task. The connectedness of the answer-set can be exhibited by the graph features like *AC* and *MST* against a given query. In a scenario where the user wants to visualize the differentiation between the returned set in terms of topic drift or uniqueness, our approach can be useful there as well. One very interesting future avenue here is to find a visual representation of the answer-set based on the characteristic of the query (easy or hard) in such a way that a clear



(a) Answer-Set Graph for an Easy Query



(b) Answer-Set Graph for a Hard Query

Fig. 8.1 Answer-Set Graph example for an Easy and a Hard query

distinction between the topically variant documents is shown to the user. We have noticed that the graph properties varies against the easy and the difficult query (as stated through cluster hypothesis [217, 197]). Figure 8.1a and 8.1b illustrate a graph visualization of an easy and hard query for the webAp test collection. We could see that an easy query returns a coherent answer-set whereas a difficult query exhibit ambiguity resulting in a diverse answer-set (more spread out clusters) covering a large number of potentially unrelated topics. We used the modularity feature [24] of Gephi [10] to formulate the clusters. The thickness of the edge represents the similarity score between two nodes. Though both graphs indicate that passages within the same documents tend to form clusters with each other. However, we did not get to explore how to represent it better visually to the user so that their overall search experience can be improved.

References

[1] Aalbersberg, I. J. (1992). Incremental relevance feedback. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 11–22. ACM.

[2] Ai, Q., O'Connor, B., and Croft, W. B. (2018). A neural passage model for ad-hoc document retrieval. In *European Conference on Information Retrieval*, pages 537–543. Springer.

[3] Albahem, A., Spina, D., Cavedon, L., and Scholer, F. (2016). Rmit@ trec 2016 dynamic domain track: Exploiting passage representation for retrieval and relevance feedback. In *TREC*.

[4] Allan, J. (2005). Hard track overview in trec 2003 high accuracy retrieval from documents. Technical report, MASSACHUSETTS UNIV AMHERST CENTER FOR INTELLIGENT INFORMATION RETRIEVAL.

[5] Alqahtani, A. S., Saravanan, P., Maheswari, M., and Alshmrany, S. (2022). An automatic query expansion based on hybrid cmo-coot algorithm for optimized information retrieval. *The Journal of Supercomputing*, 78(6):8625–8643.

[6] Arora, P., Foster, J., and Jones, G. J. (2017). Query expansion for sentence retrieval using pseudo relevance feedback and word embedding. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 97–103. Springer.

[7] Aryal, S., Ting, K. M., Washio, T., and Haffari, G. (2019). A new simple and effective measure for bag-of-word inter-document similarity measurement. *CoRR*, abs/1902.03402.

[8] Aydın, F., Hüsünbeyi, Z. M., and Özgür, A. (2017). Automatic query generation using word embeddings for retrieving passages describing experimental methods. *Database*, 2017(1).

[9] Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999). *Modern information retrieval*, volume 463. ACM press New York.

[10] Bastian, M., Heymann, S., and Jacomy, M. (2009). Gephi: an open source software for exploring and manipulating networks. In *Third international AAAI conference on weblogs and social media*.

[11] Bendersky, M., Croft, W. B., and Diao, Y. (2011). Quality-biased ranking of web documents. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 95–104.

[12] Bendersky, M. and Kurland, O. (2008a). Re-ranking search results using document-passage graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 853–854. ACM.

[13] Bendersky, M. and Kurland, O. (2008b). Utilizing passage-based language models for document retrieval. In *European Conference on Information Retrieval*, pages 162–174. Springer.

[14] Bendersky, M., Metzler, D., and Croft, W. B. (2010). Learning concept importance using a weighted dependence model. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 31–40.

[15] Benedetti, F., Beneventano, D., Bergamaschi, S., and Simonini, G. (2019). Computing inter-document similarity with context semantic analysis. *Information Systems*, 80:136–147.

[16] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

[17] Berry, M. W., Dumais, S. T., and O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595.

[18] Bi, K., Ai, Q., and Croft, W. B. (2018). Revisiting iterative relevance feedback for document and passage retrieval. *CoRR*, abs/1812.05731.

[19] Białecki, A., Muir, R., Ingersoll, G., and Imagination, L. (2012). Apache lucene 4. In *SIGIR 2012 workshop on open source information retrieval*, page 17.

[20] Blanco, R. and Lioma, C. (2012). Graph-based term weighting for information retrieval. *Information retrieval*, 15(1):54–92.

[21] Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.

[22] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

[23] Blondel, V. D., Gajardo, A., Heymans, M., Senellart, P., and Van Dooren, P. (2004). A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM review*, 46(4):647–666.

[24] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

[25] Boesch, F. T., Harary, F., and Kabell, J. A. (1981). Graphs as models of communication network vulnerability: Connectivity and persistence. *Networks*, 11(1):57–63.

