T.R.
EGE UNIVERSITY
Graduate School of Applied and Natural Science

AN INVESTIGATION ON TERM WEIGHTING METHODS FOR DISTRIBUTED INFORMATION RETRIEVAL

MSc THESIS

SHANAY ZARGARI AFSHAR

INTERNATIONAL COMPUTER INSTITUTE

İzmir
2021

T.R.
EGE UNIVERSITY
Graduate School of Applied and Natural Science

AN INVESTIGATION ON TERM WEIGHTING METHODS FOR DISTRIBUTED INFORMATION RETRIEVAL

SHANAY ZARGARI AFSHAR

Supervisor : Asst. Prof. İlker KOCABAŞ

International Computer Institute
Information Technologies Second Cycle Programme

İzmir
2021

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Yüksek Lisans Tezi olarak sunduğum "Dağıtık Bilgi Erişim Sistemleri İçin Terim Ağırlıklandırma Yöntemleri Üzerine Bir İnceleme" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

30.07.2021

Shanay Zargari Afshar

ÖZET

Dağıtık Bilgi Erişim Sistemleri İçin Terim Ağırlıklandırma Yöntemleri Üzerine Bir İnceleme

ZARGARI AFSHAR, Shanay

Yüksek Lisans Tezi, Uluslararası Bilgisayar Enstitüsü
Tez Danışmanı: Dr. Öğr. Üyesi İlker KOCABAŞ
Temmuz 2021, 40 sayfa

Son zamanlarda bilgiler ve deneysel gerçekler geçmişe göre daha fazla öneme sahiptir. Bununla beraber, teknolojinin gelişmesiyle kullanıcıların ihtiyaç duyduğu bilgiler günden güne ciddi bir şekilde artmaktadır. Bilgi erişim sisteminde, kullanıcılar tarafından erişilen bilgiler, aslında ağda başka bir kaynaktan erişilmektedir. Genellikle bir arama motoru olarak bilinen bilgi erişim sistemi bazı temel işlemlere sahiptir. Örnek olarak bilgi erişiminden önce web sitelerinde bilgi tarama ve indekslemeyi yapmak zorundadır. Bu günlerde, herhangi bir kullanıcı arama motorunu kullanarak herhangi bir web servisinde istediği bir bilgiye ulaşabilir. Bu araştırmanın amacı, bilgi geri getirim süresince, yazılmış metin kaynaklarının üzerinde, terim ağırlıklandırma , birleştirme ve skor normalizasyonu tekniklerinin performansını incelemektedir. Dağıtık bir ortamdaki bilgi erişim sistemi, her bir bilgi kaynağının sonuç kümesinde skorlama , sıralama ve bunun yanı sıra aynı kaynakların sonuçlarının birleştirilmesinden oluşur. Bu çalışmada, TERRIER'in kütüphanesini kullanarak bir uygulama geliştirildi ve bunun sonucunda deneylerin ve testlerin otomatik bir şekilde yapılmasını sağladı. Glasgow üniversitesi bilgi erişim üyeleri tarafından geliştirilmiş olan TERRIER, açık kaynak modüler yazılım platformu olduğundan, büyük çapta bilgi erişim sistemlerinin gelişimi için kullanılmıştır. Bu programın işlevleri test, indeksleme, bilgi erişimi ve performans değerlendirmesi için veri kümesi hazırlama aşamalarından oluşur. İlk olarak, Bilgi erişim konferansı tarafından hazırlanmış TREC-6, TREC-7 ve TREC-8 ad-hoc parçalarını kullanarak bir veri kümesi hazırlanmıştır. Her bir veri koleksiyonu rastgele bir şekilde daha küçük veri kümelerine bölünüp, her biri ayrı bir veri kaynağı olarak düzenlendi. İndeksleme aşamasından sonra veriler bilgi erişim aşaması için hazırlanmıştır. Test senaryoları, bilgi erişim aşamasında, her bir test veri kümesi üzerinde, ağırlıklandırma ve normalizasyon birleştirme yöntemlerini kullanarak olşturuldu.

Ağılıklandırma teknikleri olarak BM25, TF-IDF, IFB2 ve DFIZ kullanıldı. Normalleştirme yöntemleri olarak da Min-Max ve Sum kullanıldı. En sonunda oluşmuş test kümelerinin performansı R-Prec ve MAP metriklerini inceleyerek tespit edildi.

Anahtar Kelimeler: Dağıtık Bilgi Erişim, Ağırlıklandırma algoritmaları, indeksleme

ABSTRACT

An Investigation On Term Weighting Methods For Distributed Information Retrieval

ZARGARI AFSHAR, Shanay

MSc in Information Technologies
Supervisor: Asst. Prof. İlker KOCABAŞ
July , 40 pages

In recent years information and facts has become more of a priority than long ago. In addition to technical developments in computer science, the amount of information needed by users has been increasing day by day. In the context of information retrieval (IR), the information that has been accessed is actually retrieved from a source located somewhere else in the network. An IR system, often referred to as a search engine, has to do some basic operations such as crawling the information sources and indexing before retrieving requested information. Nowadays any client can access any related information in any web service using search engines. The objective of this research is to examine the performance of retrieval on written document collections in terms of term weighting algorithms and score normalization merging techniques in a distributed environment called distributed information retrieval. In such a distributed environment, information retrieval consists of document scoring and ranking each information source participated in a result-set as well as the final merging of these results that have been returned from different sources. During this study, an application was developed by using the library provided by Terrier IR platform, so that experiments can be carried out automatically. Terrier IR Platform is a modular open source software for the rapid development of large scale IR applications which was developed by members of the Information Retrieval Research Group, Department of Computing Science, at the University of Glasgow. The functions performed by this program are data-sets preparation for tests, indexing, retrieving and performance evaluation. Firstly, an experimental test data has been prepared using text collections in TREC-6, TREC-7 and TREC-8 ad-hoc tracks provided by the Text Retrieval Conference. Each test collection is divided randomly and discretely into smaller-sized collections that are each taken as documents owned by a source in distributed environment. After the indexing process, the necessary data has

become ready for the use of test cases in the retrieval stage. The test cases have been created by applying several term weighting methods and normalized merging methods at retrieval stage on each dataset. TF-IDF, BM25 are examples of term weighting methods, where Min-Max and Sum are of result merging methods. At last, the performances of tests have been evaluated by the MAP and R-Prec metrics.

Keywords: Distributed Information Retrieval, Weighting Algorithms, Indexing

FOREWORD

This thesis has been prepared as part of my studies to obtain a Master's degree in Information Technologies at the International Computer Institute of Ege University. I have been authorized to start my thesis work in the fall semester of the year 2019-2020 with the completion of my courses. Being a master's student at EGE University in the Information Technology field truly broadened my horizons. Reading numerous articles about information retrieval, weighting algorithms and my professor's considerate advice has turned me into a persevering scholar. Fortunately, my professor was always available and willing to answer my queries. In brief, every course I have attended, every short research I have conducted throughout my Master's studies has contributed to enriching this work.

İZMİR

30.07.2021

Shanay Zargari Afshar

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The growing amount of data which cannot be explored and processed by traditional computing techniques have led to development of the information retrieval (IR) systems. The required information has to be explored and investigated by a more advanced approach called IR systems, known as search engines. The retrieving of information is considered as searching a single database while distributed information retrieval is based on searching and exploring multiple databases. Handling large amount of data stored in a single database can be a very overwhelming task. This issue was the reason of emerging distributed information retrieval (DIR) systems. In the multi-database model of distributed information retrieval, the queries are being processed independently in each divided smaller-sized collection. The resulted documents which are generated, are then combined to form a single result-set. After retrieval is carried out, the results are saved in the form of a document or a set of ducuments that are ranked according to how relevant they are to the user's query. The process is s series of tasks that are conducted separately and is performed on data collections to show how well the operation has satisfied the terms formed by the user. Distributed Informatin Retrieval comprises techniques which are crucial for the retrieving task, including indexing and retrieving the data collections. In this research, we aim to carry out some experiments on weighting models in DIR environment and develop a system to do the retrieval task automatically by deploying the TREC test data collection and modular search engine called TERRIER. This program accelerates the experiment process of retrieving information in a distributed environment for TREC6, 7 and 8. The program makes the TREC data collection to be divisible to the desired number of sub-collections. All things considered, the retrieving task is performed on each sub-collection to examine and evaluate the results.

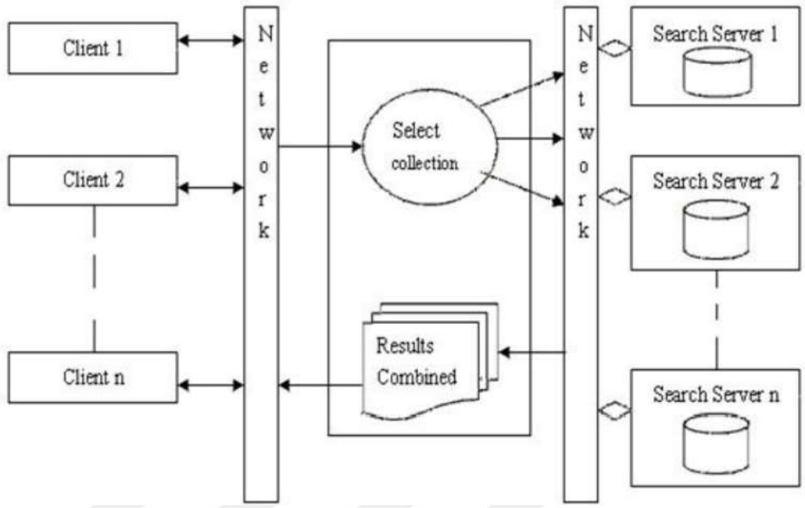

Figure 1.1: Distributed Information Retrieval Architecture(Ounis et al., 2007)

First we outline some pre-processing steps of IR systems. Through the process of information retrieval, sometimes some commonly used words have less effect on the result of the matching documents returned to satisfy the query. These low importance words are called stop-words. Stop-words are a groups of words like determiners, prepositions or sometimes adjectives that are beneficial to be removed from the process since their removal will speed up the process. However, removing adjectives for some procedures can be misleading. So in this situation, it is better to only eliminate determiners and prepositions according to the need of that process. One other important step in IR is stemming. Stemming is the process of removing suffixes to reduce them to the roots. It is one of the major steps in IR systems and errors made by the stemmer can mainly affect the information retrieval performance (Yom-Tov et al., 2005) . Stemming is also used to increase the effectiveness of retrieval and consists of different algorithms such as Table Look up approach, Successor Variety, N-Gram Stemmers and Affix Removal Stemmers. The last pre-processing task which affects the process of retrieval by finding the number of terms in the data collection is tokenization. Tokenizing the data helps to satisfy the query and makes the search process flow sharply and accurately (Danzig et al., 1991) . In this process the documents are explored in advance to produce the tokens, which are used in indexing process and are the basis of Information Retrieval procedure.

