DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

AUTOMATIC ASSIGNMENT OF MESH KEYWORDS FOR ARTICLES USING INFORMATION RETRIEVAL SYSTEM FOR PUBMED

by

Fatih DİLMAÇ

January, 2020

İZMİR

AUTOMATIC ASSIGNMENT OF MESH KEYWORDS FOR ARTICLES USING INFORMATION RETRIEVAL SYSTEM FOR PUBMED

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Computer Engineering, Computer Engineering Program

by

Fatih DİLMAÇ

January, 2020

İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"AUTOMATIC ASSIGNMENT OF MESH KEYWORDS FOR ARTICLES USING INFORMATION RETRIEVAL SYSTEM FOR PUBMED"** completed by **FATİH DİLMAÇ** under supervision of **ASSOC. PROF. DR. ADİL ALPKOÇAK** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Adil ALPKOÇAK

Supervisor

Asst. Prof. Dr. Kaan Kurtel

(Jury Member)

Assoc Prof. Dr. Şenol (ITK)

(Jury Member)

Prof.Dr. Kadriye ERTEKİN

Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor, Assoc. Prof. Dr. Adil ALPKOÇAK, for his supervision and invaluable suggestions through this study. His leadership helped me during the research and the writing of this thesis.

I would like to thank my best friends Bülent KOTLUK and Murat EMEÇ for being always with me during my undergraduate and graduate studies.

I would also like to express my gratitude to my parents, brothers and sister who have always believed in me throughout my life, and who have always supported me financially and spiritually while preparing this thesis.

Finally, I would like to thank my dear wife Leyla, who was with me during my undergraduate and graduate level, without her continuous patience I wouldn't have been able to do it. Thank you for being there.

Fatih DİLMAÇ

AUTOMATIC ASSIGNMENT OF MESH KEYWORDS FOR ARTICLES USING INFORMATION RETRIEVAL SYSTEM FOR PUBMED

ABSTRACT

Assigning keywords to research articles is a very important process. These keywords should describe the article properly. Performing this process manually is difficult and may cause an improper description of the article. Therefore, in this thesis, we designed and developed an automatic keyword suggestion system for research articles. In the application we developed two different corpus-based methods by utilizing information retrieval systems using Medline dataset in PubMed. First, proposed keyword suggestion system accepts abstract of the research article as a query to information retrieval system. Next, the information retrieval system returns a list of articles to the given query in ranked order of similarity. Then, we selected a set of documents from this list using two different methods: *k*-NN and *t*-NN representing the first *k* documents and documents whose similarity is greater than threshold value of *t*, respectively. To evaluate our proposed systems, we conducted a set of experiments using randomly chosen a thousand of articles, and provide a comparison of our system results with authors' keywords. The results we obtained showed that our system suggest keywords more than 42% match in terms of F-measure.

Keywords: Keyword suggestion, text mining, PubMed, Medline dataset, information retrieval

PUBMED BİLGİ GERİ GETİRİM SİSTEMİNİ KULLANARAK MAKALELERE OTOMATİK ANAHTAR KELİME ATAMA

ÖZ

Araştırma makalelerine anahtar kelimeler verilmesi çok önemli bir süreçtir. Bu anahtar kelimeler makaleyi doğru bir şekilde tanımlamalıdır. Bu işlemi el ile yapmak zordur ve makalenin yanlış tanımlanmasına neden olabilir. Bu nedenle, bu tezde araştırma makaleleri için otomatik anahtar kelime öneri sistemi tasarladık ve geliştirdik. Geliştirdiğimiz uygulamada, PubMed sistemindeki Medline veri setinin bilgi getirim sistemleri kullanımına dayalı iki farklı koleksiyon tabanlı yöntem kullandık. İlk olarak, önerilen anahtar kelime öneri sistemi, araştırma makalesinin özetini, bilgi getirim sistemi için bir sorgu olarak kabul eder. Daha sonra, bilgi getirim sistemi verilen sorgunun sıralı benzerlik oranına göre bir makale listesi döndürür. Daha sonra, iki farklı yöntem kullanarak bu listeden bir makale kümesi seçtik: *k*-NN ve *t*-NN. Bu yöntemler sırası ile ilk *k* tane makale ve benzerlik oranı *t*'nin eşik değerinden büyük olan makaleleri temsil etmektedirler. Önerdiğimiz sistemi değerlendirmek için, rastgele seçilmiş bin tane makaleyi kullanarak bir dizi denemeler gerçekleştirdik ve sistem sonuçlarını yazarların anahtar kelimeleriyle karşılaştırılmasını sağladık. Elde ettiğimiz sonuçlar, sistemimizin F-ölçütü bakımından %42'den fazla eşleşen anahtar kelimeler önerdiğini bize göstermiştir.

Anahtar Kelimeler: Anahtar kelime önerme, metin madenciliği, PubMed, Medline veri seti, bilgi geri alma

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

1.1 General

In recent years, technology is developing very quickly. As we all know, these technological developments touch every aspect of human life. Hundreds of technological developments such as phones, computers, autonomous vehicles, aircraft systems have become a part of human life. As a result, people's lifestyles have changed. With these developments, in recent years, the Internet has become the most widely used communication and data transfer instrument. The most widely used technology in daily life is undoubtedly the Internet. Even inter-continent communication is so fast and short-lived that even the human imagination is forced. For example, before Internet, a message sent by a long-distance letter reaches the destination months later, while a message sent across the continent reaches its destination in less than a second.

Thousands of applications using the Internet infrastructure that have enabled such a dizzying speed have emerged. The most popular ones are Facebook, Twitter, YouTube, Instagram and WhatsApp. Today, almost 50% of the world's total population uses this application in their daily lives (Kemp, 2019). Human beings use these tools to share videos and pictures, or to chat with each other. This produces millions of unstructured data every millisecond. Researchers want to access the information they want accurately and quickly in this huge unstructured data. In particular, when researchers do research on a subject, they would like to examine the previous studies. The most commonly used method for this is undoubtedly the Internet. If we have done a good job of assigning keywords to the research articles previously written on the subject and these key words describe the research well, it is possible to access these studies very quickly thanks to the search engines. However, assigning keywords is an important task, so the words should be chosen very well. Performing this process manually is costly and may cause improper assignments.

1.2 Problem Definition

Everyday hundreds or even thousands of research articles are published. Keywords are available in all of these articles. The keywords of a research article are generally a few words or word phrases that best suit the context of the article. We can get an idea about the article by just looking at the keywords. Keywords enable us to get semantic information for several text mining applications; for example, text classification, clustering, investigation and topic modeling (Zhang, Xu, Tang, & Li, 2006). Many studies have been done for keyword extraction. It is possible to divide them into two parts. Some take single article and deal with the frequencies of the words in the article, while the others are based on finding similarities from a collection. For example, Naive Bayes is keyword extraction using decision trees and some Term Frequency- Inverse Document Frequency (TF-IDF) methods. These approaches have accomplished satisfactory results and are widely used. However, keyword assignment still has some difficulties, especially the keyword assignment process. Because these keywords are sometimes assigned by authors, they can sometimes cause problems. For example, depending on the author's current status or customized according to the subject of the journal. Also, the authors may have missed the important words, or a week later the author will assign the same article different keywords, perhaps. In our thesis, we aimed to solve these problems and assign keywords to articles automatically.

1.3 Motivation

Our motivation in this study is that although there are many studies on keyword extraction, very few studies have been done on automatic keyword suggestion system. In addition, the lack of any studies on the Medline data set is one of the most important reason of motivation that pushes us to work on the domain. One of the reasons why we work on the keyword suggestion system is to remove the subjective keywords manually assigned by the authors to the research articles and to automate them.

1.4 Contribution of the Thesis

In this thesis, we proposed an automatic keyword suggestion system for scholarly articles. With our system, which we have designed and developed using information retrieval techniques, researchers can easily use to automatically assign keywords to their articles. Authors are free whether to accept the suggested keywords by our system or not, the system aims to prevent researchers from assigning irrelevant words as keywords. Moreover, one of our most important contributions in this thesis is the development of this application by combining information retrieval systems and text mining techniques for the first time on scholar articles. To the best of our knowledge, this is the first automated keyword suggestion system for scholarly articles written in the biomedical field.

1.5 Thesis Organization

This thesis consists of five main chapters. In chapter one, the introduction of the thesis is given. In this chapter, we also discussed the motivation, goal of the thesis, problem definition and our contribution of the thesis. The rest of chapters are prepared as follows: In the chapter two we present a literature review about keyword extraction using text mining techniques. We also discussed in detail the areas where keyword suggestion and extraction are widely used. In the chapter three, we explained in detail the technologies we use, the tools and every phase of the system we developed. We also explained in detail how we gained the Medline data set, which we use as a corpus in our system. In the fourth chapter, we explained the experiments and the results of the experiments in detail. In the last chapter, the conclusion of the thesis is given sharing the overall results and the contribution of them. Then, the future works are discussed.

CHAPTER TWO
LITERATURE REVIEW AND BASIC DEFINITIONS

In this chapter, we explained the literature review on the topics such as keyword suggestion, extraction, assignment and the basic definitions, concepts and terminology used in these fields. We also explained in detail the areas where keyword suggestion and extraction are widely used and why they are used.

2.1 Literature Review

There are many studies on Automatic Keyword Extraction and assignment to documents. Especially statistical-based approaches are widely used. A few of these studies are shown below.