[26] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

[27] Boole, G. (1847). *The mathematical analysis of logic*. Philosophical Library.

[28] Brill, E. (2000). Part-of-speech tagging. *Handbook of natural language processing*, pages 403–414.

[29] Bruza, P. and Song, D. (2001). Discovering information flow using a high dimensional conceptual space. In *Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval*.

[30] Buckley, C., Salton, G., Allan, J., and Singhal, A. (1995). Automatic query expansion using smart: Trec 3. *NIST special publication sp*, pages 69–69.

[31] Büttcher, S., Clarke, C. L., and Cormack, G. V. (2004). Domain-specific synonym expansion and validation for biomedical information retrieval (multitext experiments for trec 2004). In *TREC*.

[32] Callan, J. P. (1994). Passage-level evidence in document retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 302–310. Springer-Verlag New York, Inc.

[33] Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

[34] Carmel, D. and Kurland, O. (2012). Query performance prediction for ir. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1196–1197.

[35] Carmel, D., Shtok, A., and Kurland, O. (2013). Position-based contextualization for passage retrieval. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1241–1244.

[36] Carmel, D. and Yom-Tov, E. (2010). Estimating the query difficulty for information retrieval. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 2(1):1–89.

[37] Chen, Q., Kim, S., Wilbur, W. J., and Lu, Z. (2018). Sentence similarity measures revisited: ranking sentences in pubmed documents. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 531–532.

[38] Chen, R.-C., Spina, D., Croft, W. B., Sanderson, M., and Scholer, F. (2015). Harnessing semantics for answer sentence retrieval. In *Proceedings of the Eighth Workshop on Exploiting Semantic Annotations in Information Retrieval*, pages 21–27. ACM.

[39] Chen, Y., Yin, X., Li, Z., Hu, X., and Huang, J. X. (2012). A lda-based approach to promoting ranking diversity for genomics information retrieval. *BMC genomics*, 13(3):S2.

[40] Chifu, A.-G., Laporte, L., Mothe, J., and Ullah, M. Z. (2018). Query performance prediction focused on summarized letor features. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1177–1180.

[41] Chowdhury, G. G. (2010). *Introduction to modern information retrieval*. Facet publishing.

[42] Clarke, C. L., Cormack, G. V., Lynam, T. R., and Terra, E. L. (2008). Question answering by passage selection. In *Advances in Open Domain Question Answering*, pages 259–283. Springer.

[43] Cohen, D. and Croft, W. B. (2018). A hybrid embedding approach to noisy answer passage retrieval. In *European Conference on Information Retrieval*, pages 127–140. Springer.

[44] Cohen, D., Mitra, B., Hofmann, K., and Croft, W. B. (2018). Cross domain regularization for neural ranking models using adversarial learning. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1025–1028.

[45] Cronen-Townsend, S., Zhou, Y., and Croft, W. B. (2002). Predicting query performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306.

[46] Cummins, R., Jose, J., and O'Riordan, C. (2011). Improved query performance prediction using standard deviation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1089–1090.

[47] Cummins, R. and O'Riordan, C. (2009). Learning in a pairwise term-term proximity framework for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 251–258.

[48] Dai, Z. and Callan, J. (2020a). Context-aware passage term weighting for first stage retrieval. In *Proceedings of the 43rd international ACM SIGIR conference on Research and development in information retrieval, Virtual Event, China*.

[49] Dai, Z. and Callan, J. (2020b). Context-aware term weighting for first stage passage retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1533–1536, New York, NY, USA. Association for Computing Machinery.

[50] Dalton, J., Allan, J., and Smith, D. A. (2011). Passage retrieval for incorporating global evidence in sequence labeling. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 355–364.

[51] Dang, E. K., Luk, R. W., Allan, J., Ho, K. S., Chan, S. C., Chung, K. F.-L., and Lee, D. L. (2010). A new context-dependent term weight computed by boost and discount using relevance information. *Journal of the American Society for Information Science and Technology*, 61(12):2514–2530.

[52] Dang, H. T., Kelly, D., and Lin, J. J. (2007). Overview of the trec 2007 question answering track. In *Trec*, volume 7, page 63.

[53] de Campos, L. M., Fernández-Luna, J. M., and Huete, J. F. (1998). Query expansion in information retrieval systems using a bayesian network-based thesaurus. In Cooper, G. F. and Moral, S., editors, *UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, University of Wisconsin Business School, Madison, Wisconsin, USA, July 24-26, 1998*, pages 53–60. Morgan Kaufmann.

[54] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[55] Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In Lee, D., Schkolnick, M., Provost, F. J., and Srikant, R., editors, *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, San Francisco, CA, USA, August 26-29, 2001*, pages 269–274. ACM.