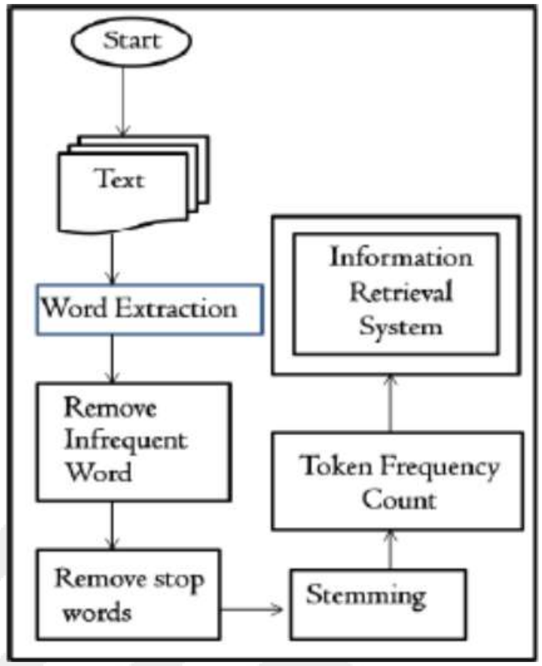

Figure 1.2: Tokenization Process(Danzig et al., 1991)

Figure 1.2 shows that pre-processing tasks include tokenization, stemming and then stop word removal. In the following we explore the function of weighting algorithms. Some mostly utilized weighting algorithms in the field are Okapi BM25 (Géry & Largeron, 2012) , DFIZ(Divergence from independence), IFB2(Inverse Term Frequency model for randomness), DFR(Divergence from randomness), TF-IDF(Term Frequency Inverse Document Frequency). By applying these weighting models, each term is assigned an individual weight in the document. TF-IDF is one of the most useful weighing algorithms that assigns a weight to each term in the related document and inverse proportion to the number of documents which the term is obtainable from. TF-IDF consists of two parts, the first one is term frequency and the other one is inverse document frequency. When a term is used more frequently in a document, that word is considered as a more crucial element in that document (WOSP 2020 Proceedings of the 8th International Workshop on Mining Scientific Publications Wuhan , China, 2020). To sum up, TF-IDF's value increases in accordance with the number of times a word is repeated in that document. BM25 is also a well-known weighting model that is mostly used in information retrieval. BM25, which is based on a probabilistic evaluation technique, is mostly used to score documents in information retrieval process. In this work BM25, TF-IDF, DFIZ and IFB2 weighting models are used for scoring the documents. DIR systems are categorized in agreement with where their related indexes are kept. These categories include peer to peer, broker-based and crawling systems. In peer-to-peer architecture the indexes are placed with data resources and queries are forwarded to the resources and results are merged by the peer that has assigned the query.

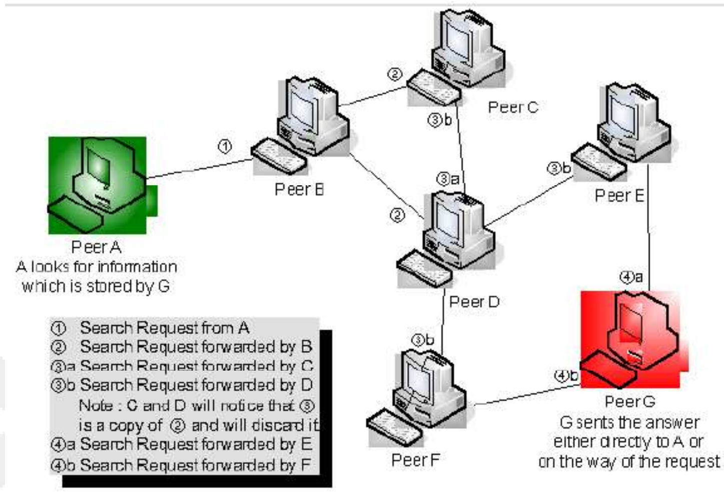

Figure 1.3: Peer-to-Peer Architecture

On the other hand, in broker-based systems indexes are placed with the resources. In this system, the queries are addressed to the resources and there is a peer agent which acts as broker and merges the results which have been returned. In this architecture, the resource description should be provided for each resource and should be managed by the broker agent.

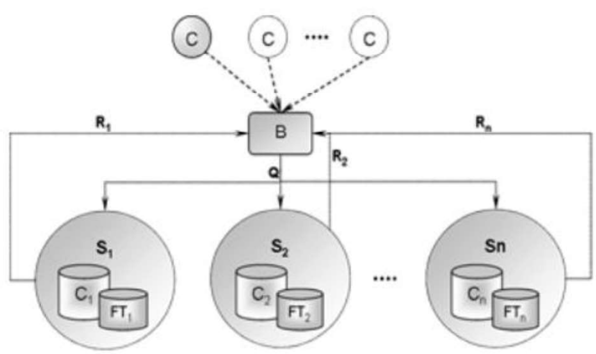

Figure 1.4: Broker-Based Architecture

The environments that are required for resource description execution fall into two categories as cooperative environment and uncooperative environment. In a cooperative environment, there is full access to the resources and their indexes, whereas in an uncooperative environment, there is limitation on access to the resources and indexes. While retrieving from multiple resources, all resources can be investigated, or some documents can be selected among them. So, resource selection task is a method to decide which documents should the query be addressed to. In other words, in this method the documents that contain the most relevant information to the user's query are chosen to be searched (Cetintas & Si, 2011). However, in this work resource selection has not been carried out and all resources have been explored and retrieved so we are not going through more details of this task.

2. RELATED RESEARCH

As we know, investigation and exploring the related information through the resources need some level of interaction between researcher and the system. The interaction encompasses relation between the researcher and the system, the system and the data collection and the researcher and data collection. In late 1950s, the use of the term information retrieval emerged when retrieving the information from data collections appeared for the first time (Cool & Belkin, 2018).

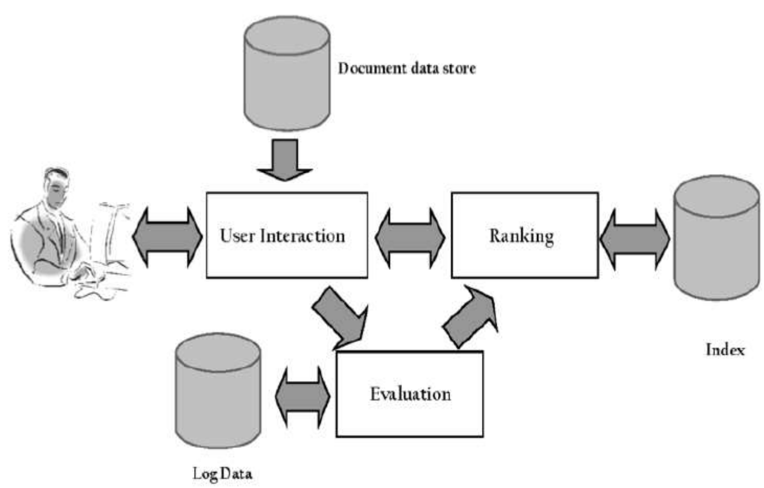

Figure 2.1: User Interaction and Query Process

The need for information retrieval systems has risen when the collected data size reached a limit where extracting the related information for the desired query became impossible by former techniques. IR systems were first used in commercial and intelligence applications. The development of IR systems first started from manual library-based techniques of indexing and retrieving of the documents. In this process, IR systems started to Show progress where tasks cruical for IR systems started to become automated (Sanderson & Croft, 2012a). IR systems are categorized as centralized and distributed systems. In centralized systems, documents are available and managed by a single system. On the other hand, the distributed system handles the documents which reside on different systems and resources. The distributed architecture processes the information by addressing the queries to the specified systems. The results of the retrieval in DIR systems are merged and unified. Thus, merging techniques are cruical and must be developed to attain a better result (Zhu & Gauch, 2000). Speaking of DIR systems, in Jinxi Xu and Jamie Callan's work, the retrieval effectiveness of DIR systems in realistic environment is the objective of this work. Their work targeted to indicate that the performance of a system falls when there are large number of collections in comparison to a centralized system (Xu & Callan, 1998). Another work proposing a multi-database DIR system is by Jamie Callan . In his work IR consists of discovering the databases, ranking the databases and merging the results. In this model Of IR systems controlling the data is more complicated when data is retrieved from a single database (Callan, 2005). The complex tasks cruical for DIR systems are as follows.

Resource description: It consists of describing each data collection.

Resource selection: Selecting the data colllections according to the submitted query. Resource description plays a cruical role in selecting the data collections to be searched.

Result merging: Integration and sorting the ranked list returned by each data-collection to form a single ranked list.

Performing any retrieval task on data collections must be planned in advanced. The objective of the experiment and used test data collections must be defined. The approaches that are going to be used should be assigned. So, the results obtained from the designed system are ready to be evaluated. Evaluation of IR systems is crucial for developing high quality retrieval systems that meets users' needs (Clough & Sanderson, 2013).

A method of retrieving the information in DIR environment is by sumbitting a query where a receptionist examines the query and passes it to the librarians. Each librarian evaluates the query according to the local information it has access to. Later, each evaluation result is returned to the receptionist. The receptionist finally merges the results and demonstrates the k top results to the user(de Kretser et al., 1998).

In DIR systems, broker-based and peer-to-peer architectures are glaring when speaking of decentralized architectures. Each architecture has its own disadvantages. This is the work of Abdel Naser Pouamoun and İlker kocabaş where they have designed a broker-based peer-to-peer system which is based on agents. In this designed system, the broker holds the list of available agents and the agents which contain resources. Peers joining and leaving the system are managed by the broker agent. In this system the information transfer among peers is handled by the broker agent (Pouamoun, 2021).

The work of Prof. Y M Naik, Shilpa Tarihal, Roopali Swami, Ashwini Purandare, Kiran Adike is to solve the problem of information overload on a single database. The novel approach of using mobile agents to be dispatched to the data source to be processed remotely is one of the examples of retrieving information in a distributed environment (Tarihal et al., 2013).

Another novel system which is based on agents and Semantic web is porposed by Khaled M. Fouad, Ahmed R. Khalifa, Nagdy M. Nagdy and Hany M. Harb. This system which is called SPIRS improves accuracy and effectiveness when retrieving the web collections. This online system supplies related web documents to the submitted query (Fouad et al., 2012).

Search engines are softwares that retrieve information from the data collections. The users in search engines interact with the user interface to find the desired information. One such mechanism which was for searching a pattern through the roll of microfilms was invented by Emanuel Golberg (Sanderson & Croft, 2012a). One of the earliest search engines is Archie which was created by Alan Emtag (Seymour et al., 2011). Archie was in nature a database containing file directories from systems. The result of the search in Archie was the directory path on the specific system which the file was available on (Seymour et al., 2011). Gopher (Seymour et al., 2011) was another tool which was a menu system. In this system, the list of files were visible for users who connected to the server using Gopher on remote computers.

Todays, search engines have improved significantly where they can present videos, images beside text information. Such search engines have improved in performance where they can find results which are more similar to the desired queries. Google's search engine could outrank other similar comanies in the domain by initiating the PageRank algorithm that was based on the number of webpages which contained more links. Google also provided a simpler interface (Seymour et al., 2011). An agent based complex querying engine has been proposed by Subrata das, Kurt Shuster, Curt Wu and Igor levit in 2005 which could overcome the many limitations of IR, such as large amount of data being heterogeneous and distributed. This system translates each query to several sub-queries, and at the same time the system consists several agents which contain the retrieval code. The agents are then dispatched to process each sub-query remotely. The retrieved data then gets merged to form a single list of documents that satisfies the query (Das et al., 2005). Search engines have become one of the inseparable elements of the internet which help users to navigate to specific websites. One of the search engines that has affected the retrieval performance of web search engines is Sponsored search engine. The sponsored search engine supports crawling and indexing of many web pages and documents, and at the same time it accepts millions of queries. Such search engines are developed to find the desired and most relevant documents and information. With the advent of the Sponsored search engine, known search engines like Yahoo, Google and Ask have the online commerce (Jansen & Mullen, 2008).

Search engines are divided into two groups as open source and commercial search engines. While Google and Yahoo are categorized as commercial search engines, there are open source search engines such as MozDex and Indri. Mozdex has been enhanced by using different open-source technologies. Indri is an open source search engine which integrates best characteristics of inference nets and language modeling where it also supports features such as, accepting large databases, fast indexing (Strohman et al., 2005). Beside these open-source search engines which have been mentioned, there are some other search engines that are crucial for information retrieval purposes. One of these search engines is TERRIER which is used in this work.

Evaluation of the results in the retrieval process is crucial. The measures mostly used in evaluation of information retrieval systems are precision and recall values that are taken into consideration in this work. These evaluation measures were first proposed by Kent et al (Saracevic, 1995) for evaluation of IR systems. He proposed relevance as the basic value to evaluate the retrieval systems. Relevance was then a benchmark to evaluate data in many research experiments. But later, beside relevance value, some other measures were considered for evaluation. Finally, those benchmarks did not last long, so the relevance was the only criteria to evaluate the IR systems.