In 2008, Zhang et al. collect Chinese textual data from database of "Information Center for Social Sciences of RUC". They arbitrarily selected 600 research paper in domain of finances from database. They separated the data into 10 data sets and used 10-fold cross-validation for Conditional Random Fields (CRF). Each document contains title, abstract, full-text, keywords. After collecting data, the most important thing is data preprocessing and POS tagging what they have done. They used Set Tag tools of NLP library for POS tagging which is do automatically processing the labels. The results obtained at the end of the study were calculated using precision, recall and f-measure which are the general assessment metrics (Zhang, et al., 2008) . As a result, their studies results indicate that the CRF approach better than other machine learning approaches for instance SVM and MLR model etc. In the process of keyword extraction.

In 2010, Rose et al. tried to show how efficient Rapid Automatic Keyword Extraction algorithm (RAKE) is. RAKE is a type of machine learning which is unsupervised learning and domain independent for extracting keywords from single documents. This algorithm uses text mining and NLP techniques. For this reason, they did not need to use data for the training part. Moreover, they also claimed that RAKE

is faster than other algorithms by comparing RAKE with existing keyword extraction algorithms such as TextRank, unsupervised learning techniques. To get this study done they got dataset from technical abstracts, they showed that RAKE method is more effectual than TextRank method. Consequently, they have shown that RAKE, acquires similar recall and higher precision in comparison to existing techniques. Finally, RAKE's efficiency and simplicity make able its use in several applications where keywords can be leveraged (Rose, Engel, & Cramer, 2010).

Rak et al. in 2007 they proposed a novel system called ACRI (Associative Classifier with Reoccurring items) which automates the classification of Medline articles to Mesh Keywords, in other words, assigning Mesh Keywords to article references, their proposed system was modified to take in multilabel classification. They proposed five dissimilar classification configurations in conjunction with different methods of determining classification quality. They have verified their system on the OHSUMED corpus, which consists of a comprehensive set of almost 350,000 documents. Their results show the advantage of methods based on re-occurrence of words in an article over nonrecurrent-based associative classification (Rak, Kurgan, & Reformat, 2007). The calculated value for macro F1 was 46% which indicates the high quality of their proposed system. Also, the calculated value for the Accuracy of the proposed classifier was 90%.

Liu et al. in 2010 proposed a new graph-based framework, which combine topic information within random walk for key phrase extraction. They have built a Topical PageRank (TPR) on word graph to measure word importance with respect to unlike topics. After that, given the subject distribution of the document, they calculated the ranking scores of the words and extracted the first rank as keywords. They have done studies on two datasets; first dataset contains 308 news articles in DUC2001 with 2488 manually annotated key phrases. The other dataset includes 2000 abstracts of research articles and 19,254 manually assigned keywords. Their studies in terms of evaluation metrics on these datasets show that TPR accomplishes better performance than other baseline approaches (Liu, Huang, Zheng, & Sun, 2010).

Matsuo and Ishizuka proposed a new algorithm which can be apply on a single document without using such a massive corpus. There 20 authors technical papers in artificial intelligence research used. In this study, authors did not need a big corpus. Therefore, they focus on sentences. First, they did data preprocessing after this they extracted frequent terms and select top frequent terms with an arbitrary ratio. Secondly, they count number of co-occurrence terms to create a co-occurrence matrix. If two term stays in a sentence this two-word co-occurrence. The calculated value for precision was 0.51 which indicates the high quality of their proposed algorithm. Consequently, the beneficial of their algorithms is that the algorithm does not need any corpus and domain independent (Matsuo & Ishizuka, 2003).

In 1958, Luhn was introduced to keyword extraction for the first time, while he was doing research on the text abstract. Statistics-based keyword extraction methods includes word frequency, word cooccurrence frequency, etc. Some machine learning algorithms, support vector machine, CRF and so on. such algorithms are used for keyword extraction. Thanks to the link between keywords extraction and linguistics, research studies on linguistics, part of the conversation, grammar, syntax, semantics, etc. Including it has become increasingly important (Luhn, 1958).

The studies we summarized represent important studies we have obtained as a result of the literature review. These studies show that keyword extraction or suggestion are used to obtain meaningful and important information from huge data using text mining techniques. Therefore, in the remaining of this chapter we are explained these topics in more details, such as text mining, text clustering, text summarization etc. In addition, we have explained the process of keyword extraction and suggestion which is the main subject of our thesis.

2.2 Text Mining

In recent years, with the acceleration of technology studies, especially after the emergence of social networks and their continuous production of textual data, the data volume in digital media increases exponentially. Almost 20% (Hafez, 2017) of this incremental data is stored in a structured. Therefore, it is possible to obtain some statistical information by applying data mining on the data in this structured state or if the data is stored in any database, the desired information can be obtained easily by using database query language. However, the remaining 80% (Hafez, 2017) of the data in the digital environment is unstructured and this unstructured data contains very important information. Unfortunately, it does not seem possible to literally read the unstructured data and reveal meaningful information. Because the data we mentioned is not a page or a book, we are talking about millions of books, maybe billions of books. At this point, concepts such as text mining and machine learning gain importance.

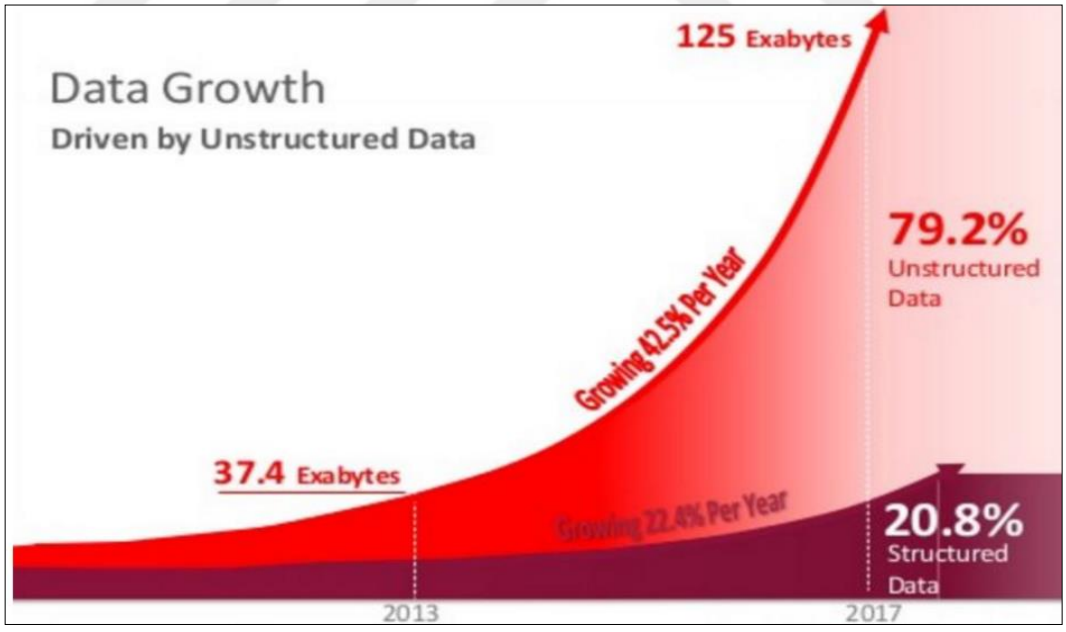


Figure 2.1 Increase of digital data over the years (ISD, 2014)

Text Mining is the method of automatically acquiring important knowledge from different textual data by the computer of previously unknown new information. A key element is the combination of more traditional experimental methods and the information extracted to form new realities or new hypotheses for further investigation.

Text mining is very different from what we are familiar with using web-based search engines. In an exploration, the users are classically looking for something inside a data that is already known and stored in a structured manner by others. However, the problem is to push aside all the materials that aren't currently related to your needs to find relevant information. The purpose of text mining is to notice unknown information, something that nobody knows and still cannot write (Gupta & Lehal, 2009).

Text mining is the method of analyzing and exploring textual data, often using software-assisted applications that can define concepts, patterns, topics, keywords, and other features within unstructured big data. Text mining has become a widely used method for data scientists who can analyze large unstructured data sets (Rouse, 2019).

In accordance with their explanations for text mining Figure 2.2 summarizes the subject well below.