[56] Diaz, F., Mitra, B., and Craswell, N. (2016). Query expansion with locally-trained word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 367–377, Berlin, Germany. Association for Computational Linguistics.

[57] Djidjev, H., Sandine, G., Storlie, C., and Vander Wiel, S. (2011). Graph based statistical analysis of network traffic. In *Proceedings of the Ninth Workshop on Mining and Learning with Graphs*.

[58] Dkaki, T., Mothe, J., and Truong, Q. D. (2007). Passage retrieval using graph vertices comparison. In *2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System*, pages 71–76. IEEE.

[59] Dong, Q., Liu, Y., Cheng, S., Wang, S., Cheng, Z., Niu, S., and Yin, D. (2022). Incorporating explicit knowledge in pre-trained language models for passage re-ranking. In Amigó, E., Castells, P., Gonzalo, J., Carterette, B., Culpepper, J. S., and Kazai, G., editors, *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 1490–1501. ACM.

[60] Edwards, A., Camacho-Collados, J., De Ribaupierre, H., and Preece, A. (2020). Go simple and pre-train on domain-specific corpora: On the role of training data for text classification. In *Proceedings of the 28th international conference on computational linguistics*, pages 5522–5529.

[61] Erkan, G. and Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

[62] Esposito, M., Damiano, E., Minutolo, A., De Pietro, G., and Fujita, H. (2020). Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering. *Information Sciences*, 514:88–105.

[63] Faggioli, G. and Marchesin, S. (2021). What makes a query semantically hard? In Alonso, O., Marchesin, S., Najork, M., and Silvello, G., editors, *Proceedings of the Second International Conference on Design of Experimental Search & Information REtrieval Systems, Padova, Italy, September 15-18, 2021*, volume 2950 of *CEUR Workshop Proceedings*, pages 61–69. CEUR-WS.org.

[64] Faggioli, G., Zendel, O., Culpepper, J. S., Ferro, N., and Scholer, F. (2021). An enhanced evaluation framework for query performance prediction. In *European Conference on Information Retrieval*, pages 115–129. Springer.

[65] Farhi, S. H. and Boughaci, D. (2018). Graph based model for information retrieval using a stochastic local search. *Pattern Recognition Letters*, 105:234–239.

[66] Figueiredo, F., Rocha, L., Couto, T., Salles, T., Gonçalves, M. A., and Jr., W. M. (2011). Word co-occurrence features for text classification. *Inf. Syst.*, 36(5):843–858.

[67] Fontelo, P., Liu, F., and Ackerman, M. (2005). ask medline: a free-text, natural language query tool for medline/pubmed. *BMC medical informatics and decision making*, 5(1):5.

[68] Forbus, K. D., Gentner, D., and Law, K. (1995). Mac/fac: A model of similarity-based retrieval. *Cognitive science*, 19(2):141–205.

[69] Fox, W. and Bayat, M. S. (2008). *A guide to managing research*. Juta and company Ltd.

[70] Freeman, L. C., Borgatti, S. P., and White, D. R. (1991). Centrality in valued graphs: A measure of betweenness based on network flow. *Social networks*, 13(2):141–154.

[71] Galke, L., Saleh, A., and Scherp, A. (2017). Word embeddings for practical information retrieval. *INFORMATIK 2017*.

[72] Galkó, F. and Eickhoff, C. (2018). Biomedical question answering via weighted neural network passage retrieval. In *European Conference on Information Retrieval*, pages 523–528. Springer.

[73] Ganesh, S. and Varma, V. (2009). Exploiting the use of prior probabilities for passage retrieval in question answering. In *RANLP*, pages 99–102.

[74] Ganguly, D., Leveling, J., and Jones, G. J. (2011). Query expansion for language modeling using sentence similarities. In *Information Retrieval Facility Conference*, pages 62–77. Springer.

[75] Gao, L. and Callan, J. (2022). Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853, Dublin, Ireland. Association for Computational Linguistics.

[76] Ghasemi, N. and Hiemstra, D. (2021). BERT meets cranfield: Uncovering the properties of full ranking on fully labeled data. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 58–64, Online. Association for Computational Linguistics.

[77] Globerson, A., Chechik, G., Pereira, F., and Tishby, N. (2007). Euclidean embedding of co-occurrence data. *Journal of Machine Learning Research*, 8(Oct):2265–2295.

[78] Graham, S., Weingart, S., and Milligan, I. (2012). Getting started with topic modeling and mallet. Technical report, The Editorial Board of the Programming Historian.

[79] Griffiths, T. L. and Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235.

[80] Gu, Z. and Luo, M. (2004). Comparison of using passages and documents for blind relevance feedback in information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 482–483, New York, NY, USA. ACM.