A novel method has been proposed to evaluate the quality of the retrieval systems by Elad Yom-Tov, Shai Fine, David Carmel, Adam Darlow in 2004. This method is based on machine learning technique and estimates the difficulty of the query. In this work the top results of the query should overlap with top results of each term of the query. This overlap shows the agreement between the query and the terms of that query. Difficult queries are those whose words do not overlap and are not in agreement. In this machine learning-based work the quality of estimation for systems and effectiveness of the learning method have been demonstrated. The top results of each subquery are gathered while the main query is being processed by the system. Actually, this research aimed to make predictions according to the overlap between the results of the main query and its sub-queries. This novel method can affect the performance of IR systems to determine the difficult queries in order to manage the function of the retrieving system(Yom-Tov et al., 2005).

Finding the information which is close to the desired query can be overwhelming because addressing the query to all databases in real world search engines is expensive. When a user submits a query, there may be millions of databases that the query can be addressed to, so if the user submits the query to all databases, then it would not be efficient enough. At the same time, broadcasting the queries by all users can overload the databases. So, this problem can be solved by forwarding the queries to the databases that are likely to have the related information for that query. The work of Danzig, Peter B, Ahn, Jongsuk, Noll John ,Obraczka and Katia which is called distributed indexing merges autonomous and heterogeneous retrieval systems to solve this issue. This system is not owned by a single agency, where millions of heterogeneous retrieval systems are put together in this architecture (Danzig et al., 1991).

The applications for the field of IR are improving day by day. To manage personal and social information new methods have been developed. Some research in this field are user tagging, filtering and recommendation and collaborative search(Sanderson & Croft, 2012b).

3. TERRIER

Furthermore, Terrier is an open-source platform to develop the IR process. Terrier is a modular platform that can index many test collections like TREC (The Text Retrieval Conference) data collection and web collections (Ounis et al., 2006). One of the crucial characteristics of this platform which makes it the best fit to be used in research and experiment in IR systems is its flexibility that can be used to index large data collections. In this work, TERRIER has been applied as test-bed, due to its high flexibility and being appropriate for large data collections.

3.1 INDEXING

Indexing in Terrier can be accomplished by performing tokenization on the data documents. Meanwhile, Terrier has its own tokenizer and stemmer for English corpora**;** it also has different tokenizer classes that can be used according to the language that is being processed. In addition, there are different stemmers available to use in Terrier. Some of them are Danish SnowBall Stemmer, German SnowBall Stemmer, Turkish SnowBall Stemmer which belong to SnowBall stemmer project. Stop-word removal and stemming are performed on the terms which were processed through the Term-Pipeline. As terrier contains modules to carry out indexing and retrieving, they can be deployed to form new applications. Terrier has the API to index corpus and this has been made possible by parsing the documents in different forms. Indexing task can be completed by using TREC-Collection property for different types of corpora. TERRIER encompasses the parser for the TREC-formatted corpora, TREC collections but at the same time DOCHDR-tagged documents, Corpora in the format of WARC, HTML, Microsoft Word, Excel, PowerPoint, PDF, Text documents and XML corpora formatted files (Ounis et al., 2006). At the indexing step the indexer has different properties including, the number of tokens the indexer is going to index and some properties to enable block-indexing that have their own default values. Indexing in Terrier comprises three types such as MapReduce, Two-Pass and Single-Pass indexing. Selecting one of these types can be according to the goal of the research and query expansion. Two-Pass indexing cannot be executed on a large number of documents while Single-Pass indexing actually cannot create direct indexing, but it can process the data faster. In this work, the indexing module of TERRIER is used to index the data collections which have been divided and prepared in advance.

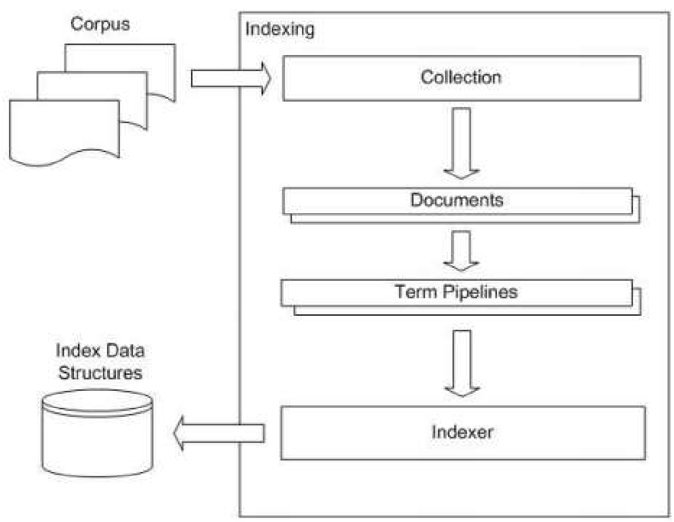

Figure 3.1: Indexing Architecture of Terrier (Ounis et al., 2006)

3.2 RETRIEVAL

Retrieval module of Terrier is performed after indexing the corpus documents. To make retrieving feasible, the topic files of TREC (tracks provided by The Text Retrieval Conference) are needed. The topic files are in two formats. The format which can be processed by Terrier is in the form of XML-like tags and the other format which is plain text. Moreover, according to the purpose of the research, a weighting model should be chosen to complete the retrieving task. Terrier lays out implementation of weighting models such as BM25, IFB2, TFIDF, DFIZ, DFR. Additionally, Terrier includes a function called "Query

Expansion" which adds some most illuminating terms extracted from the documents, which were the most relevant ones after the first time matching to the result-set that forms a new query and then the process is repeated to get the best results.

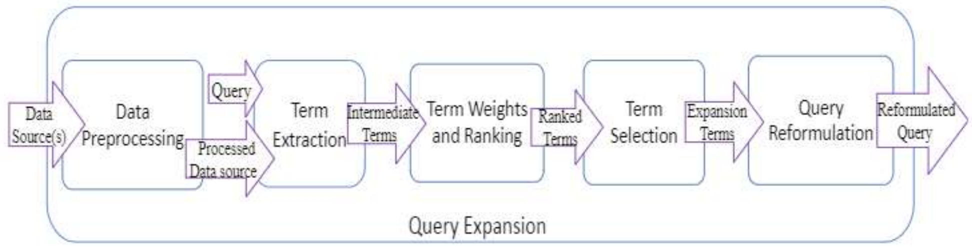

Figure 3.2: Query Expansion Process(Azad & Deepak, 2019)

The Query Expansion task consists of 5 steps. First the pre-processing phase
and term extraction are carried out. When term extraction is completed, then the term weighting algorithms are applied and then the terms are selected. Finally the query gets formulated again (Azad & Deepak, 2019).

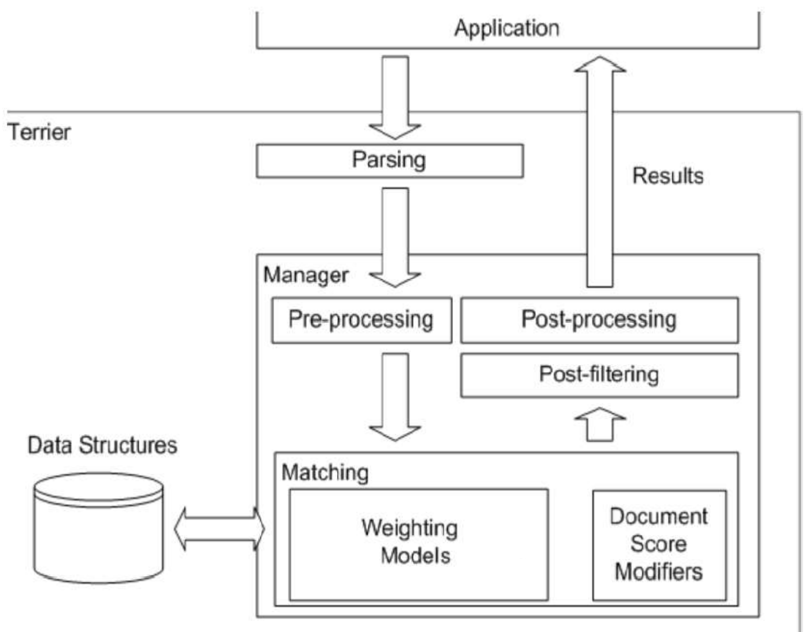

Figure 3.3: Retrieval architecture of Terrier(Ounis et al., 2006)

As seen in Figure 3.3, the retrieval process of TERRIER is demonstrated. In this procedure, the query is parsed and then handed down to the manager component. After the query being pre-processed, the weighting model and document score modifier is initialized. In post-processing and post-filtering steps the result-set can be modified and finally the result-set is addressed to the client (Ounis et al., 2006).

3.3 Evaluation

Terries contains a function which is used to evaluate the results after retrieving the related information. The relevance assessments are considered to evaluate the resulted documents which are going to be discussed in details.

3.4 Text Retrieval Conference

The Text Retrieval Conference (TREC) provided a framework to evaluate IR methodologies for large-scaled data. TREC, which started in 1992 afforded text documents and questions for test retrieval purpose of text data and also provided evaluation infrastructure for documents in other languages. The data provided by TREC consists of three types such as: Text Documents, Topics and Relevance Judgments.

3.4.1 TREC DOCUMENT COLLECTION

The TREC workshops motivate researchers to carry out retrieval on large text collections and also help to improve some retrieval methodologies on real world problems (Conference et al., 1999). Text collections that are available in TREC document collections comprise materials from the Wall Street Journal, Associated Press, U.S. Patents, Computer Selected Disks, the Federal Register, Department of Energy abstracts, San Jose Mercury News, Financial Times, the Congressional Record, the Foreign Broadcast Information Service and the Los Angeles Times (Voorhees & Harman, 2000). TREC conferences encompass routing and the ad hoc tasks. The purpose of the routing task is to explore how well the systems with standing queries can perform in searching new streaming documents. On the other hand, the ad hoc task utilizes a static set of text collections and examines the performance of the systems (Voorhees & Harman, 2000). Here in this work, the ad hoc task is performed with its related topics, collection documents and relevance assessments which are used to evaluate the retrieval performance. In this research experiments were conducted on Disks 4 and 5 of TREC test documents which contain The Financial Times, Federal Register, Congressional Records, Foreign Broadcast Information Service and The LA Times. The text data on Disks 4 and 5 is included in TREC-6, 7 and 8. TREC 6 contains 2 GB of data and 556,077 numbers of documents. TREC-7 and 8 include the files of Disk 4 and 5 except Congressional Records that makes 1.8 GB of data and 528,155 documents. The TREC text documents contain spelling errors and are close to the original text (Conference et al., 1999). The dataset used in Disk 4 and 5 are formed by mark-up language that are tagged using SGML to make parsing task easier. The documents which are formed by mark-up language, start with DOC tag and have inner DOCNO, PROFILE or DOCID, DATE, HEADLINE and TEXT tags (Voorhees & Harman, 2000).

```
<DOC>
<DOCNO>FT911-3</DOCNO>
<PROFILE>AN-BE0A7AAIFT</PROFILE>
<DATE>910514
</DATE>
<HEADLINE>
FT 14 MAY 91 / International Company News:  Contigas plans DM900m east German
project
</HEADLINE>
<BYLINE>
By DAVID GOODHART
</BYLINE>
<DATELINE>
BONN
</DATELINE>
<TEXT>
CONTIGAS, the German gas group 81 per cent owned by the utility Bayernwerk, said
yesterday that it intends to invest DM900m (Dollars 522m) in the next four years
to build a new gas distribution system in the east German state of Thuringia.  ...
</TEXT>
</DOC>
```

Figure 3.4: Financial Times Document extract(Conference et al., 1999)

3.4.2 TOPIC FILES

Topic files which are used as queries to carry out the experiments consist of 50 topics for each TREC-6, TREC-7 and TREC-8 data collections. As seen in Figure 3.5, there are topic files prepared for each TREC file. Topics which are utilized for TREC-6, TREC-7 and TREC-8 are numbered as 301-350, 351-400, 401-450. The queries are produced from topic statements with manual and automatic techniques. The topics are in the form of top tags and the inner tags such as, num, title, desc, and narr tags. Since, topics which are provided in TREC-3 are too long compared with queries, which are utilized in operational retrieval systems, the topics in TREC-4 are made even shorter. On the other hand, shorter topics in TREC-4 caused some issues in manual and automatic systems. So, the topics in TREC-5 changed to the format of TREC-3 topics. After the discussed change in TREC-5, then TREC-6, TEC-7 and TREC-8 were created with the same format. The titles in TREC-6, TREC-7 and TREC-8 can be used to investigate how length of queries affect the retrieval performance. The queries of mentioned TRECs are shorter queries, which contain no more than 3 words. The description field in TREC-6, TREC-7 and TREC-8 has all the words in topic and narrative. It also explains how a document can be relevant to the submitted query. As shown in Figure 3.5 the number of tokens in topic statements are demonstrated for each TREC file including the stop words (Conference et al., 1999).