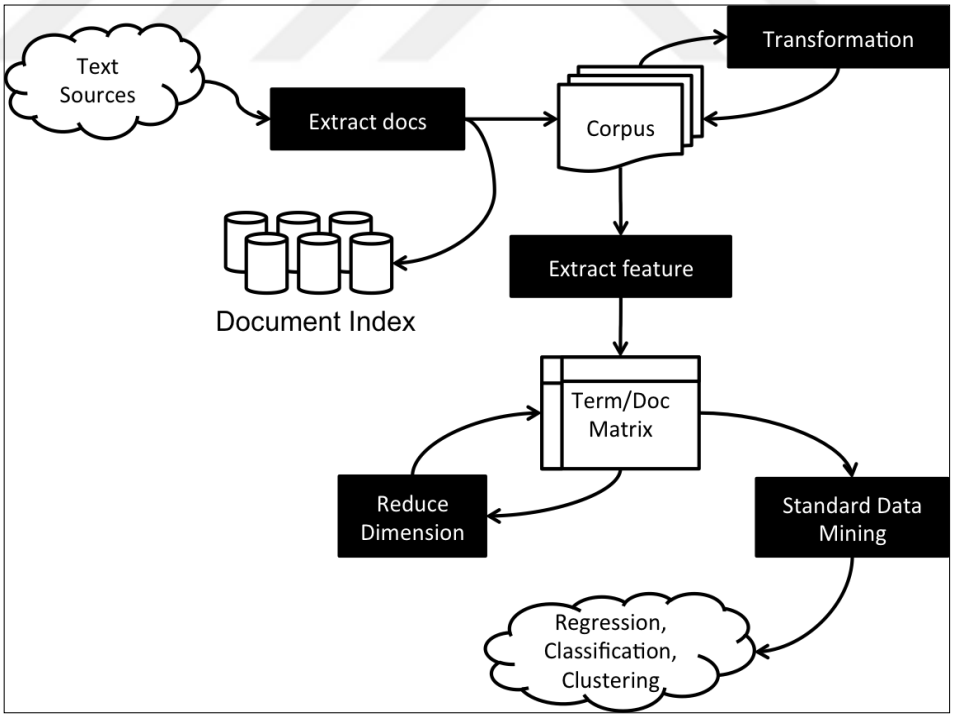


Figure 2.2 Text mining process (Truong, 2018)

Consequently, many methods are used in text mining. The commonly used methods are Naive-Bayes methods, deep learning, artificial neural networks, etc. Various statistical learning methods are used. In addition, there are many subsections in the text mining field. We will briefly explain some of the commonly used. We also explained the keyword suggestion process which is the most important part of our thesis.

Text Classification

Text classification, one of the most common fields of the study in text mining, is the process of assigning predetermined categories to text documents for natural language processing. Text classification involves designing optimal features to select the best possible machine learning classifiers. To date, almost all text classification techniques are based on words in which simple statistics of some sequential word groupings (like n-gram) generally perform best (Joachims, 1998).

Text Clustering

Text clustering is an unsupervised machine learning approach that does not have a predetermined class but searches together document groups (Aggarwal & Zhai, 2012). Clustering algorithms do not require training data, and algorithms generally perform more computations than supervised learning algorithms. The more similar the contents of documents in a cluster, the better the clustering quality (Cai & Sun, 2009).

Text Summarization

Another popular subject of text mining is text summarization, used to shorten long textual data. The aim is to make a consistent and regular abstract of the main theme in the document. It allows us to learn about the text by looking at the summary extracted from reading the entire text. Automatic text summarization is a major problem in machine learning and natural language processing (NLP). Machine learning

algorithms are often trained to understand textual data and filter useful information before extracting the necessary summary texts (Garbade, 2018).

Topic Modeling

Topic Modeling, a rule-based text mining technique that uses dictionary-based keyword search techniques. It is an unsupervised method to find many important words in the big textual data. Topic Modeling is a widely used technique for retrieving important information from unstructured texts (Newman, Chemudugunta, Smyth, & Steyvers, 2006). For example, companies such as Netflix and YouTube use Topic Modeling techniques to propose videos with content similar to the videos they watch. Moreover, Topic Modeling is used for various recruitment processes. For example, large e-mail datasets, customer reviews, and user social media account information are taken and topic modeling is applied to this data. It then allows they to extract the hidden attributes of job descriptions and match them to the right candidates (Bansal, 2016).

2.3 Keyword Extraction and Suggestion

Keyword extraction is the automatic definition of terms that best describe the main theme of a textual data. Keywords are used to describe phrases that represent the most relevant knowledge in text data. Keyword extraction and suggestion is an important issue in Text Mining, Natural Language Processing and Information Retrieval.

Keywords that we define as one or more words are a reflection of the document. Ideally, keywords refer to the main theme of a document. Keywords are commonly used in information retrieval systems because documents with well-defined keywords are very easy to access with queries in information retrieval (IR) systems (Rose, Engel, & Cramer, 2010). Jones and Paynter have developed Phrasier, a system that supports the use of keywords as a link between documents, lists documents related to the document's keywords, and allows the user to quickly access the relevant documents (Jones & Paynter, 2002). Gutwin et al. describes Keyphind, which uses keywords from textual data as the basic building chunk for an Information Retrieval System.

Keywords can also be used to increase the presentation of search results (Gutwin, Paynter, Witten, Nevill-Manning, & Frank, 1999). Andrade and Valencia provide a system that robotically describes function with keywords derived from scientific literature associated with a particular protein (Andrade & Valencia, 1998).

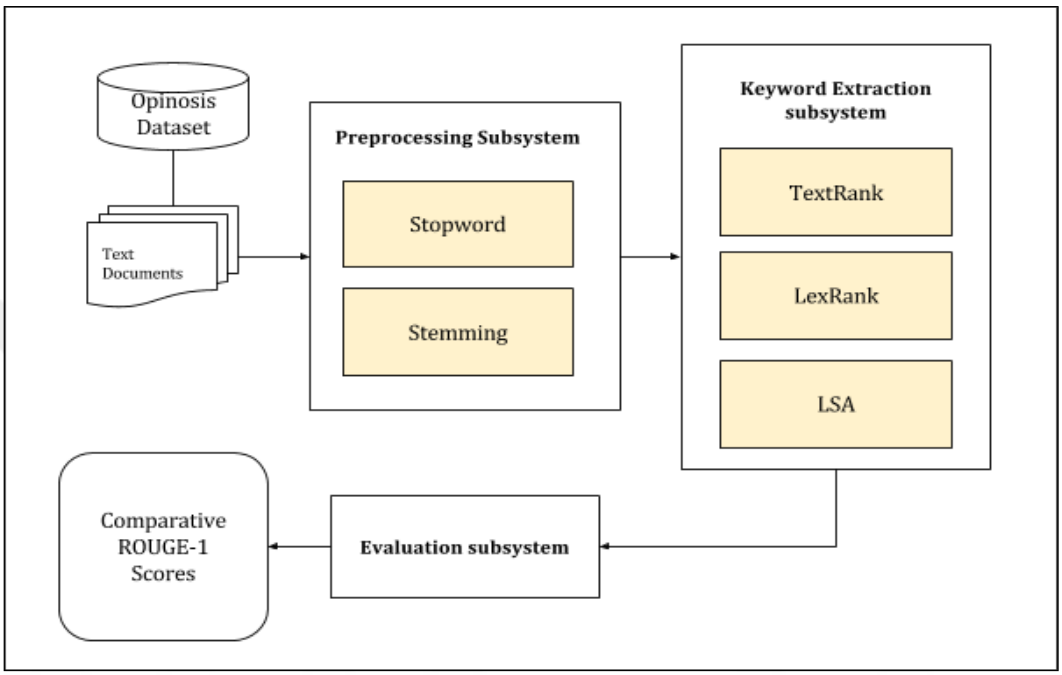


Figure 2.3 Keyword Extraction process (Sharma, 2017)

Keywords are important words or phrases within the textual data that identify the text. These keywords are like mirrors that show meaning of the text. When performing the keyword extraction process, the model given in Figure 2.3 is generally used. That is, it is possible to have an idea about the document or the text by means of well-defined keywords without reading the entire contents of the document. In recent years, the Internet has become the most used communication tool. With the expansion of the Internet, many social media tools and web-based applications have emerged. For example, Facebook, Twitter, Instagram. Using these applications, people produce millions of unstructured data every millisecond. We now have a lot of data that has become a valuable asset to many data scientist researchers. However, it is impossible for people to read and analyze these huge data. Of course, too many large unstructured text data have no keywords. If this were the keywords of the big data, we'd have an idea about that data without having to read it. However, manual keyword selection may not be able to clearly identify the text. At this point, we can automatically assign

keywords to these gigantic data by applying machine learning or text mining techniques. This way, we can immediately decide whether it is worth reading for us, or just a text of interest to us, simply by looking at the keywords of the texts. If there were no keywords, unfortunately we would have read the entire text, and if the text is not relevant to our area of interest, the time we spend is wasted.

As a result, almost all of the text mining, text mining subfields and keyword extraction methods mentioned above extract the important information in the big data by using the steps mentioned in Figure 2.4 below.

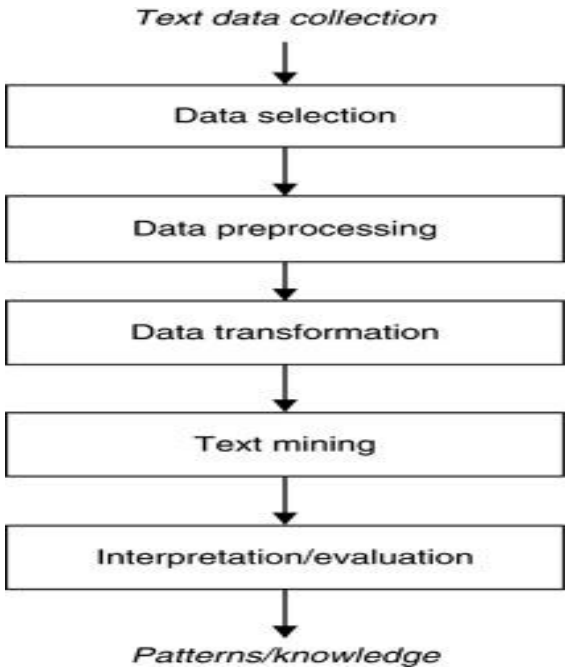


Figure 2.4 Steps of keyword discovery from text data (Cai & Sun, 2009)

CHAPTER THREE
USED TOOLS AND IMPLEMANTATION

In this chapter, we describe the technologies we used in our thesis, the tools and the development processes of our application, and the methods we use in detail under the titles of used technologies and implementation.