[81] Guo, J., Fan, Y., Ai, Q., and Croft, W. B. (2016). A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 55–64.

[82] Guthrie, G. (2010). *Basic research methods: An entry to social science research.* SAGE Publications India.

[83] Hall, G. R. and Taghva, K. (2015). Using the web 1t 5-gram database for attribute selection in formal concept analysis to correct overstemmed clusters. In *2015 12th International Conference on Information Technology-New Generations*, pages 651–654. IEEE.

[84] Hamers, L. et al. (1989). Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. *Information Processing and Management*, 25(3):315–18.

[85] Handschuh, S., O'Riain, S., Thai, V., O'Sullivan, D., and Davis, B. (2008). Semantically enhanced passage retrieval for business analysis activity.

[86] Harman, D. W. (1986). An experimental study of factors important in document ranking. In *Proceedings of the 9th annual international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 186–193.

[87] Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

[88] Hauff, C., Hiemstra, D., and de Jong, F. (2008). A survey of pre-retrieval query performance predictors. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 1419–1420.

[89] He, B. and Ounis, I. (2004). Inferring query performance using pre-retrieval predictors. In *International symposium on string processing and information retrieval*, pages 43–54. Springer.

[90] He, J., Larson, M., and De Rijke, M. (2008). Using coherence-based measures to predict query difficulty. In *European conference on information retrieval*, pages 689–694. Springer.

[91] Hearst, M. A. (1997). Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33–64.

[92] Hearst, M. A. and Plaunt, C. (1993). Subtopic structuring for full-length document access. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 59–68. ACM.

[93] Hersh, W. R., Cohen, A. M., Roberts, P. M., and Rekapalli, H. K. (2006). Trec 2006 genomics track overview. In *TREC*, volume 7, pages 500–274.

[94] Hiemstra, D. (2000). A probabilistic justification for using tf$\times$ idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2):131–139.

[95] Hiemstra, D. (2001). *Using language models for information retrieval*. Citeseer.

[96] Hofstätter, S., Mitra, B., Zamani, H., Craswell, N., and Hanbury, A. (2021). Intra-document cascading: Learning to select passages for neural document ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1349–1358, New York, NY, USA. Association for Computing Machinery.

[97] Hollis, G., Westbury, C., and Lefsrud, L. (2017). Extrapolating human judgments from skip-gram vector representations of word meaning. *Quarterly Journal of Experimental Psychology*, 70(8):1603–1619.

[98] Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

[99] Huang, W., Trotman, A., and O'Keefe, R. A. (2006). Element retrieval using a passage retrieval approach. *Aust. J. Intell. Inf. Process. Syst.*, 9:80–83.

[100] Jae-Hyun Park, B. C. (2012). Passage-level evidence for cross-language information retrieval.

[101] Jafarzadeh, P. and Ensan, F. (2022). A semantic approach to post-retrieval query performance prediction. *Information Processing & Management*, 59(1):102746.

[102] Jong, M., Ri, C., Choe, H., and Hwang, C. (2015). A method of passage-based document retrieval in question answering system. *CoRR*, abs/1512.05437.

[103] Kalyanpur, A., Parsia, B., Sirin, E., Grau, B. C., and Hendler, J. (2006). Swoop: A web ontology editing browser. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(2):144–153.

[104] Kandylas, V., Upham, S. P., and Ungar, L. H. (2008). Finding cohesive clusters for analyzing knowledge communities. *Knowledge and Information Systems*, 17(3):335–354.

[105] Karim, M. R. (2018). *Java Deep Learning Projects: Implement 10 real-world deep learning applications using Deeplearning4j and open source APIs*. Packt Publishing Ltd.

[106] Kaszkiel, M. and Zobel, J. (2001). Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology*, 52(4):344–364.

[107] Keikha, M., Park, J. H., and Croft, W. B. (2014a). Evaluating answer passages using summarization measures. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 963–966. ACM.

[108] Keikha, M., Park, J. H., Croft, W. B., and Sanderson, M. (2014b). Retrieving passages and finding answers. In *Proceedings of the 2014 Australasian Document Computing Symposium*, page 81. ACM.

[109] Khodabakhsh, M. and Bagheri, E. (2021). Semantics-enabled query performance prediction for ad hoc table retrieval. *Information Processing & Management*, 58(1):102399.

[110] Khwileh, A., Way, A., and Jones, G. J. (2017). Improving the reliability of query expansion for user-generated speech retrieval using query performance prediction. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 43–56. Springer.

[111] Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

[112] Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184. IEEE.

[113] Kotov, A. and Zhai, C. (2011). Interactive sense feedback for difficult queries. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 163–172. ACM.