	Min	Max	Mean		Min	Max	Mean
TREC-1 (51–100)	44	250	107.4	TREC-5 (251–300)	29	213	82.7
title	1	11	3.8	title	2	10	3.8
description	5	41	17.9	description	6	40	15.7
narrative	23	209	64.5	narrative	19	168	63.2
concepts	4	111	21.2				
TREC-2 (101–150)	54	231	130.8	TREC-6 (301–350)	47	156	88.4
title	2	9	4.9	title	1	5	2.7
description	6	41	18.7	description	5	62	20.4
narrative	27	165	78.8	narrative	17	142	65.3
concepts	3	88	28.5				
TREC-3 (151–200)	49	180	103.4	TREC-7 (351–400)	31	114	57.6
title	2	20	6.5	title	1	3	2.5
description	9	42	22.3	description	5	34	14.3
narrative	26	146	74.6	narrative	14	92	40.8
TREC-4 (201–250)	8	33	16.3	TREC-8 (401–450)	23	98	51.8
				title	1	4	2.5
				description	5	32	13.8
				narrative	14	75	35.5

Figure 3.5: Topic Length Statistics

3.4.3 Relevance Judgment Files and Evaluation

The relevance judgment files are necessary to complete the retrieval task and are used as a benchmark to make result-set comparable. These files contain columns with topic, iteration, document number and relevancy tags. Relevancy tag is a binary number. When the binary number is 1, it means the topic is relevant to the specified document and when it is 0, it means it is not relevant. Furthermore, the topic numbers available in these files are not sorted according to their degree of relevancy. Relevancy tag column is the only demonstration of relevancy of a topic number. To create the relevance judgement files, documents are gathered by participant systems. Then a topic is selected and first top 1000 retrieved documents are taken and merged to the pool of assessments. Furthermore, to refine the perfectness of the relevance judgment pool, it was suggested to run fewer number of judgments for topics with less relevant documents and conversely, run more judgment for topics with higher number of relevant documents (Conference et al., 1999). The relevance assessment is used to evaluate the effectiveness of the retrieval system. The TREC-Eval package which is written by Chris Buckley has been used to evaluate the performance of the system. The results generated by TREC-Eval are average scores over the set of topics. The precision and recall are taken into account to measure the effectiveness of weighting algorithms in the retrieval system. If the relevant documents are only retrieved (irrelevant documents are not retrieved) the precision reaches 1 and if all available relevant documents are retrieved the recall value will be 1. Performing experiments on TREC documents have goals like estimating the effectiveness of the retrieval systems and trying to increase the designed systems' performance (Conference et al., 1999).

4.DISTRIBUTED INFORMATION RETRIEVAL EXPERIMENTAL SET-UP

As discussed earlier, Terrier, as an open-source modular platform, can be utilized to be extensible for research purpose. Throughout this research, indexing and retrieving modules of Terrier are integrated with our java program to make experiments on TREC dataset. In this work, TREC-6, TREC-7 and TREC-8 are selected to make experiments on. TREC 6 encompasses all Files from Disks 4 and 5 while TREC 7 and 8 contain all the files of Disks 4 and 5 except CR (Congressional Records). Disks 4 and 5 of TREC 6, 7 and 8 were first split into 1, 2, 4, 8, 16, 32 partitions. Terrier desktop has been used in this work to generate a file which contains the path of the TREC collection files. The splitting task is done by utilizing this file. The content of the file has been iterated and has been split into aforementioned partitions. These specific numbers to create smaller sized collections are selected to make this experiment, but according to the architecture of the designed program, it is possible to create any desired number of sub-collections. Each generated divided partition includes 10 tests, which are created randomly. Above all, random means creating sub-collections with different combinations of files.

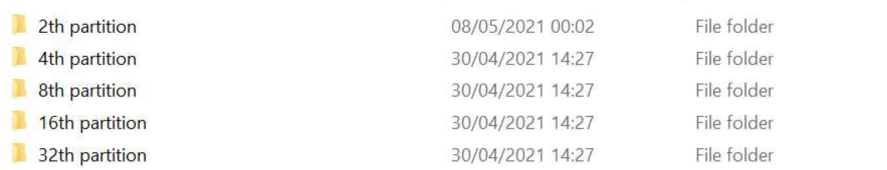

Figure 4.1: Files containing sub-collections

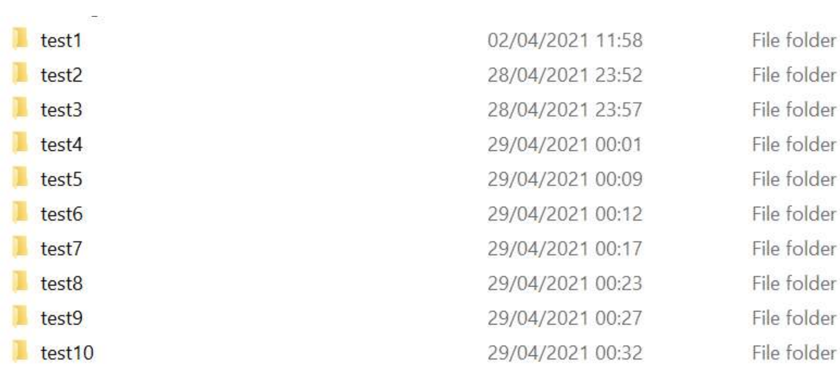

Figure 4.2: Test files generated forTREC-7

So, after the first task has been carried out, the generated smaller sized collections for every 10 test is ready to be indexed. As, the indexing phase has been discussed previously and some details were given about the Term-Pipeline, it is clear that the tokenizer goes through the text collection and sends each term to the Term-Pipeline to be processed. Consequently, the stop word removal and stemming are performed through this step. Both Document id and document score are accessible after indexing has been carried out. Once these structures have been created, The Inverted-Index and Direct index are created (Ounis et al., 2006). In other words, each smaller-sized collection goes through this process independently and each document metadata becomes accessible after the indexing step.

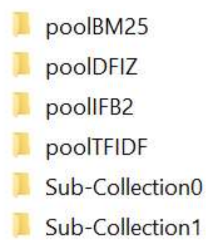

Figure 4.3: Test-3 of 2 Sub-Collections of TREC-7

In Figure 4.3, an example of test 3 of the TREC-7 data collection folder, which is divided to 2 sub-collections, is demonstrated. Here in this figure, 4 used weighting models' file structure is shown. Each of these sub-collection folders contain the indexing files and the divided smaller-sized collection which contains the index is shown in Figure 4.4:

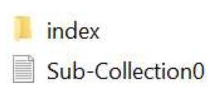

Figure 4.4: An example of a sub-collection with its index folder

Retrieval is the next step which is going to be discussed. Retrieving process needs the index of each sub-collection to be searched. Afterwards, Retrieval is carried out by processing the index of each divided collection. The queries used in this experiment are topic files prepared for TREC-6, TREC-7 and TREC-8 data collections. The topic files used for this task contain 50 queries which are prepared by TREC. As mentioned above, the topic files prepared for each TREC are passed to the search request for each TREC separately.

TREC6: Contains topics starting from 301 to 350.
TREC7: Contains topics starting from 351 to 400.
TREC8: Contains topics starting from 401 to 450.

After topics have been assigned, the weighting model should be specified. In this work BM25, TFIDF, IFB2 and DFIZ weighting models have been used.

4.1 TF-IDF WEIGHTING ALGORITHM

TFIDF is a weighting algorithm that is used to show the importance of a term in a document. Furthermore, this algorithm can be used in stop word filtering, and checks how relevant the term is to the specified text document. TF-IDF is an algorithm that is made up of two parts which are Term Frequency and Inverse Document Frequency (Al-Talib & Hassan, 2013). Each term in the corpus has a TF and IDF value. The formula of TF-IDF is as shown:

$$\textbf{TF-IDF(ti,dj)=TF(ti,dj)*log[N/DFti]} \qquad (1)$$

TF(ti,dj) is the number of occurance of term t_i in document d_j.

N is the total number of documents in the corpus.

DF_{ti} is the number of documents which contain the term t_i.

4.2 BM25 WEIGHTING ALGORITHM

The other weighting algorithm which is used to rank the documents related to the queries is BM25. This algorithm retrieves and ranks documents based on the terms of the query which are available in the documents. The BM25 class implements Okapi BM25 weighting model. The score function in this class calculates the weighting result by using the parameters as follows:

T_f which is the term frequency for a term in the document.

Doc-Length which is the document length in words.

Key-Frequency which is the term frequency in the query.

Then the weighting value is returned by the score function.

After TF-IDF and BM25 weighting models are performed, the ranking results are returned and sorted for each query. Each divided collection has its own sorted result-set for each weighting algorithm. Accordingly, all sub-collections' results should be merged to constitute the final result-set, but the results should be normalized to be sortable and combinable. Therefore, two normalization algorithms have been applied on the results. These techniques are Min-Max and Sum normalization methods.

4.3 IFB2 WEIGHTING ALGORITHM

This weighting model, which is known as the Inverse Term Frequency model for randomness, uses T_f (The term frequency in the document) and Doc-Length (the document's length) parameters to assign a score to each document. IFB2 weighting model which is defined in TERRIER is used in this experiment to retrieve the information related to each topic file.

4.4 DFIZ WEIGHTING ALGORITHM

The DFI term weighting model is related to the divergence from randomness (DFR) model. The DFR model assumes that the important terms of a document are the terms whose frequencies diverge from the frequency of a basic randomness model, on the other hand, DFI model assumes that the important terms are the terms whose frequencies diverge from the frequency suggested by the independence model (Dinçer et al., 2010). Alongside other weighting models, in this experiment the DFIZ weighting model with 2 normalization techniques is used to evaluate the results of the work, which are discussed in this research.

4.5 MIN-MAX NORMALIZATION ALGORITHM

When evaluating the results from different resources, the values should be comparable. Thus, for a more accurate evaluation the values should be normalized. Min-Max (Patro & sahu, 2015) is one of the decimal scaling techniques that gives a range between 0 and 1 . For this purpose, the minimum and maximum values get modified to zero and one. In this work the top-ranked document's score is mapped to 1 and the least-ranked one is mapped to 0. The values for Min-Max are calculated by the following formula:

$$\mathbf{S_{MinMax}(d|q) = S(d|q) - S_{Min}(q)/S_{max}(q) - S_{min}(q)} \qquad (2)$$

S(d|q) is the document score.

$S_{Min}(q)$ is the minimum score which was the result of retrieving documents related to the query.

$S_{max}(q)$ is the maximum score which was the result of retrieving documents related to the query.

4.6 SUM NORMALIZATION ALGORITHM

Sum is the other normalization method which transforms the sum of the values to be 1. In this work Sum method is used as the second technique to normalize the scores and the formula is as follows:

$$\mathbf{S_{sum}(d|q) = S(d|q) - S_{min}\,(q)/\textstyle\sum_i S(d|q) - S_{min}} \qquad (3)$$

$S(d|q)$ is the document score.