3.1 Used Tools

This section describes the technologies used to develop the application of our thesis. How the data we will use in our application is obtained from the relational database PostgreSQL and the tables used for the data in this database are given. In addition, Elasticsearch, which is commonly used in IRS, is similar to a distributed database structure and used as a search engine.

We also used Python 3.7, one of the programming languages widely used by data scientists and even used as popular programing language all over the world to write our application. We used the flask-web framework which is commonly used in web-based applications to write the user interface of our application. On the front-end we used html 5 and css3 technologies.

Finally, we used JavaScript's library named chart.js to analyze and visualize the evaluation metrics of our application.

3.1.1 PostgreSQL

PostgreSQL is a general-purpose relational database management system which is an advanced open source database system. PostgreSQL is designed to work on UNIX-based platforms (PostgreSQL Tutorial, 2019). Due to the platform independence of PostgreSQL, it can work on other operating systems. PostgreSQL is also an open source and free database management system software. Therefore, you can use it and change its the source code. PostgreSQL runs stable, so if you develop PostgreSQL-

based applications, the cost is low compared to other database management systems. In addition, PostgreSQL has several advanced properties that other database management systems offer (PostgreSQL About, 2019) . There are many free and open source management system tools for PostgreSQL database management. For example, HeidiSQL, PgAdmin and Adminer. We used the Adminer management system in this study.

3.1.2 Elasticsearch

Elasticsearch is an open-source search and analysis engine that is distributed for all types of data, whether structured or unstructured (Akdal, Keskin, Gül, Ekinci, & Kardas, 2018). Elasticsearch is built on Apache Lucene, an open source information collection library written in Java (What is Elasticsearch, 2019).

Compared to other storage systems Elasticsearch can be described as a distributed, high availability, JSON document-oriented storage solution targeted at full text searching (Balamaci, 2016).

In terms of data storage, Elasticsearch is file based and distributed search engine which is not similar to relational databases. The Figure 3.1 shows the concepts of a relational database and the corresponding concepts of Elasticsearch.

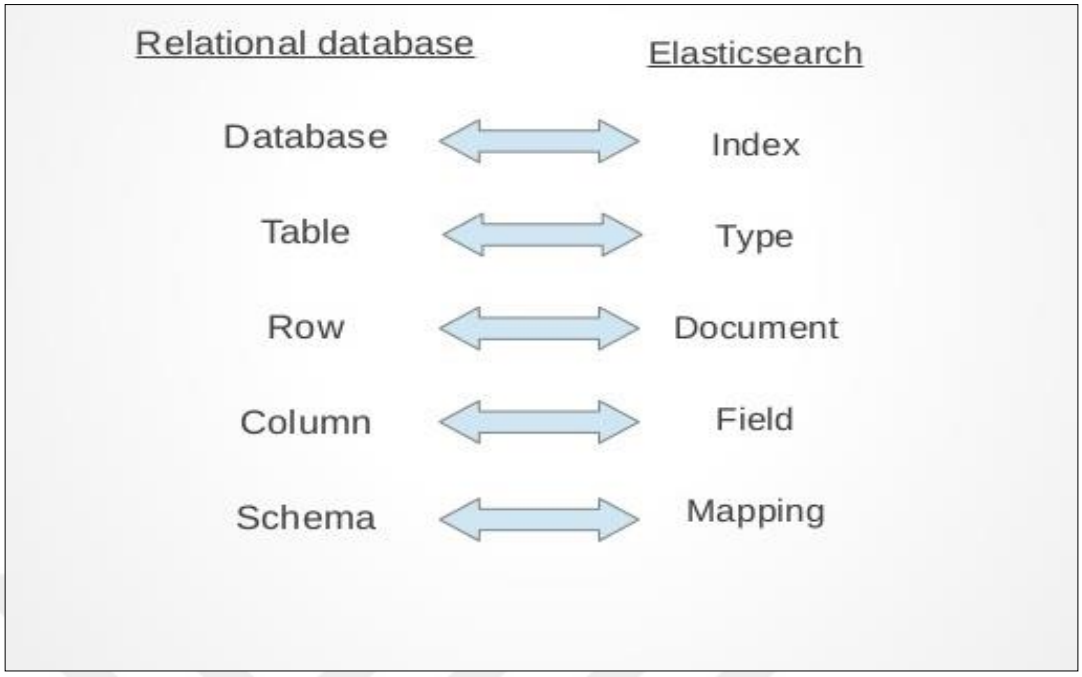


Figure 3.1 Relational DB vs Elasticsearch (Mohapatra, 2015)

Elasticsearch information is one of the most important reasons for the use of retrieval systems, especially in text search fields, because it has a strong indexing structure. The Figure 3.2 shows the internals inverted indexing structure of Elasticsearch.

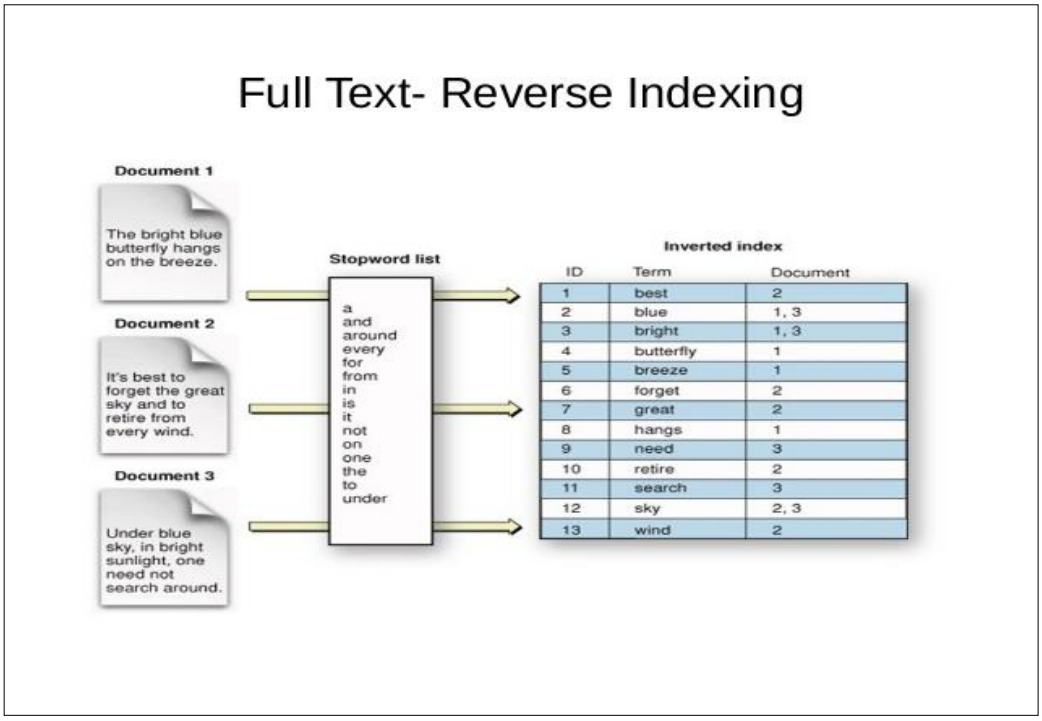


Figure 3.2 Elasticsearch internals-inverted indexing process (Mehta, 2014)

As a result, Elasticsearch creates an reverse indexed list as shown in Figure 3.2. It then records the information which word is used in which document with key value logic. Therefore, the search process produces very fast results in such systems.

Finally, if we give an example of the operation of Elasticsearch system;
When we type query = "blue sky" the Elasticsearch will look at the index table and will find that blue available in “1,3" and sky = "2,3" documents. Elasticsearch returns document 3, which is the intersection of these two sets of results.

3.1.3 Python 3.7

Python is a high-level, object oriented and interpreted programming language. This language is easy to learn thanks to its easy syntax and is very readable as it is very close to the spoken language. Python allows the handling and reuse of code written in program modularity. In addition, the Python interpreter and the general standard library are available free of charge for all major platforms and can be freely distributed. Powerful data structures make it very attractive to use as an adhesive tongue for Fast Application Development and to connect existing components together. Because of this feature, it is the most used programming language by data scientists (Rossum, 2016).

Because it is an interpreted language, the edit-test-debug cycle is very fast. Debugging is easy with programs written in Python: An error or incorrect input never causes a segmentation fault. Instead, when the interpreter finds an error, it throws an exception. When the program does not catch the exception, the interpreter prints the stack trace (Maisam, 2019).

The idea that coding in a high-level language could greatly improve productivity began to emerge in the 1950s (Fairhead , 2017). Many data scientist researchers encounter one or more interpreted scientific computing environments in the early stages of their profession. Python, on the other hand, offers these researchers a very good environment for writing scientific applications with unique features. Therefore,

the Python programming language becomes an entertainment environment for programmers and researchers rather than being a high-level language for doing things easily (Oliphant, 2007).