[114] Krikon, E., Carmel, D., and Kurland, O. (2012). Predicting the performance of passage retrieval for question answering. In *Proceedings of the 21st ACM International Conference on Information and Knowledge management*, pages 2451–2454.

[115] Krikon, E., Kurland, O., and Bendersky, M. (2010). Utilizing inter-passage and inter-document similarities for reranking search results. *ACM Transactions on Information Systems (TOIS)*, 29(1):3.

[116] Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50.

[117] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

[118] Kurland, O. (2014). The cluster hypothesis in information retrieval. In *European Conference on Information Retrieval*, pages 823–826. Springer.

[119] Kurland, O. and Lee, L. (2006). Respect my authority! hits without hyperlinks, utilizing cluster-based language models. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 83–90.

[120] Kurland, O. and Lee, L. (2010). Pagerank without hyperlinks: Structural reranking using links induced by language models. *ACM Transactions on Information Systems (TOIS)*, 28(4):1–38.

[121] Kuzi, S., Shtok, A., and Kurland, O. (2016). Query expansion using word embeddings. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 1929–1932. ACM.

[122] Lashkari, A. H., Mahdavi, F., and Ghomi, V. (2009). A boolean model in information retrieval for search engines. In *Information Management and Engineering, 2009. ICIME'09. International Conference on*, pages 385–389. IEEE.

[123] Lee, H.-g., Kim, M., Kim, H., Kim, J., Kwon, S., Seo, J., Kim, Y.-r., and Choi, J.-K. (2016). Ksanswer: question-answering system of kangwon national university and sogang university in the 2016 bioasq challenge. In *Proceedings of the Fourth BioASQ workshop*, pages 45–49.

[124] Lee, J. H. (1994). Properties of extended boolean models in information retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 182–190. Springer-Verlag New York, Inc.

[125] Levy, O. and Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.

[126] Li, C., Yates, A., MacAvaney, S., He, B., and Sun, Y. (2020). PARADE: passage representation aggregation for document reranking. *CoRR*, abs/2008.09093.

[127] Li, L., Shang, Y., and Zhang, W. (2002). Improvement of hits-based algorithms on web documents. In *Proceedings of the 11th international conference on World Wide Web*, pages 527–535.

[128] Li, X. and Chen, E. (2010). Graph-based answer passage ranking for question answering. In *Computational Intelligence and Security (CIS), 2010 International Conference on*, pages 634–638. IEEE.

[129] Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain.

[130] Lin, C.-Y. and Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 605. Association for Computational Linguistics.

[131] Lin, J., Quan, D., Sinha, V., Bakshi, K., Huynh, D., Katz, B., and Karger, D. R. (2003). The role of context in question answering systems. In *CHI'03 Extended Abstracts on Human Factors in Computing Systems*, pages 1006–1007.

[132] Liu, S., Liu, F., Yu, C., and Meng, W. (2004). An effective approach to document retrieval via utilizing wordnet and recognizing phrases. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 266–272.

[133] Liu, T.-Y. (2011). Learning to rank for information retrieval.

[134] Liu, T.-Y. et al. (2009). Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331.

[135] Liu, X. and Croft, W. B. (2002). Passage retrieval based on language models. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 375–382. ACM.

[136] Lund, K. and Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

[137] Lv, Y. and Zhai, C. (2010). Positional relevance model for pseudo-relevance feedback. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 579–586.

[138] Malliaros, F. D. and Skianis, K. (2015). Graph-based term weighting for text categorization. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1473–1479.

[139] Metzler, D. and Croft, W. B. (2005). A markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 472–479. ACM.

[140] Miao, Y.-Q. and Kamel, M. (2011). Pairwise optimized rocchio algorithm for text categorization. *Pattern Recognition Letters*, 32(2):375–382.

[141] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In Bengio, Y. and LeCun, Y., editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

[142] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[143] Mitra, B. and Craswell, N. (2019). An updated duet model for passage re-ranking. *CoRR*, abs/1903.07666.

[144] Mitra, M., Singhal, A., and Buckley, C. (1998a). Improving automatic query expansion. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 206–214, New York, NY, USA. ACM.

[145] Mitra, M., Singhal, A., and Buckley, C. (1998b). Improving automatic query expansion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 206–214. ACM.

[146] Monz, C. (2004). Minimal span weighting retrieval for question answering. In *Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering*, volume 2.

[147] Mothe, J. and Tanguy, L. (2005). Linguistic features to predict query difficulty. In *ACM Conference on research and Development in Information Retrieval, SIGIR, Predicting query difficulty-methods and applications workshop*, pages 7–10.

[148] Na, S.-H., Kim, J., Kang, I.-S., and Lee, J.-H. (2008). Exploiting proximity feature in bigram language model for information retrieval. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 821–822.