S_{min} (q) is the least ranked document score(minimum score) .

In this technique the least ranked document is set to 0.

When normalizing the scores, the result-set of each query is sent to the normalization functions to be normalized. Since they have to be in the form of an ordered list, each normalization technique sends back the normalized result-set. In the Min-Max normalization method the top-ranked document score will be set to 1, Thus all top scores of divided smaller sized collections, no matter what their scores were before normalization, have the same value which equals to 1 after normalization. Accordingly, the sub-collections have to be merged after normalization. With this intention, the scores for each sub-collection of data are gathered and sorted to create a single final list of scores. Hence, the list for each topic file is available as a file which comprises the first 1000 top-ranked scores. The normalized and merged result file contains the information of each file including the topic number, the query order number, document id, the normalized score and the weighting model applied.

```
351 Q1 FT931-10913 0 34.18633358291051 BM25
351 Q1 FT941-13429 1 33.027843392993674 BM25
351 Q1 FT932-6577 2 31.172274180129918 BM25
351 Q1 FT922-15099 3 29.918534906269244 BM25
351 Q1 FT921-6603 4 28.935110302131367 BM25
351 Q1 FT931-932 5 24.764833766135336 BM25
```

Figure 4.5: normalized result-set of TREC-7

In Figure 4.5, an example of TREC-7 result-set which has been normalized is demonstrated. This file consists of columns of topic number, query number, document number, the order number which belongs to the unnormalized file, normalized score and applied weighting model. The order number here in this file has no effect on the retrieval and evaluation process.

4.7 EVALUATION

On the whole, the retrieving, normalizing and merging operations have been applied after all sub-collections are indexed for TREC-6, TREC-7 and

TREC-8. After merging has been finished all the results are evaluated. The evaluation task is made possible by using the query relevance (qrel) files for TREC-6, TREC-7 and TREC- 8 data collections. The result-set which contains the retrieved results of all topic numbers is compared to the specified qrel file. Accordingly, the evaluation result is written into the Min-Max and Sum files for each partition. In fact, the evaluation results for each weighting algorithm and normalization method are available on different files.

Figure 4.6: Min-Max and Sum evaluation results files of BM25 weighting model of TREC-7

4.8 AVERAGE PRECISION AND RECALL

Average Precision is used to measure the performance of the retrieval task and is the combination of recall and precision values. In other words, Average Precision is the mean of precision scores when each relevant document is retrieved in the result-set of the search (Thom & Scholer, 2007). The formula is as follows:

$$\mathbf{MAP} = \sum_{q}^{Q} \mathbf{Avep(q)/Q} \qquad (4)$$

This formula demonstrates the value of precision, then mean of all Average precision scores are calculated. Thus, this formula represents the quality of the work in retrieving the results for a given query.

Recall is another metric to evaluate the performance of retrieval task. The relevant documents available for the given query which have been retrieved are taken into account to calculate the mean of precision (Thom & Scholer, 2007). To be specific, R precision is the precision at R^{th} position for a given query in the result-set of relevant documents.

4.9 EXPERIMENTAL ENVIRONMENT ARCHITECTURE

As mentioned before, 10 tests are created for each sub-collection and the evaluation results for each division is averaged. In brief, 1, 2, 4, 8, 16 and 32 partitions contain 10 tests that have been split randomly. Each smaller-sized collection contains its index where, each 4 weighting models are selected to be applied to each sub-collection. Each weighting model contains the unnormalized retrieved results for 50 queries. After the results are normalized by Min-Max and Sum normalization techniques, the normalized results are then merged into a single file of 1000 results. These 1000 top results are first generated for each query. After all, the result-set of each query is combined to make a file containing results of all topics of each TREC collection. The result file of all queries for each TREC is evaluated and averaged for 10 experiments. Consequently, the average value of Map and R-Prec are taken into account to determine the performance of the weighting models used in this experiment. In this experiment each weighting model is used with one normalization method. So, for each TREC file, there are 4 final result files created which are normalized. For instance, BM25 weighting model's retrieved result file is normalized with both SUM and Min-Max normalization techniques. As prementioned, 4 weighting models are used to retrieve the results for each TREC collection, the result files are each normalized with 2 normalization techniques separately.

MinMax-351-.res	23/01/2021 14:06	RES File	48 KB
MinMax-351-4001.res	23/01/2021 14:08	RES File	2,367 KB
MinMax-352-.res	23/01/2021 14:06	RES File	49 KB
MinMax-353-.res	23/01/2021 14:06	RES File	49 KB
MinMax-354-.res	23/01/2021 14:06	RES File	49 KB
MinMax-355-.res	23/01/2021 14:06	RES File	51 KB
MinMax-356-.res	23/01/2021 14:06	RES File	50 KB
MinMax-357-.res	23/01/2021 14:06	RES File	49 KB
MinMax-358-.res	23/01/2021 14:06	RES File	50 KB
MinMax-359-.res	23/01/2021 14:06	RES File	50 KB
MinMax-360-.res	23/01/2021 14:06	RES File	50 KB
MinMax-361-.res	23/01/2021 14:06	RES File	51 KB
MinMax-362-.res	23/01/2021 14:06	RES File	50 KB
MinMax-363-.res	23/01/2021 14:06	RES File	50 KB
MinMax-364-.res	23/01/2021 14:06	RES File	7 KB
MinMax-365-.res	23/01/2021 14:06	RES File	52 KB

Figure 4.7: Min-Max normalized result-set files for topics 351-365 and file containing all normalized results of the TREC-7

In Figure 4.7, an example of the result files of TREC-7 is demonstrated. These files start from 351 to 400, and are all integrated to a file shown in this figure. The file containing results of all topic is "Min-Max-351-400" for test 1 which all of them have been normalized by BM25 weighting model.

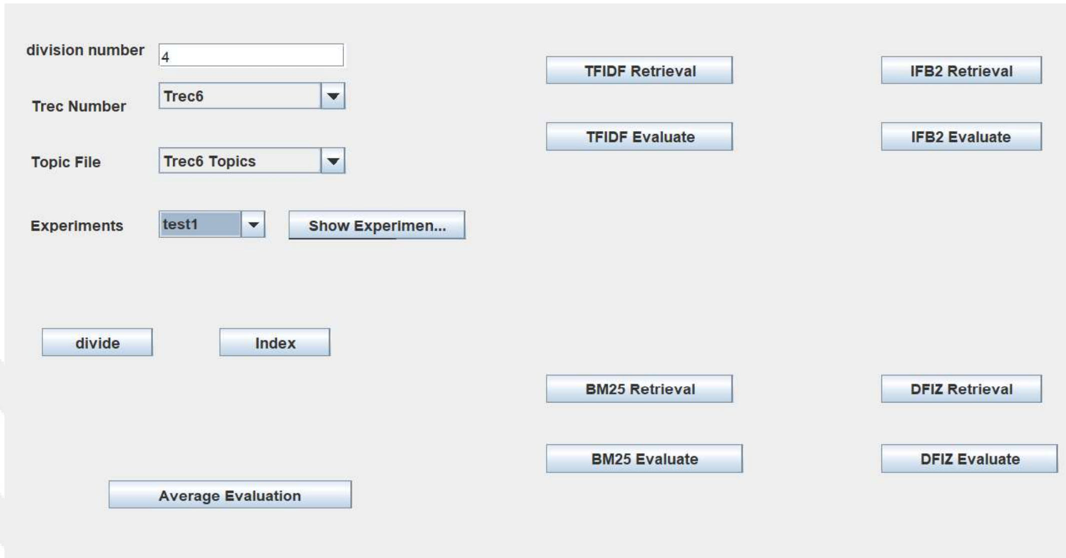

Figure 4.8: Designed Program GUI-TREC-6

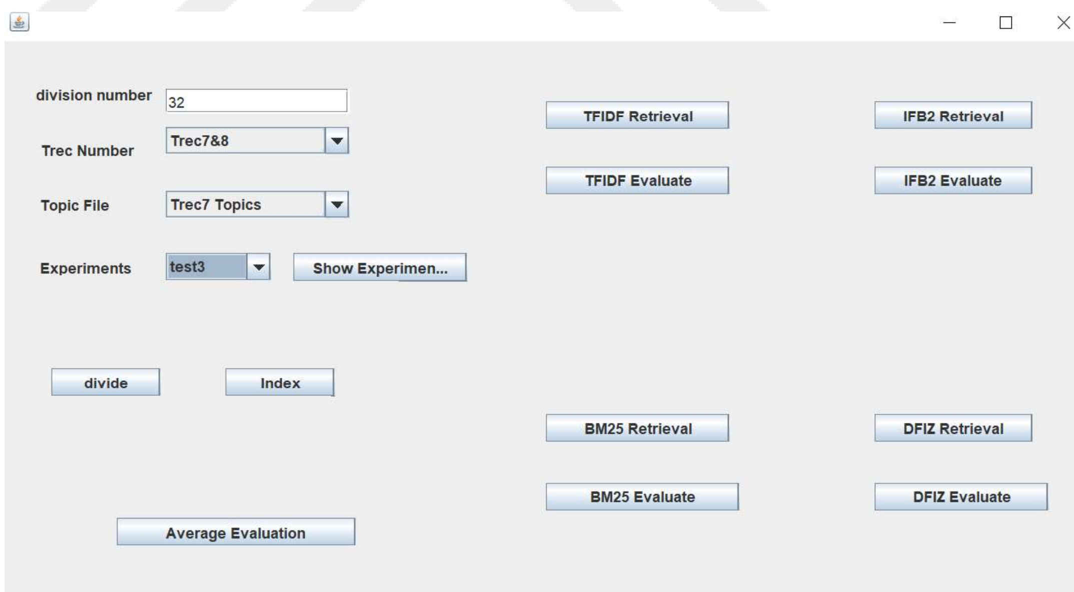

Figure 4.9: Designed Program GUI-TREC-7 and TREC-8

The designed program is shown in Figure 4.8 and 4.9. In the division number field, the number of sub-collections has been entered and it can be any desired

number. TREC number shows the 3 utilized TREC collections. Here in Figure 4.8 TREC-6 and in Figure 4.9 TREC-7 and TREC-8 have been selected. After the collection file has been used, the related topic file is shown and can be selected. In Figure 4.9 whereTREC-7 and TREC-8 are selected the topics should be selected accordingly. In this Figure topic files of TREC-7 are selected to be processed. In the Experiment field the tests are shown (if there are any created).
The number of tests created for each TREC is 10. Finally, after selecting the desired test number, the index button is to create the index files for the selected test and later, one of the weighting models can be selected to carry out the retrieval task. After retrieval is performed for the selected test, the evaluation can be carried out accordingly. When all tests have their own evaluations ready, the average evaluation is to create the average evaluation results for 10 tests for each divided smaller sized collection.

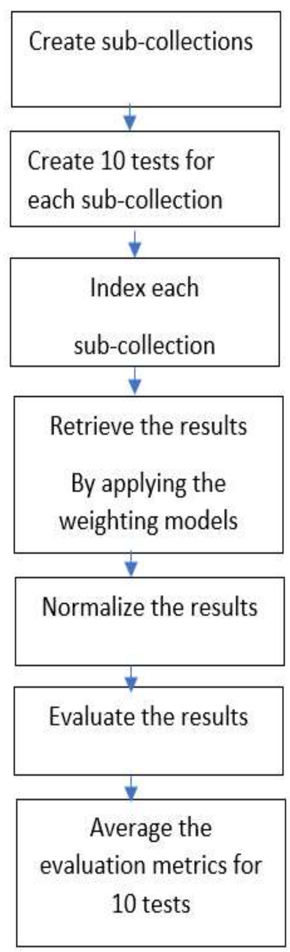

Figure 4.10: Tasks to create the automatic test environment

As seen in Figure 4.10 the tasks performed to create the test environment is demonstrated. While retrieving the results in 4th , BM25, TFIDF, DFIZ, IFB2 weighting methods and in 5th step SUM and Min-Max normalization methods have been chosen to apply. At final step, R-Prec and MAP values are taken into account and averaged for 10 tests.