3.1.4 Flask Web Framework

Flask is the most popular micro framework written in python programming language for developing small or medium web-based software. This framework, which is commonly used among web-based software developers, is ideal to do something quickly and bring out certain results in time. It is also used in web services for fast results. Flask is a high-performance framework that can be learned quickly due to the syntax of python. Of course, the flask doesn't stop counting. One of the most important features is the microframework, so MVC (Model View Controller) logic can be configured very easily (Hunt-Walker, 2018). We show that the flask logical diagram in Figure 3.3.

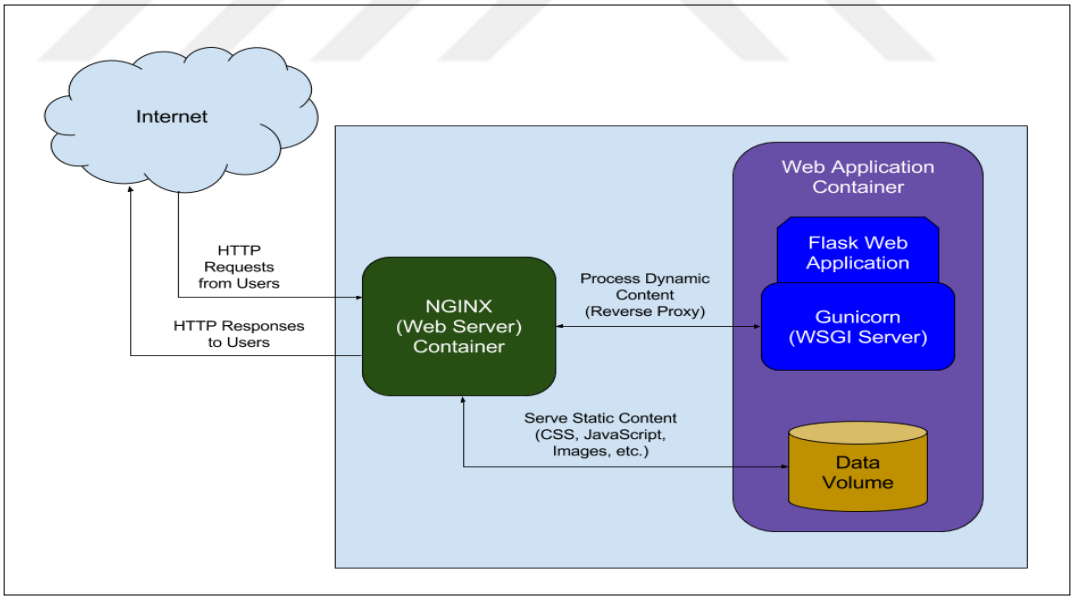


Figure 3.3 Flask web framework diagram (Patrick, 2018)

3.2 Implementation

In this section, we will explain in detail getting data, application architecture and method that we use when writing the application of our thesis. We have discussed keyword extraction on Medline data as a method. While searching literature, we came across a lot of studies in the field of keyword extraction. Most of these studies, while calculating word frequencies in the document, accept high-frequency words as keywords, while others perform keyword assignment by looking at the co-occurrence of words. However, the most important feature that distinguishes our study from the studies done before is that we offer a collaborative approach solution. This collaborative system benefits from the efficiency of Elasticsearch, one of the information retrieval systems, as well as the term frequency calculation as in previous studies.

3.2.1 Data Gathering

We used Medline dataset from PubMed system in this study. PubMed is a free-of-charge database of the US National Institute of Health, where a number of biomedical publications are indexed together (Hunter & Cohen, 2006). Publications of academic journals with specific characteristics are examined and abstract texts and publication information are added to the database. PubMed publications can also be used for evidence-based medicine (Leaman & lu, 2004). There are around 28 million articles in the PubMed System. All of this data, approximately 150 GB in size, was previously downloaded with special software by members of Dokuz Eylül Multimedia Information Retrieval (DEMIR) group and then they this gigantic data uploaded to the PostgreSQL database, a relational database. In Figure 3.4, the entity relationship diagram of the database created for the Medline dataset in PostgreSQL is shown.

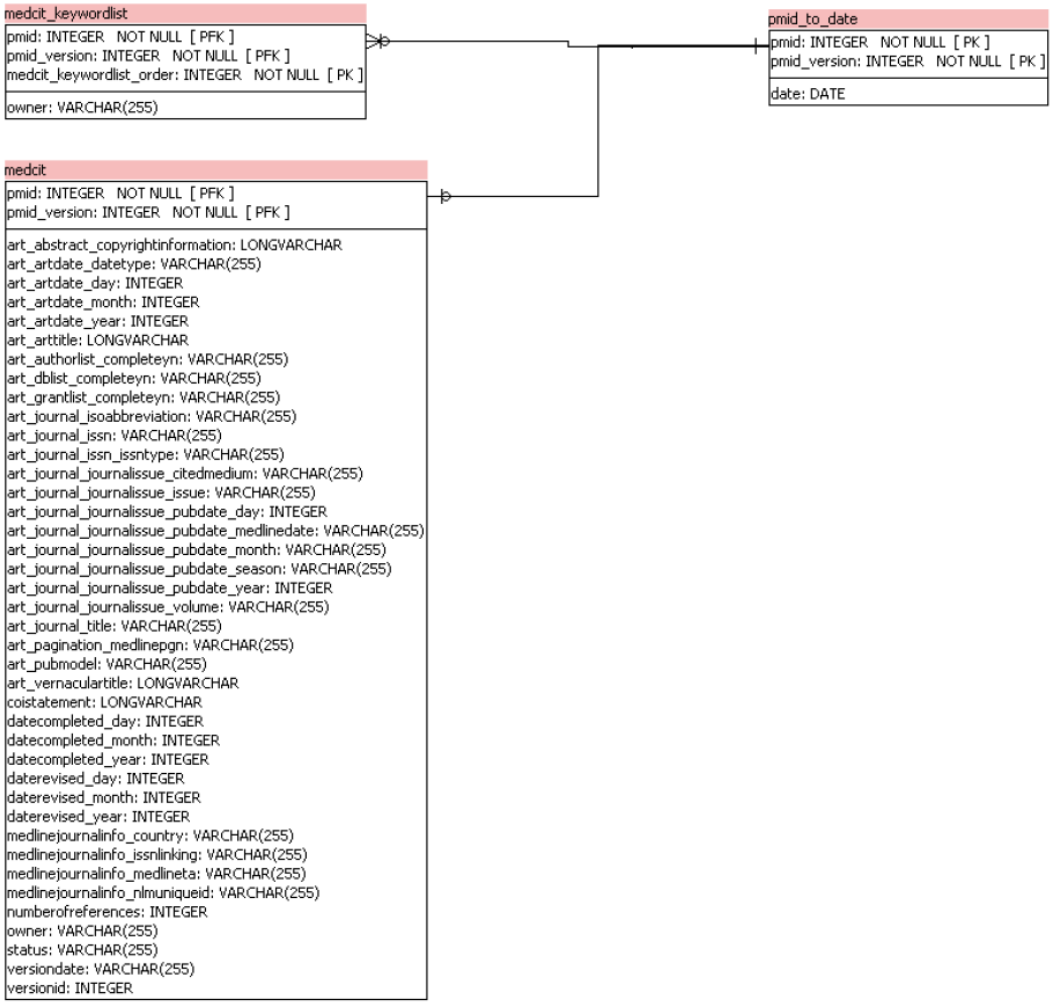


Figure 3.4 PostgreSQL PubMed database diagram

There are so many articles in PostgreSQL database. Due to that huge number of articles we decided to get a portion of this data. Therefore, we have created a SQL query to obtain a subset of the Medline articles on our PostgreSQL database. Our SQL statement is **"SELECT pmid, title, abstract, keywords FROM public. article WHERE article.completedyear=2018 AND abstract IS NOT NULL AND keywords IS NOT NULL ORDER BY pmid DESC".** After running this query, we obtained 80,000 articles, we separated these articles into two groups. First group was our like test data containing 10,000 articles, and the second group was our training data. As a result of this query, a cross section of the data set we received from PostgreSQL is shown in the Table 3.1.