[149] Nogueira, R. F. and Cho, K. (2019). Passage re-ranking with BERT. *CoRR*, abs/1901.04085.

[150] O'Connor, J. (1980). Answer-passage retrieval by text searching. *Journal of the American Society for Information Science*, 31(4):227–239.

[151] Oh, H.-S. and Jung, Y. (2015). Cluster-based query expansion using external collections in medical information retrieval. *Journal of biomedical informatics*, 58:70–79.

[152] Otterbacher, J., Erkan, G., and Radev, D. R. (2005). Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 915–922. Association for Computational Linguistics.

[153] Otterbacher, J., Erkan, G., and Radev, D. R. (2009). Biased lexrank: Passage retrieval using random walks with question-based priors. *Information Processing & Management*, 45(1):42–54.

[154] Oussalah, M. and Kostakos, P. (2017). On web based sentence similarity for paraphrasing detection. In *KDIR*, pages 289–292.

[155] Padaki, R., Dai, Z., and Callan, J. (2020). Rethinking query expansion for bert reranking. *Advances in Information Retrieval*, 12036:297.

[156] Padigela, H., Zamani, H., and Croft, W. B. (2019). Investigating the successes and failures of BERT for passage re-ranking. *CoRR*, abs/1905.01758.

[157] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

[158] Pang, L., Lan, Y., Guo, J., Xu, J., Wan, S., and Cheng, X. (2016). Text matching as image recognition. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[159] Peat, H. J. and Willett, P. (1991). The limitations of term co-occurrence data for query expansion in document retrieval systems. *Journal of the american society for information science*, 42(5):378–383.

[160] Pérez, R. A. and Pagola, J. E. M. (2010). An incremental text segmentation by clustering cohesion. *HaCDAIS 2010*, page 65.

[161] Pérez-Iglesias, J. and Araujo, L. (2010). Standard deviation as a query hardness estimator. In *International Symposium on String Processing and Information Retrieval*, pages 207–212. Springer.

[162] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

[163] Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281. ACM.

[164] Prim, R. C. (1957). Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401.

[165] Raiber, F. and Kurland, O. (2014). Query-performance prediction: setting the expectations straight. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 13–22.

[166] Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

[167] Ren, R., Lv, S., Qu, Y., Liu, J., Zhao, W. X., She, Q., Wu, H., Wang, H., and Wen, J. (2021). PAIR: leveraging passage-centric similarity relation for improving dense passage retrieval. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2173–2183. Association for Computational Linguistics.

[168] Renoust, B., Melançon, G., and Viaud, M.-L. (2013). Measuring group cohesion in document collections. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 01*, pages 373–380. IEEE Computer Society.

[169] Riedl, M. and Biemann, C. (2012). Topictiling: a text segmentation algorithm based on lda. In *Proceedings of ACL 2012 Student Research Workshop*, pages 37–42. Association for Computational Linguistics.

[170] Roberts, I. and Gaizauskas, R. (2004). Evaluating passage retrieval approaches for question answering. In *European Conference on Information Retrieval*, pages 72–84. Springer.

[171] Robertson, S., Zaragoza, H., et al. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

[172] Robertson, S. E. (1990). On term selection for query expansion. *Journal of documentation*.

[173] Robertson, S. E. and Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146.

[174] Robertson, S. E., van Rijsbergen, C. J., and Porter, M. F. (1980). Probabilistic models of indexing and searching. In *SIGIR*, volume 80, pages 35–56.

[175] Rocchio, J. J. (1971). Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*.

[176] Roitman, H. (2018a). An extended query performance prediction framework utilizing passage-level information. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 35–42.

[177] Roitman, H. (2018b). Query performance prediction using passage information. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 893–896.

[178] Roitman, H., Erera, S., and Weiner, B. (2017). Robust standard deviation estimation for query performance prediction. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 245–248.

[179] Rosenfeld, R. (2000). Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278.

[180] Rousseau, F. and Vazirgiannis, M. (2013). Graph-of-word and tw-idf: new approach to ad hoc ir. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 59–68. ACM.

[181] Roy, N. (2019). Word embedding models for query expansion in answer passage retrieval. Master's thesis.

[182] Salton, G., Allan, J., and Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–58. ACM.

[183] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

[184] Salton, G. and Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American society for information science*, 41(4):288–297.

[185] Salton, G., Fox, E. A., and Wu, H. (1983). Extended boolean information retrieval. *Communications of the ACM*, 26(11):1022–1036.

[186] Salton, G. and McGill, M. J. (1986). Introduction to modern information retrieval.

[187] Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

[188] Sarwar., G. and Bradshaw., S. (2018). Investigating the use of semantic relatedness at document and passage level for query augmentation. In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - KDIR,*, pages 237–244. INSTICC, SciTePress.