5. EXPERIMENTAL SET-UP

In this research, the experiment was carried out on TREC-6, TREC-7 and TREC-8 text data collections. As mentioned before, TREC-6 consists of Congressional Record, Financial Register, Foreign Broadcast Information Service and Los Angeles Times collection which has 557 tokens. On the other hand, TREC-7 and TREC-8 include all mentioned collections above except Congressional Record collection which makes 512 tokens (Kocabaş et al., 2011). The experiment is accomplished by using TERRIER platform where direct and inverted indexes are used. Meanwhile, the stemming and stop word removal has to be carried out through the term pipeline. In this work, Porter's stemming algorithm and stop-words filtering have been used in indexing task. Each index includes a Statistics file which shows the number of documents and tokens. The token number and document number are extracted and put into the Statistics file by using Get-Collection-Statistics. The queries which have been submitted to obtain the result-set are selected from topic files which contain num, title and description. The titles are used as queries. TREC-6 topic file consists of titles which start from 301 to 350. TREC-7 and TREC-8 also have topics starting from 351 to 400 and 401 to 450. After each topic is sent, a retrieved file containing the results for each topic is created for the related sub-collection. In the evaluation step, the evaluation of the retrieved documents has been carried out according to the instance of TREC-Evaluation class. The evaluate function which accepts the result file, is utilized and the results have been written to the specified path by using the write-Evaluation-Result in TREC-Evaluation class. The evaluation step has been repeated for each used weighting algorithm TF-IDF, BM25, IFB2 and DFIZ separately.

6.EXPERIMENTAL ANALYSIS

As mentioned in previous sections, this experiment has been carried out by using TERRIER platform. Terrier contains implementation of some weighting algorithms that 4 of them are used in this work such as of TF-IDF, BM25, IFB2 and DFIZ. The text collection which has been divided to 1, 2, 4, 8, 16, 32 sub-collections have each been indexed and then retrieved thoroughly. Two types of files are generated by running this program, the files with unnormalized results for each sub-collection and normalized ones which are the combination of all smaller sized collections of the related division. The process which has been carried out after the retrieval to form the single evaluation result includes two normalization algorithms including Min-Max and Sum methods which have been explained with details in previous sections. In this experiment, the results retrieved for each sub-collection have been recorded on a file containing all first 1000 top ranked results for 50 topics. The evaluation results after evaluating the results-set of 50 topics have been written on the files for each used weighting algorithm separately. Each BM25, TF-IDF, IFB2 and DFIZ evaluation results' average have been recorded for 10 tests of each division. In this case, Average Map and R-Precision values for each divided collection of 10 tests have been recorded on files.

6.1 TREC-6 RESULTS EVALUATION

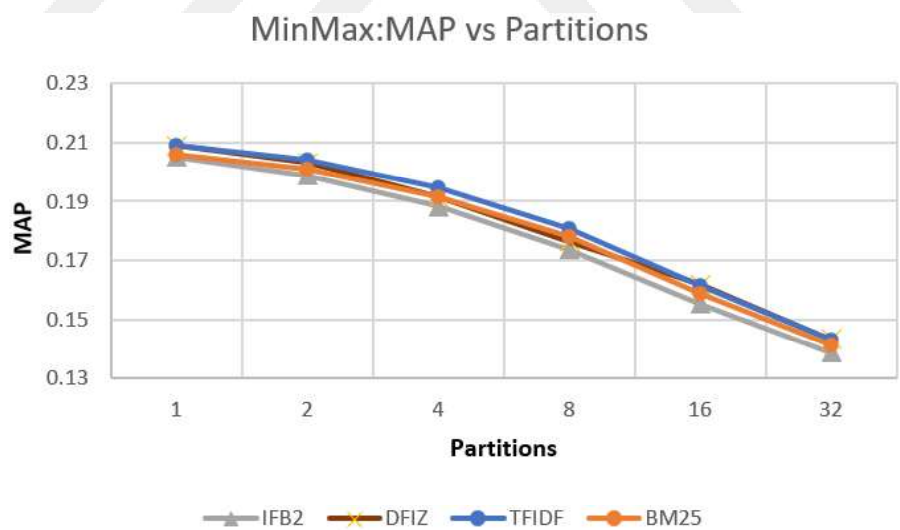

Figure 6.1: Min-Max: Disks 4 and 5 of TREC6, topics 301-350, MAP

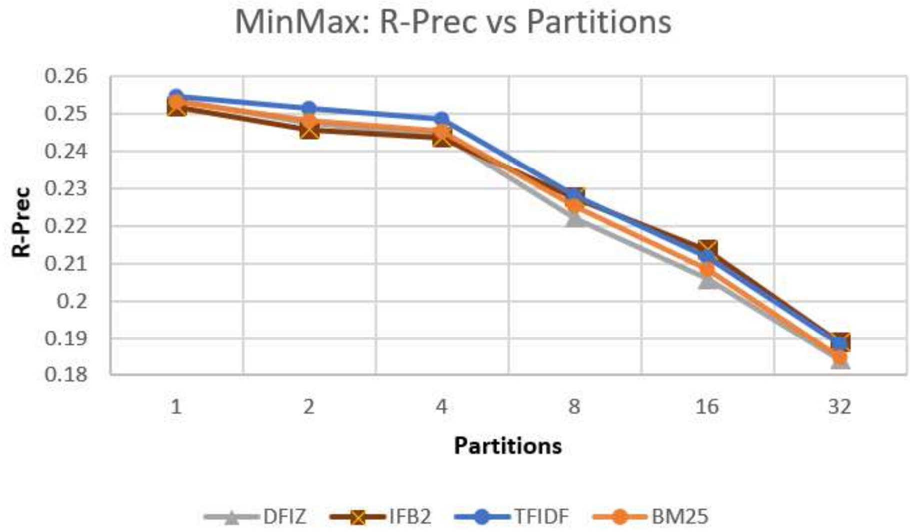

Figure 6.2: Min-Max: Disks 4 and 5 of TREC6,topics 301-350,R-prec

As seen in Figure 6.1, the Map values start to decrease when the number of partitions increase. There is not a huge difference in Map values between 4 weighting models in TREC-6, but TFIDF has the highest Map values and IFB2 has the lowest Map values when Min-Max normalization algorithm is applied.

As illustrated in Figure 6.2 the Recall value shows the graph decreases by the number of partitions increase from 2 to 32 for all 4 weighting models when Min-Max normalization method has been applied.

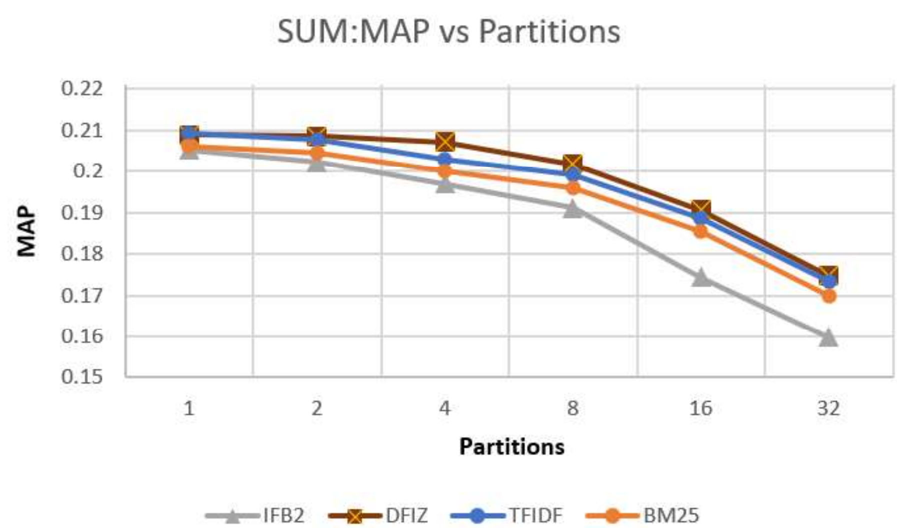

Figure 6.3: Sum: Disks 4 and 5 of TREC6,topics 301-350,MAP

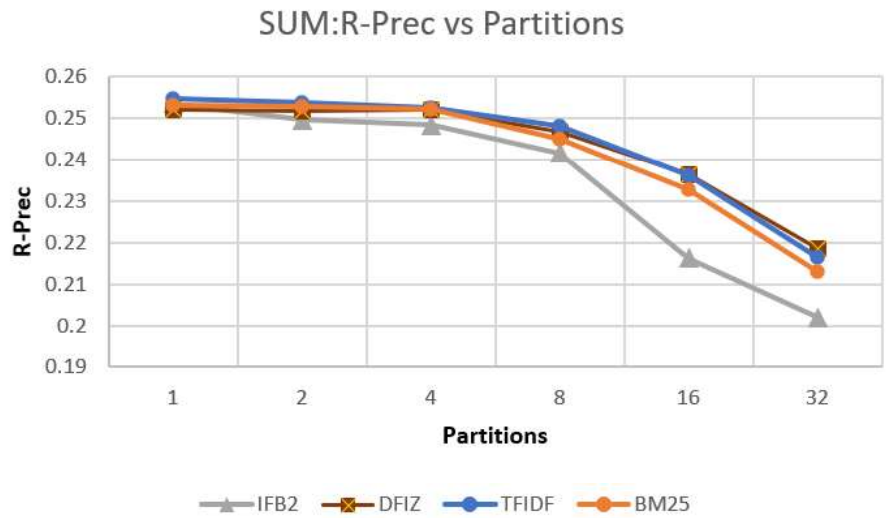

Figure 6.4: Sum: Disks 4 and 5 of TREC6, topics 301-350, R-Prec

As seen in Figure 6.3, Map values of all weighting models start to fall after the 4th partition. Furthermore, IFB2 shows a considerable fall at 8th, 16th and 32nd partitions in comparison with BM25, TFIDF and DFIZ weighting models. As shown in Figure 6.4 the R-Prec graph decreases when the number of partitions increase from 2 to 32. At the same time, values of the IFB2 weighting model show a significant fall after the 4th partition. Here, in TREC-6 when Sum normalization technique is applied, IFB2 weighting method shows the lowest performance where, DFIZ and TFIDF have the highest performance.