Table 3.1 Result set from PostgreSQL query

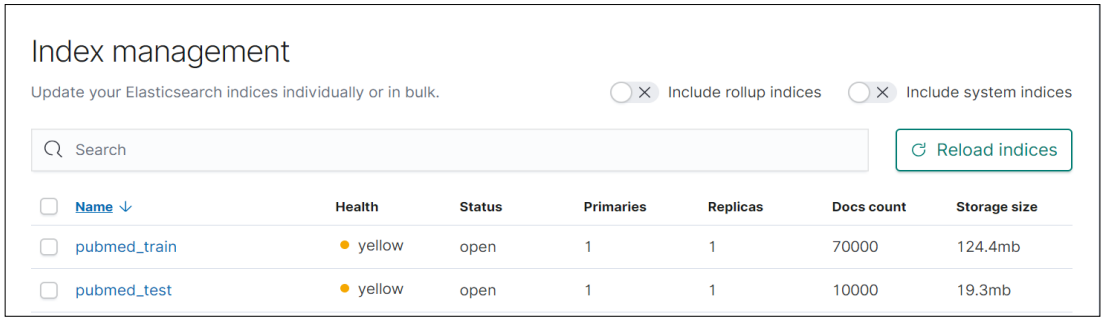
pmid	title	abstract	keywords
30462630	Lead in Spices, Herb ...	The number of pediatric cases ...	Retrospective Studies, Housing, Child, P ...
30462629	Self-Directed Walk W ...	Arthritis occurs in 27% of adu ...	Montana, Self Report, Aged, Walking, Pro ...
30462626	Prevalence of Amyotr ...	Amyotrophic lateral sclerosis ...	Aged, 80 and over, Registries, Risk Fact ...
30461713	Perspective on cultu ...	With an emphasis on Chinese se ...	Delivery of Health Care, Aged, Health Se ...
30461712	Mass shootings: A ca ...	Nurses everywhere may someday ...	Humans, United States, Hospitals, Wounds ...
30461711	Removing epidural ca ...	Short-term epidural analgesia ...	Device Removal, Pain, Analgesia, Epidura ...
30461710	Understanding the ro ...	Collaboration between nurses a ...	Nurses, Professional Role, Nursing Asses ...
30461708	Make connections by ...	Although some may feel hesitan ...	Cost-Benefit Analysis, Societies, Nursin ...
30461619	Medullary thyroid ca ...	No sign of relapse or metastas ...	Thyroid Neoplasms, Female, Breast Neopla ...
30461618	Cerebrospinal fluid ...	The thin-slice computerized to ...	Postoperative Complications, Frontal Sin ...
30461617	Surgical correction ...	We present a case of a 46-year ...	Postoperative Complications, Lip, Esthet ...
30461616	Effects of motivatio ...	The aim of our study was to qu ...	Orthodontists, Patient Education as Topi ...
30461615	Equipment failure of ...	A 16-year-old boy, 85 kg in we ...	Midazolam, Anesthetics, Inhalation, Male ...
30461614	Comparison of the ef ...	Polled results yielded that po ...	Esophageal Neoplasms, Esophageal Squamou ...
30461613	Three mutations of a ...	A 24-year-old Chinese female w ...	Enzyme Replacement Therapy, Gaucher Dise ...
30461612	Efficacy and safety ...	This study aimed to explore th ...	Liver Function Tests, Muscle, Skeletal, ...
30461611	Interleukin-6 for ea ...	The overall pooled sensitivity ...	Humans, Early Diagnosis, Fetal Membranes ...
30461610	Comorbidities, risk, ...	Cancer patients can be well-co ...	Risk Factors, Aged, Continental Populati ...
30461609	Lymphocyte hydrogen ...	The aim of the study was to id ...	Mucocutaneous Lymph Node Syndrome, Femal ...
30461608	Improve the ethical ...	Recently, there is an increasi ...	Registries, Research Support as Topic, I ...
30461607	Assessing the method ...	The methodological and reporti ...	Conflict of Interest, Publications, Huma ...
30461606	Is dynamic locking p ...	We searched Pubmed, Embase, We ...	Humans, Prosthesis Design, Hip Fractures ...
30461605	A meta-analysis and ...	Our findings provide evidence ...	Pancreatitis, Acute Necrotizing, Humans, ...
30461604	Efficacy of rational ...	The researchers adopted a grou ...	Female, Health Education, Risk Reduction ...

We needed to import the data generated as a result of the PostgreSQL query to Elasticsearch. Therefore, we exported the data from the query to Microsoft Excel format. While doing research on how to transfer the data we received from PostgreSQL to Elasticsearch, we met an application called Exelastic.jar. Exelastic.jar is an application developed with java programming language to be used in windows operating systems to transfer data in Elasticsearch in XLSX / XLS / CSV file formats. Can be used with Kibana or other visualization applications. The application comes with a web interface to simplify data transfer.

As a result of our query we have previously divided the data into two parts as test data and train data. Our goal here is that Elasticsearch does not know the articles we will use as queries. Otherwise it will return the same article to us as a result. Then, using Excelastic.jar, we transferred both files to the Elasticsearch indexes pubmed_test and pubmed_train.

After exporting the non-NULL data as CSV in PostgreSQL, it was necessary to convert this file to Excel format and then transfer it to Elasticsearch. We finished converting this file to Excel and finally uploading the data to the Information Retrieval System. As a result of these processes, we created the information of the indexes we created in Elasticsearch by using the kibana tool. The table of Elasticsearch indexes is shown in Table 3.2, the information of pubmed_test index is shown in Figure 3.5 and the information of pubmed_train index is shown in Figure 3.6.

Table 3.2 Elasticsearch index table



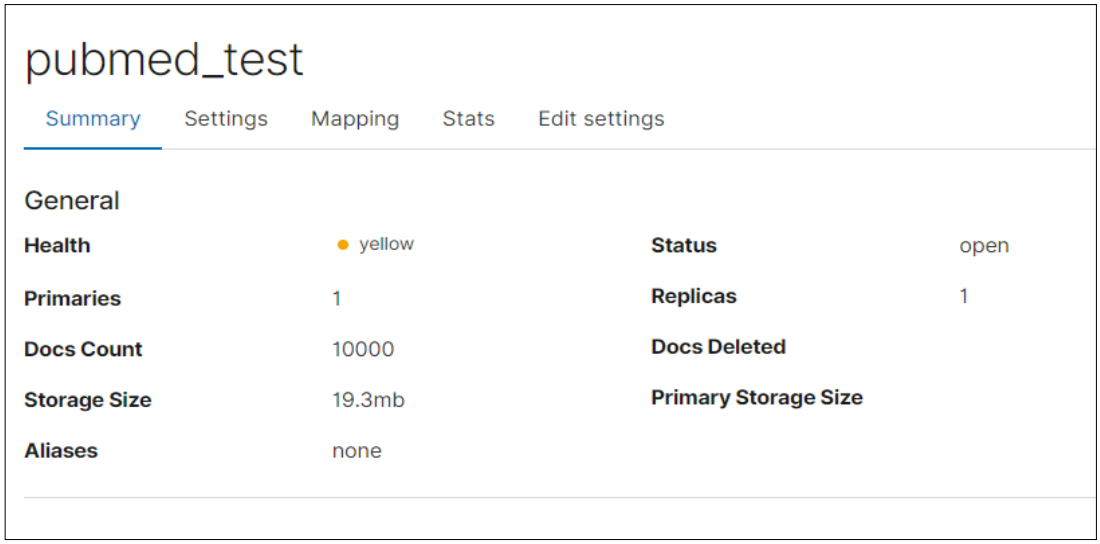


Figure 3.5 PubMed test index information

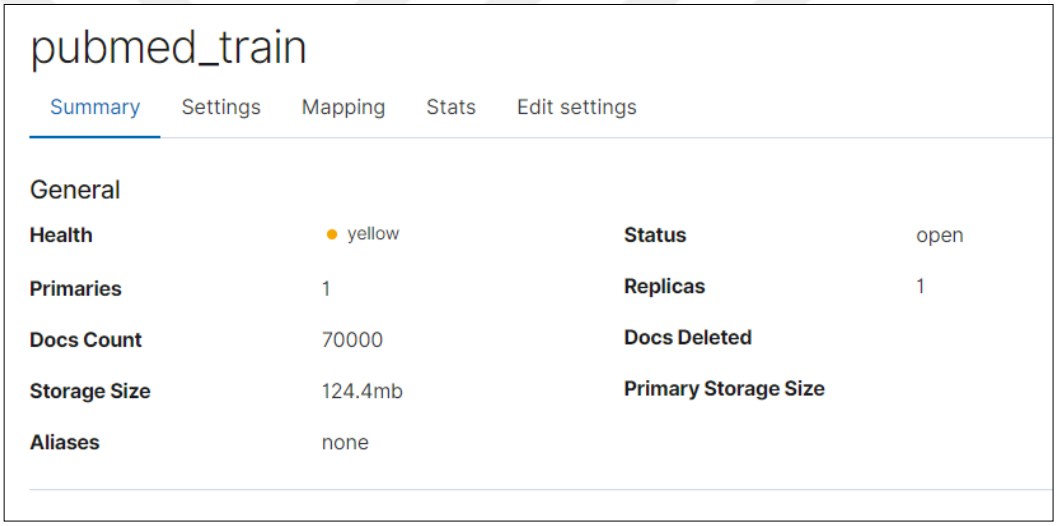


Figure 3.6 PubMed train index information

When we first time connected to Elasticsearch with the Python programming language and retrieved the data randomly then printed it on the screen, the program sometimes gave an error. When we examine the error in detail, we noticed that some data do not have keywords attribute. The reason for this is that when we convert the CSV (Comma Separated Value) data from PostgreSQL into Excel, some records' keywords field is deleted and null. Therefore, Elasticsearch did not create a keyword attribute for records with a null field when importing data to Elasticsearch. When we try to access this attribute, we normally get errors. As a result, we reviewed our data

and transferred Elasticsearch with no null fields. An example of records without a keyword field is shown in Figure 3.7.