[189] Sarwar, G. and O'Riordan, C. (2021). Passage based answer-set graph approach for query performance prediction. In *Proceedings of the 25th Australasian Document Computing Symposium*, ADCS '21, New York, NY, USA. Association for Computing Machinery.

[190] Sarwar, G., O'Riordan, C., and Newell, J. (2017). Passage level evidence for effective document level retrieval. pages 83–90.

[191] Sarwar, G., O'Riordan, C., and Newell, J. (2019). Investigation of passage based ranking models to improve document retrieval. In Fred, A., Aveiro, D., Dietz, J. L. G., Liu, K., Bernardino, J., Salgado, A., and Filipe, J., editors, *Knowledge Discovery, Knowledge Engineering and Knowledge Management*, pages 100–117, Cham. Springer International Publishing.

[192] Sarwar., G. and O'Riordan., C. (2021). A graph-based approach at passage level to investigate the cohesiveness of documents. In *Proceedings of the 10th International Conference on Data Science, Technology and Applications - DATA,*, pages 115–123. INSTICC, SciTePress.

[193] Selvaretnam, B. and Belkhatir, M. (2012). Natural language technology and query expansion: issues, state-of-the-art and perspectives. *Journal of Intelligent Information Systems*, 38(3):709–740.

[194] Seymore, K. and Rosenfeld, R. (1997). Using story topics for language model adaptation. In *Fifth European Conference on Speech Communication and Technology*.

[195] Shah, C. and Croft, W. B. (2004). Evaluating high accuracy retrieval techniques. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 2–9.

[196] Shahi, D. (2016). *Apache Solr*. Springer.

[197] Sheetrit, E. and Kurland, O. (2019). Cluster-based focused retrieval. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2305–2308.

[198] Sheetrit, E., Shtok, A., and Kurland, O. (2020). A passage-based approach to learning to rank documents. *Information Retrieval Journal*, 23(2):159–186.

[199] Sheetrit, E., Shtok, A., Kurland, O., and Shprincis, I. (2018). Testing the cluster hypothesis with focused and graded relevance judgments. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1173–1176.

[200] Shtok, A., Kurland, O., and Carmel, D. (2016). Query performance prediction using reference lists. *ACM Transactions on Information Systems (TOIS)*, 34(4):1–34.

[201] Shtok, A., Kurland, O., Carmel, D., Raiber, F., and Markovits, G. (2012). Predicting query performance by query-drift estimation. *ACM Transactions on Information Systems (TOIS)*, 30(2):1–35.

[202] Singhal, A., Buckley, C., and Mitra, M. (2017). Pivoted document length normalization. In *ACM SIGIR Forum*, volume 51, pages 176–184. ACM New York, NY, USA.

[203] Singhal, A., Salton, G., Mitra, M., and Buckley, C. (1996). Document length normalization. *Information Processing & Management*, 32(5):619–633.

[204] Srikanth, M. and Srihari, R. (2002). Biterm language models for document retrieval. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–426.

[205] Strohman, T., Metzler, D., Turtle, H., and Croft, W. B. (2005). Indri: A language model-based search engine for complex queries. In *Proceedings of the international conference on intelligent analysis*, volume 2, pages 2–6. Citeseer.

[206] Tan, J., Wan, X., and Xiao, J. (2017). Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

[207] Thakker, D., Osman, T., and Lakin, P. (2009). Gate jape grammar tutorial. *Nottingham Trent University, UK, Phil Lakin, UK, Version*, 1.

[208] Thammasut, D. and Sornil, O. (2006). A graph-based information retrieval system. In *2006 International Symposium on Communications and Information Technologies*, pages 743–748. IEEE.

[209] Thesprasith, O. and Jaruskulchai, C. (2014). Query expansion using medical subject headings terms in the biomedical documents. In *Asian Conference on Intelligent Information and Database Systems*, pages 93–102. Springer.

[210] Tiedemann, J. and Mur, J. (2008). Simple is best: experiments with different document segmentation strategies for passage retrieval. In *Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering*, pages 17–25.

[211] Tomović, A., Janičić, P., and Kešelj, V. (2006). n-gram-based classification and unsupervised hierarchical clustering of genome sequences. *Computer methods and programs in biomedicine*, 81(2):137–153.

[212] Truong, Q. D., Dkaki, T., Mothe, J., and Charrel, P.-J. (2008). Gvc: a graph-based information retrieval mode. In *CORIA*, pages 337–351.

[213] Van Rijsbergen, C. J. (1979). Information retrieval.

[214] Van Rijsbergen, C. J. and Croft, W. B. (1975). Document clustering: An evaluation of some experiments with the cranfield 1400 collection. *Information Processing & Management*, 11(5-7):171–182.