6.2 TREC-7 RESULTS EVALUATION

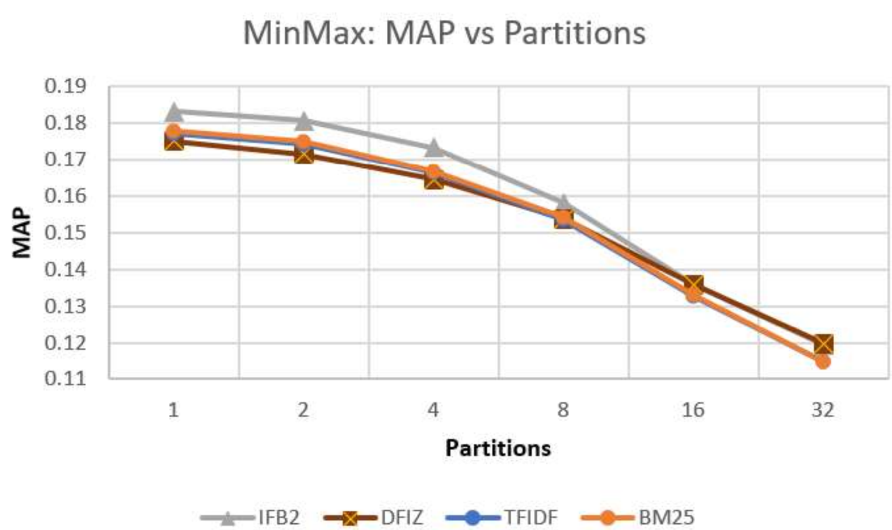

Figure 6.5: Min-Max: Disks 4 and 5 of TREC7, topics 351-400, MAP

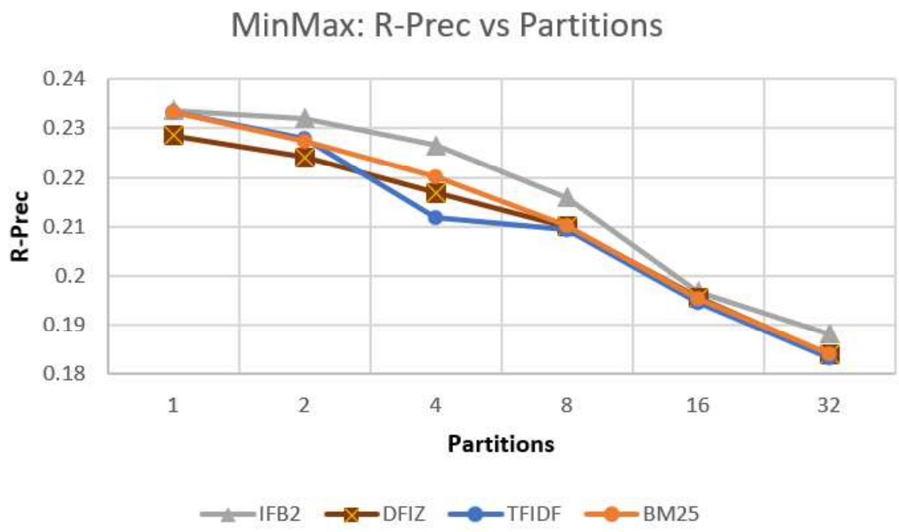

Figure 6.6: Min-Max Disks 4 and 5 of TREC7, topics 351-400, R-Prec

As seen in Figure 6.5, the Map value of IFB2 is higher than the other weighting models in partition 2 but after the 4th partition the Map value starts to fall significantly. After the 16th division the DFIZ weighting model shows a better performance. In Figure 6.6, the Recall value of IFB2 is considerably higher than Recall values of the other weighting models and they start to overlap after the 8th partition. By increasing the number of sub-collections, the Recall value starts to fall.

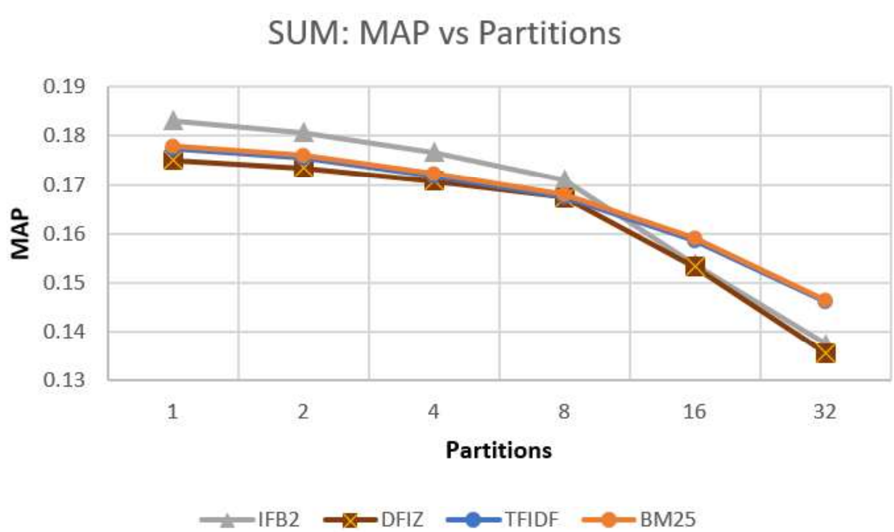

Figure 6.7: Sum: Disks 4 and 5 of TREC7, topics 351-400, MAP

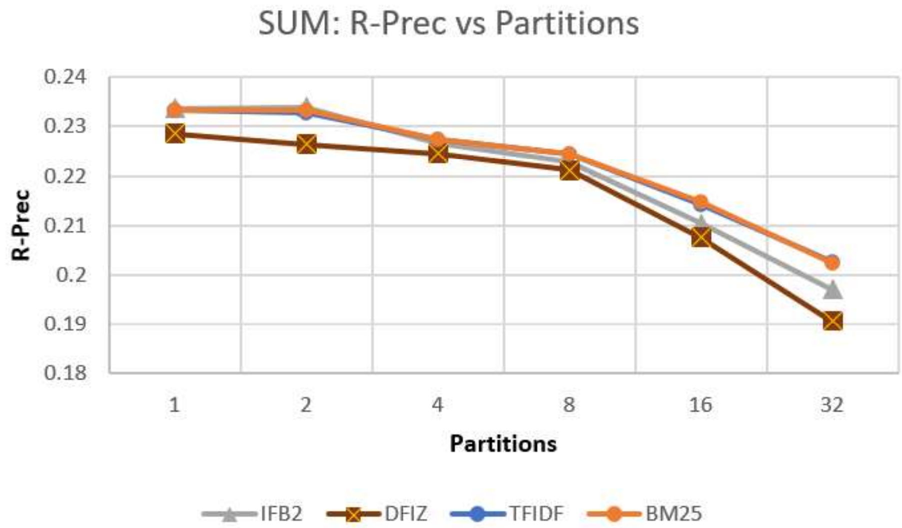

Figure 6.8: Sum Disks 4 and 5 of TREC7,topics 351-400,R-prec

As illustrated in Figure 6.7, the Map value of weighting models starts to fall in a significant way after the 8th partition and IFB2 starts show a high performance at 2nd partition, but the Map value of IFB2 start to fall below the other weighting models' values after the 8th partition. In Figure 6.8, Recall value starts to show a considerable fall in 16th and 32nd sub-collections.

6.3 TREC-8 RESULTS EVALUATION

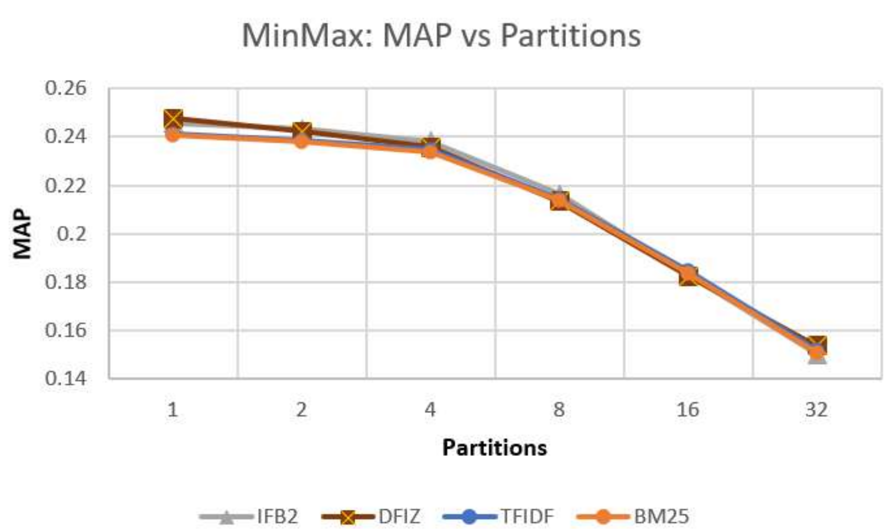

Figure 6.9: Min-Max: Disks 4 and 5 of TREC8, topics 401-450, MAP

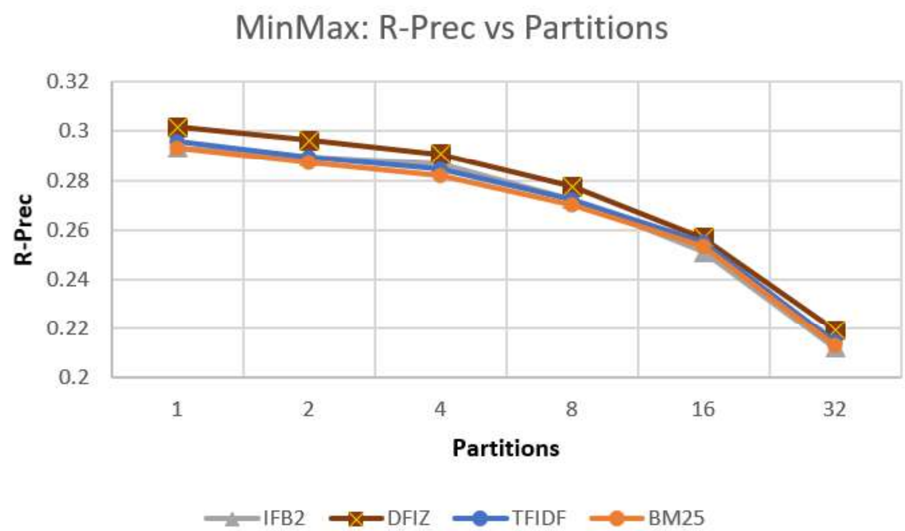

Figure 6.10: Min-Max: Disks 4 and 5 of TREC8, topics 401-450, R-Prec

In both Figures 6.9 and 6.10, for TREC-8 data collection the Map and Recall values fall at 8th partition. The Map and Recall values of 4 weighting models overlap and there is not much difference between the performance of all utilized weighting models.

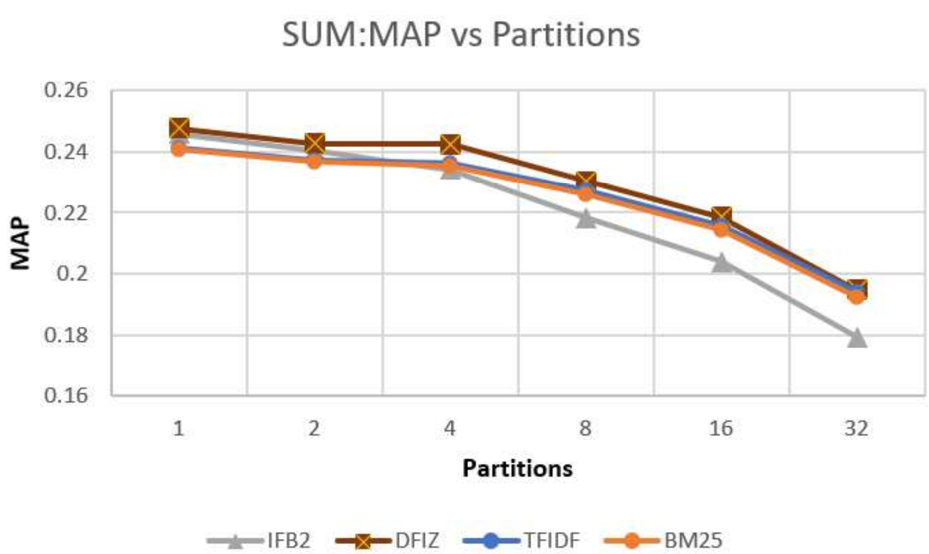

Figure 6.11: Sum Disks 4 and 5 of TREC8,topics 401-450,MAP

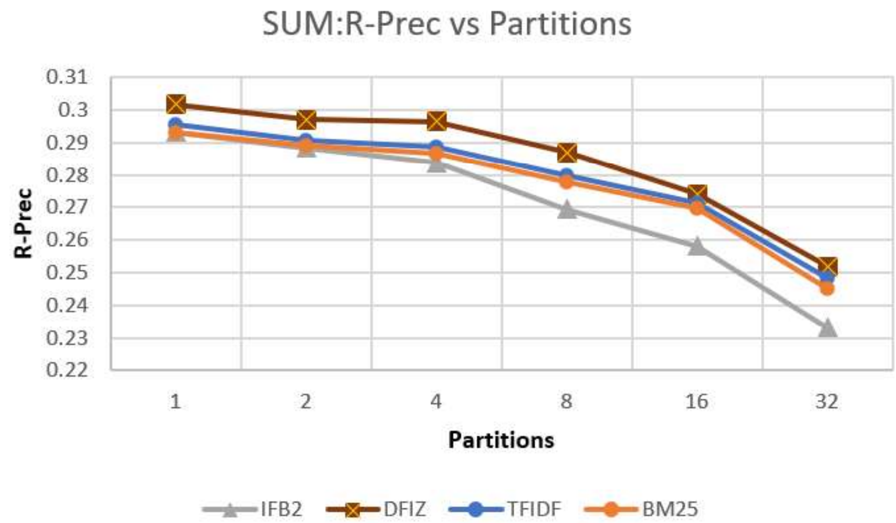

Figure 6.12: Sum Disks 4 and 5 of TREC8, topics 401-450, R-prec

As mentioned in previous graphics TREC-8 has the same behaviour, that the performance decreases by increasing the number of partitions. When Sum normalization technique is applied, utilized weighting models show the same behaviour and they all start to fall after the 4th partition. IFB2 weighting model's performance starts to fall after the 4th partition in comparison with other applied weighting models. The DFIZ weighting model has the highest MAP value, but it starts to overlap with TF-IDF and BM25 weighting models at the 16th partition. IFB2 shows the lowest performance when compared with 3 other weighting models.