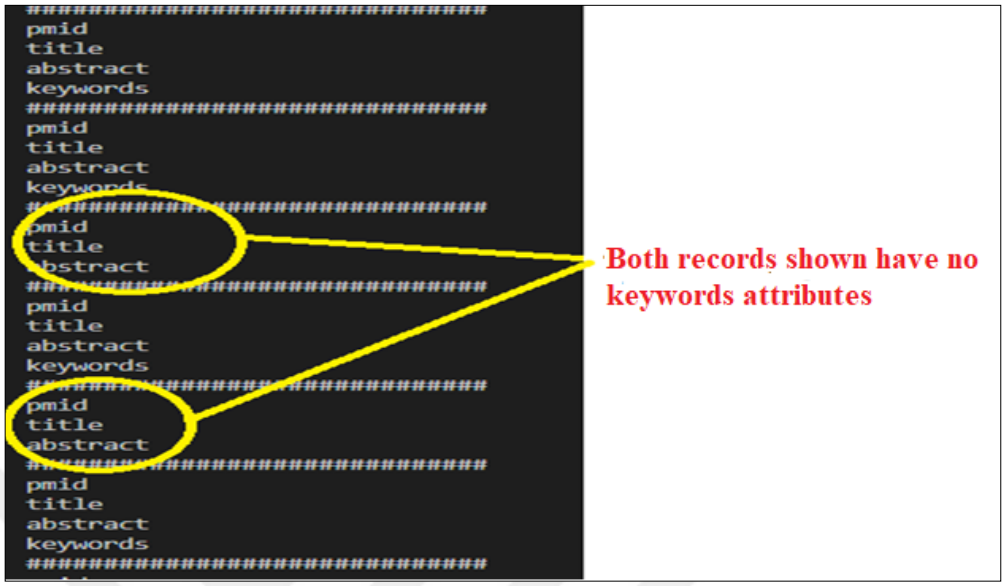


Figure 3.7 Records without keywords attribute in Elasticsearch index

3.2.2 Methods

First, we set up an Information Retrieval (IR) Systems for a set of research articles. We used subset of PubMed dataset containing 80,000 articles which they are written in year 2018. Then, abstract of a new article is given as query to IR systems, and received a set of documents similar to given query. Then, we considered first part of the results set. We investigated two different approaches: the first *k* documents or documents whose similarity score is greater than a threshold value of *t*. We called *k*-NN, and *t*-NN, respectively. After choosing similar document set, we consolidated the keywords and maintained keyword frequencies. Then, we assign the top *n* keywords to the new article. We described the proposed approaches in detail in this section. We also investigated different *k*, *t* and *n* values maximizing the performance of the system.

3.2.2.1 k-NN Method

In the *k*-NN approach we send the abstract of the new article with constant *k* (the size of the returned list) to IR system which will return the most *k* similar articles

according to the given abstract. Then we will apply term frequency to the returned result set and accomplish keywords list. In this list we assign the top *n* (Number of Keywords) keywords to the new article. We obtain the *k* and *n* parameters by applying our methods. We performed a benchmark test to calculate the precision and recall values for the *k* value. At the end of this test we applied our method by selecting the most optimal *k* and *n* values considering the precision and recall values. Figure 3.8 shows the flow diagram of the algorithm we designed for *k*-NN.

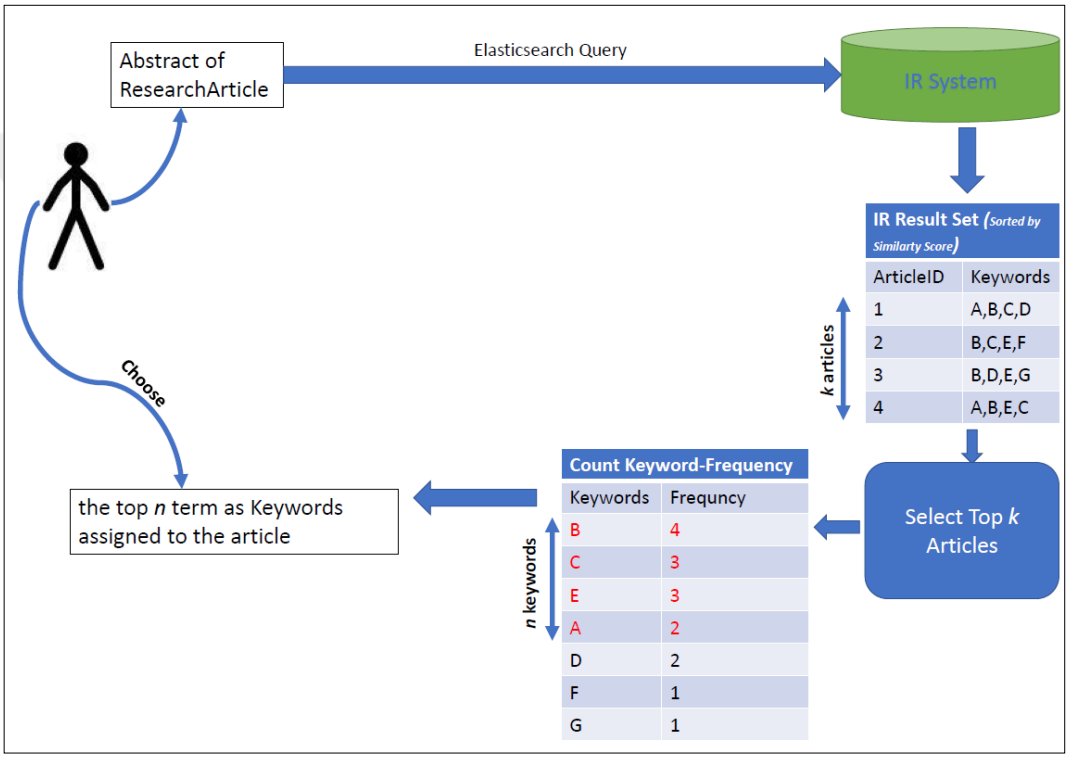


Figure 3.8 Blog diagram of *k*-NN aproach

Figure 3.8 shows block diagram of the system we developed using the *k*-NN method. For users who will use this application we have designed a simple interface as seen in Figure 3.9.

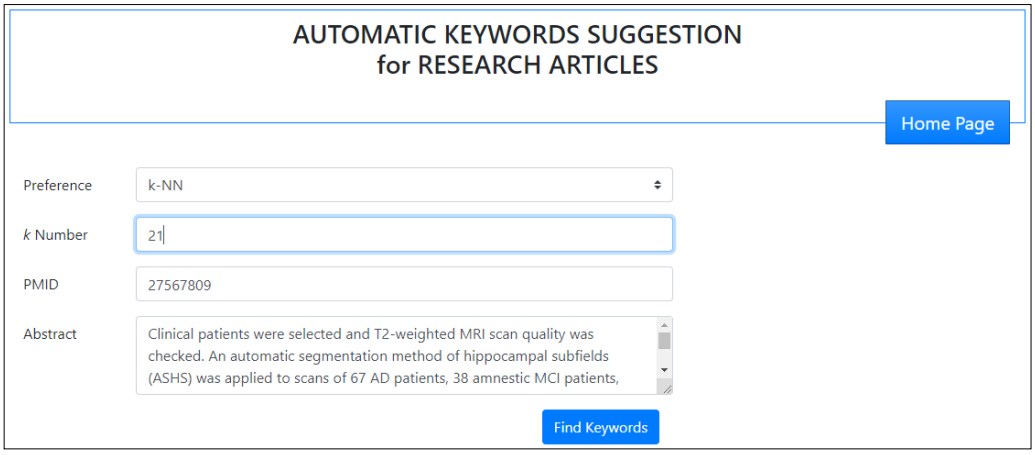
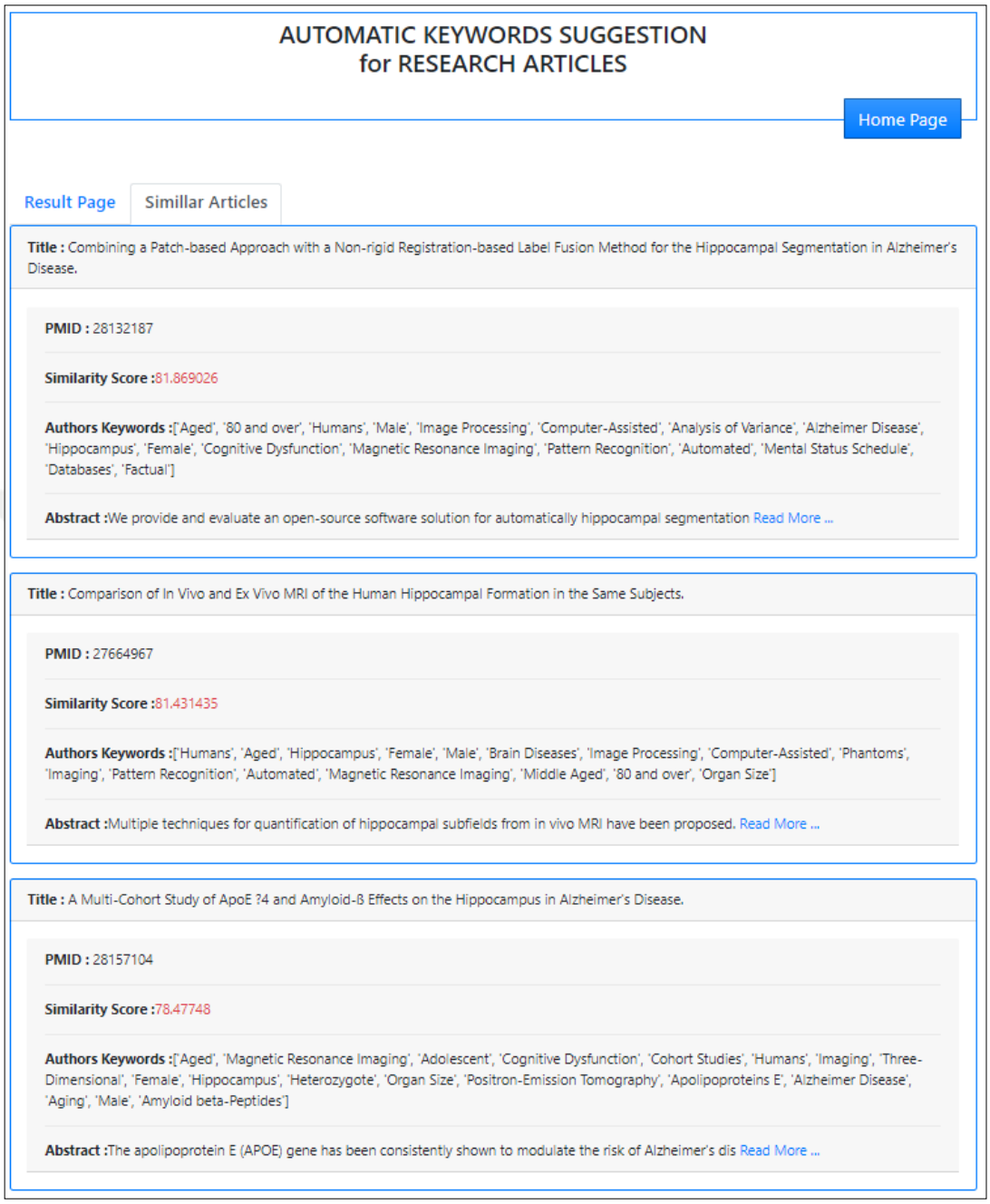