[215] Vechtomova, O. and Karamuftuoglu, M. (2008). Lexical cohesion and term proximity in document ranking. *Information Processing & Management*, 44(4):1485–1502.

[216] Vinay, V., Cox, I. J., Milic-Frayling, N., and Wood, K. (2006). On ranking the effectiveness of searches. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 398–404, New York, NY, USA. ACM.

[217] Voorhees, E. M. (1985). The cluster hypothesis revisited. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 188–196.

[218] Voorhees, E. M. (2003). Overview ofthe trec 2003 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC2003)*.

[219] Voorhees, E. M. et al. (1999). The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82.

[220] Wade, C. and Allan, J. (2005). Passage retrieval and evaluation. Technical report, MASSACHUSETTS UNIV AMHERST CENTER FOR INTELLIGENT INFORMATION RETRIEVAL.

[221] Wan, X.-J. and Peng, Y.-X. (2005). A new retrieval model based on texttiling for document similarity search. *Journal of Computer Science and Technology*, 20(4):552–558.

[222] Wang, B., Wang, A., Chen, F., Wang, Y., and Kuo, C.-C. J. (2019). Evaluating word embedding models: methods and experimental results. *APSIPA Transactions on Signal and Information Processing*, 8.

[223] Witschel, H. F. (2008). Global term weights in distributed environments. *Information processing & management*, 44(3):1049–1061.

[224] Witten, I. H., Witten, I. H., Moffat, A., Bell, T. C., Bell, T. C., and Bell, T. C. (1999). *Managing gigabytes: compressing and indexing documents and images*. Morgan Kaufmann.

[225] Xu, Y., Jones, G. J., Li, J., Wang, B., and Sun, C. (2007). A study on mutual information-based feature selection for text categorization. *Journal of Computational Information Systems*, 3(3):1007–1012.

[226] Yagcioglu, S., Erdem, E., Erdem, A., and Cakıcı, R. (2015). A distributed representation based query expansion approach for image captioning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 106–111.

[227] Yang, G. H. and Soboroff, I. (2016). Trec 2016 dynamic domain track overview. In *TREC*.

[228] Yang, L., Ai, Q., Spina, D., Chen, R.-C., Pang, L., Croft, W. B., Guo, J., and Scholer, F. (2016). Beyond factoid qa: effective methods for non-factoid answer sentence retrieval. In *European Conference on Information Retrieval*, pages 115–128. Springer.

[229] Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Icml*, volume 97, page 35. Citeseer.

[230] Yenala, H., Kamineni, A., Shrivastava, M., and Chinnakotla, M. K. (2015). Iiith at bioasq challange 2015 task 3b: Bio-medical question answering system. In *CLEF (Working Notes)*.

[231] Yulianti, E., Chen, R.-C., Scholer, F., Croft, W. B., and Sanderson, M. (2017). Document summarization for answering non-factoid queries. *IEEE transactions on knowledge and data engineering*, 30(1):15–28.

[232] Zamani, H., Croft, W. B., and Culpepper, J. S. (2018). Neural query performance prediction using weak supervision from multiple signals. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 105–114.

[233] Zendel, O., Shtok, A., Raiber, F., Kurland, O., and Culpepper, J. S. (2019). Information needs, queries, and query performance prediction. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 395–404.

[234] Zhang, B., Li, H., Liu, Y., Ji, L., Xi, W., Fan, W., Chen, Z., and Ma, W.-Y. (2005). Improving web search results using affinity graph. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 504–511. ACM.

[235] Zhang, D. and Lee, W. S. (2003). A language modeling approach to passage question answering. In *TREC*, pages 489–495.

[236] Zhao, Y., Scholer, F., and Tsegay, Y. (2008). Effective pre-retrieval query performance prediction using similarity and variability evidence. In *European conference on information retrieval*, pages 52–64. Springer.

[237] Zhao, Z., Liu, T., Li, S., Li, B., and Du, X. (2017). Ngram2vec: Learning improved word representations from ngram co-occurrence statistics. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 244–253.

[238] Zheng, Z., Hui, K., He, B., Han, X., Sun, L., and Yates, A. (2020). BERT-QE: contextualized query expansion for document re-ranking. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4718–4728. Association for Computational Linguistics.

[239] Zhou, W., Torvik, V. I., and Smalheiser, N. R. (2006). Adam: another database of abbreviations in medline. *Bioinformatics*, 22(22):2813–2818.

[240] Zhou, W., Yu, C., Smalheiser, N., Torvik, V., and Hong, J. (2007). Knowledge-intensive conceptual retrieval and passage extraction of biomedical literature. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 655–662. ACM.

[241] Zhou, Y. and Croft, W. B. (2007). Query performance prediction in web search environments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 543–550.