7. FUTURE WORK

In this work, a distributed information retrieval experiment has been carried out and an environment to test and evaluate the results has been implemented. For this purpose, dividing the text data source, indexing, querying and retrieval are the major steps that have to be carried out. As the collection source has been divided to sub-collections, all mentioned phases has been performed separately on each part of the divided part.

Indexing is the major task in the process of information retrieval, and due to the large amount of data collections, skipping this phase makes the retrieval process impossible. Indexing consists of phases such as tokenization, processing the terms and storing the indexed files.

Before retrieving the related information, the index structure is taken into account to find the most relevant information. At the same time, for a better and a faster search the resources can be selected to make sure which document is more relevant to the information. Federated text search includes this major task. Resource selection can be performed by static information of resources that have been gathered in an offline manner. One method that has been proposed revolves around calculating the similarities of the current query with past queries that combines two resource selection algorithms (Cetintas & Si, 2011).

Retrieving of information involves applying the appropriate weighting method. The weighting algorithms propose an ordering of documents in accordance with the terms of the query which has been addressed to the data collection by the user (Liu et al., 2005).

The information retrieved from different resources are merged to form a final single result-set and is addressed to the user.

In this experiment the resource selection task has been skipped and the topics inside the topic files are addressed to the resources and all documents have been explored. So, this experiment can be carried out on the same TREC files by applying one of the resource selection techniques which have been proposed in other research works.

8. CONCLUSION

All in all, people are trying to access more information around the world and information retrieval systems are the core systems which make this task feasible. On the other hand, stored data has been spread around the world and reaching this information needs a technique which can retrieve data in a distributed manner. Here is when distributed information retrieval systems come to help. This system is a multi-database system in which the information can be retrieved separately from different smaller sized data collections on different systems. The TREC corpora which has been used in this experiment made it possible to access the large number of resources (Callan, 2005).

Performing the whole process of extracting information related to the query consists of indexing, retrieving and evaluation of the results. The tasks following the indexing task involve processing the query on each chunk of data to assign a score to each document according to the similarity between the terms of the document and the query (de Kretser et al., 1998).

The retrieval task which tries to estimate the similarities is based on weighting algorithms which have been used in this experiment to rank the documents. In addition, TERRIER is a platform which provides rapid development. This platform has been used as the search engine due to its high flexibility and efficiency (Ounis et al., 2007). The results-set of each technique has been used and had to be normalized with Sum and Min-Max normalization methods. Finally, generated merged result-sets of each weighting and normalization technique have been evaluated. Creating a GUI to make the retrieval task automatically as well as, performance examination and study of TF-IDF, BM25, IFB2 and DFIZ weighting models have been the objective of this work. Finally, generated merged result-sets of each weighting and normalization technique have been studied. All in all, when Sum normalization method has been applied the Map values show a better performance. Although, IFB2 and DFIZ have higher values in some cases, but TFIDF and BM25 weighting models are still more robust when number of partitions are increasing.

REFERENCES

Al-Talib, G., & Hassan, H. (2013). A Study on Analysis of SMS Classification Using TF-IDF Weighting. *International Journal of Computer Networks and Communications Security*, *1*(5), 189–194. http://www.ijcncs.org/published/volume1/issue5/p3_1-5.pdf

Azad, H. K., & Deepak, A. (2019). Query expansion techniques for information retrieval: A survey. *Information Processing and Management*, *56*(5), 1698–1735. https://doi.org/10.1016/j.ipm.2019.05.009

Callan, J. (2005). Distributed Information Retrieval. *Advances in Information Retrieval*, 127–150. https://doi.org/10.1007/0-306-47019-5_5

Cetintas, S., & Si, L. (2011). *Using Past Queries for Resource Selection in Distributed Information Retrieval*. 11–12.

Clough, P., & Sanderson, M. (2013). Evaluating the performance of information retrieval systems using test collections. *Information Research*, *18*(2).

Conference, T. R., Technology, I., Advanced, D., Jones, S., & Trec, T. (1999). *Ellen M . Voorhees , Donna Harman National Institute of Standards and Technology Gaithersburg , MD 20899 1 Introduction 2 The Ad Hoc Retrieval Task*.

Cool, C., & Belkin, N. J. (2018). Interactive information retrieval: history and background. *Interactive Information Seeking, Behaviour and Retrieval*, 1–14. https://doi.org/10.29085/9781856049740.003

Danzig, P. B., Ahn, J., Noll, J., & Obraczka, K. (1991). Distributed indexing: A scalable mechanism for distributed information retrieval. *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1991*, *August 2014*, 220–229. https://doi.org/10.1145/122860.122883

Das, S., Shuster, K., Wu, C., & Levit, I. (2005). Mobile agents for distributed and heterogeneous information retrieval. *Information Retrieval*, *8*(3), 383–416. https://doi.org/10.1007/s10791-005-6992-6 de Kretser, O., Moffat, A., Shimmin, T., & Zobel, J. (1998). Methodologies for distributed information retrieval. *Proceedings - International Conference on Distributed Computing Systems*, 66–73. https://doi.org/10.1109/icdcs.1998.679488

Dinçer, B. T., Kocabas, I., & Karaoğlan, B. (2010). IRRA at TREC 2010: Index term weighting by divergence from independence model. *NIST Special Publication*, 1–4.

Fouad, K. M., Khalifa, A. R., Nagdy, N. M., & Harb, H. M. (2012). Web-based Semantic and Personalized Information Retrieval. *International Journal of Computer Science*, *9*(3), 266–276.

Géry, M., & Largeron, C. (2012). BM25t: A BM25 extension for focused information retrieval. *Knowledge and Information Systems*, *32*(1), 217–241. https://doi.org/10.1007/s10115-011-0426-0

Jansen, B. J., & Mullen, T. (2008). Sponsored search: an overview of the concept, history, and technology. *International Journal of Electronic Business*, *6*(2), 114. https://doi.org/10.1504/ijeb.2008.018068

Kocabaş, I., Dinçer, B. T., & Karaoğlan, B. (2011). Investigation of Luhn's claim on information retrieval. *Turkish Journal of Electrical Engineering and Computer Sciences*, *19*(6), 993–1004. https://doi.org/10.3906/elk-1003-448

Liu, G., Yu, Y., & Zhu, X. (2005). A learning-based term-weighting approach for information retrieval. *Proceedings of the National Conference on Artificial Intelligence*, *3*(1), 1418–1423.

Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., & Ioma, C. (2006). Terrier: A High Performance and Scalable Information Retrieval Platform. *Proceedings of the Open Source Information Retrieval Workshop*, 18–25. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Terrier+:+A+High+Performance+and+Scalable+Information+Retrieval+Platform#0

Ounis, I., Lioma, C., Macdonald, C., & Plachouras, V. (2007). Research Directions in Terrier: a Search Engine\nfor Advanced Retrieval on the Web. *Upgrade*, *VIII*(1), 49–56.

Patro, S. G. K., & sahu, K. K. (2015). Normalization: A Preprocessing Stage. *Iarjset*, 20–22. https://doi.org/10.17148/iarjset.2015.2305

Pouamoun, A. N. (2021). *Multi-agent-based hybrid peer-to-peer system for distributed information retrieval*. 1–15. https://doi.org/10.1177/01655515211010392

Sanderson, M., & Croft, W. B. (2012a). The history of information retrieval research. *Proceedings of the IEEE*, *100*(SPL CONTENT), 1444–1451. https://doi.org/10.1109/JPROC.2012.2189916

Sanderson, M., & Croft, W. B. (2012b). The history of information retrieval research. *Proceedings of the IEEE*, *100*(SPL CONTENT), 1444–1451. https://doi.org/10.1109/JPROC.2012.2189916

Saracevic, T. (1995). Evaluation of evaluation in information retrieval. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, 137–146. https://doi.org/10.1145/215206.215351

Seymour, T., Frantsvog, D., & Kumar, S. (2011). History Of Search Engines. *International Journal of Management & Information Systems (IJMIS)*, *15*(4), 47. https://doi.org/10.19030/ijmis.v15i4.5799

Strohman, T., Metzler, D., Turtle, H., & Croft, W. (2005). Indri: A language model-based search engine for complex queries (extended version). *CIIR Technical Report*. http://ciir.cs.umass.edu/pubfiles/ir-407.pdf

Tarihal, S., Swami, R., Purandare, A., & Adike, K. (2013). Distributed Information Retrieval Using Mobile. *Computer Society of India*, *January 2012*, 182–185.

Thom, J. A., & Scholer, F. (2007). A comparison of evaluation measures given how users perform on search tasks. *ADCS 2007 - Proceedings of the Twelfth Australasian Document Computing Symposium*, *December*, 100–103.

Voorhees, E. M., & Harman, D. (2000). Overview of the Sixth Text REtrieval Conference (TREC-6). *Information Processing and Management*, *36*(1), 3–35. https://doi.org/10.1016/S0306-4573(99)00043-6

WOSP 2020 Proceedings of the 8th International Workshop on Mining Scientific Publications Wuhan , China. (2020).

Xu, J., & Callan, J. (1998). Effective retrieval with distributed collections. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, 112–120. https://doi.org/10.1145/290941.290974

Yom-Tov, E., Fine, S., Carmel, D., & Darlow, A. (2005). *Learning to estimate query difficulty*. *Sigir*, 512. https://doi.org/10.1145/1076034.1076121

Zhu, X., & Gauch, S. (2000). Incorporating quality metrics in centralized/distributed information retrieval on the World Wide Web. *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, *February 2000*, 288–295. https://doi.org/10.1145/345508.345602

ACKNOWLEDGEMENTS

I would like to express my special appreciation and thanks to my advisor Asst. Prof. İlker Kocabaş , you have been an excellent mentor for me. I would like to thank you for encouraging me while doing my research. I also must express my very profound gratitude to my family and friends for providing me with endless support and encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

30.07.2021

SHANAY ZARGARI AFSHAR

RESUME

Academic Studies:

- Master's Degree in Information Technologies, International Computer Institute, Ege University, İzmir-Turkey (January 2018-July 2021);
- Bachelor's Degree in Computer Engineering, , URMIA University, 2008-2012

Professional Experiences:

- (Full time basis), January 2013 – March 2014 ➢ Position :Asp.net, Database Designer and C# Programmer
- Coding Instructor, April 2012 – October 2012
- Front-End Developer, October 2011 – December 2011,HTML5,CSS

Skills and Abilities:

- Application Programmer (Web/Windows)
- DBMS Programmer
- IOT
- Python
- Java
- Asp.Net
- Programming skills in HTML 5, CSS 3, JavaScript
- Operating Systems (Linux, Windows Server & Windows)

Languages:

- English (fluent);
- Turkish (fluent).
- Farsi (Native)