Figure 3.9 *k*-NN user interface

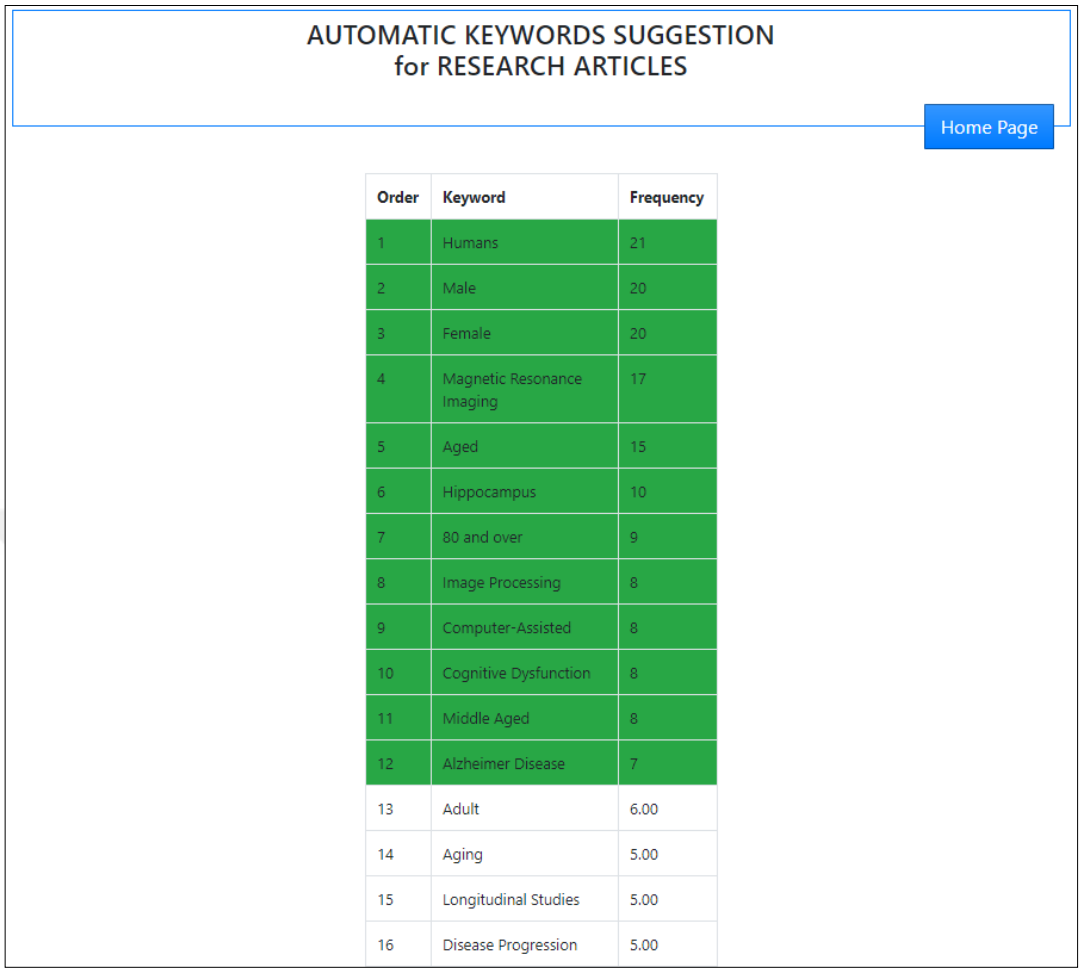
As we see in Figure 3.9, we filled the fields that appear in the user interface, *k* was set to value of 21, PMID which presents the id of the PubMed article was set to 27567809 and in addition to that we have entered the abstract of the article. In the system testing phase, PMID is used to get the author's keywords to compare them with the keywords suggested by our system. Finally, when we click the Find Keywords button, the information retrieval system returns a list of 21 articles that are the most similar to the abstract we have filled. A part of this list is shown in Table 3.3. Moreover, the PMID and the abstract of the article we type are recorded in the Elasticsearch pubmed_test index. Therefore, the system searches for similar articles in the pubmed_train index. In this way, the article whose pmid is given is excluded from the list of similar articles.

Table 3.3 Result set of the most similar articles for k=21



We have applied the k-NN method that we have developed to the articles set in Table 3.3, it calculated the frequencies of the keywords that belong to the articles, the results we have obtained are shown in Table 3.4.

Table 3.4 Keyword list suggested by the *k*-NN method



Order	Keyword	Frequency
1	Humans	21
2	Male	20
3	Female	20
4	Magnetic Resonance Imaging	17
5	Aged	15
6	Hippocampus	10
7	80 and over	9
8	Image Processing	8
9	Computer-Assisted	8
10	Cognitive Dysfunction	8
11	Middle Aged	8
12	Alzheimer Disease	7
13	Adult	6.00
14	Aging	5.00
15	Longitudinal Studies	5.00
16	Disease Progression	5.00

We have done a lot of benchmark tests. According to these tests, we found that the most appropriate number of keywords in the *k*-NN method is 11. We explained in detail how we did these tests in the experimentation and result chapter. Therefore, we recommend the first 11 words of the keyword's list shown in Table 3.4 to users as keywords. We have also compared the keywords that the author has assigned with the keywords proposed by our system and calculated how many of them are in common by using the Jaccard Similarity measure as shown in Figure 3.10.

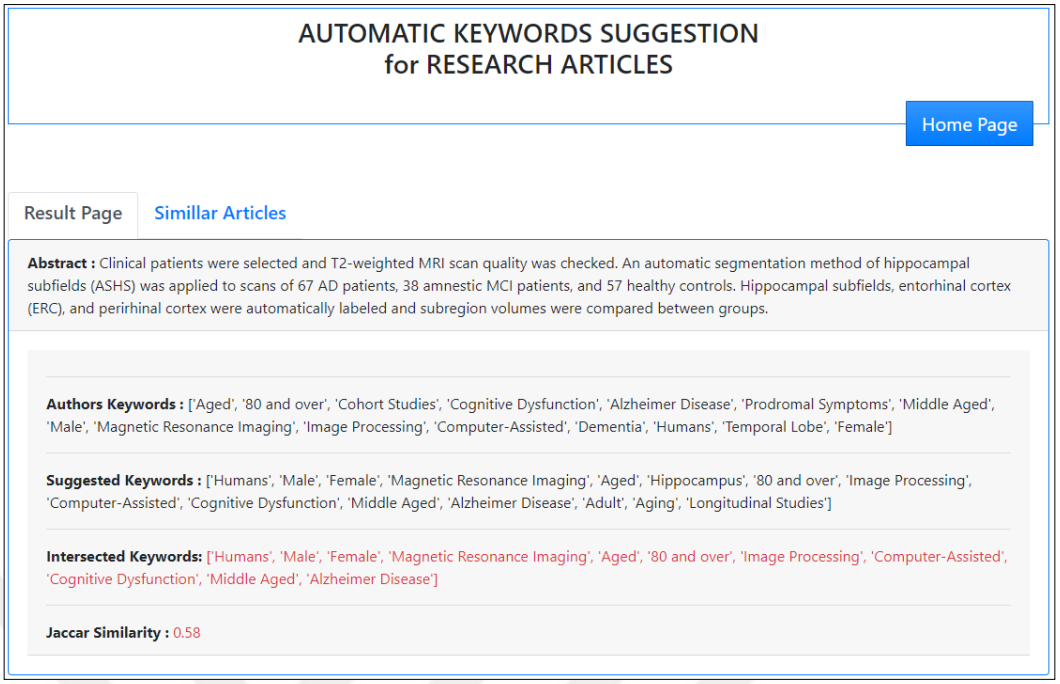


Figure 3.10 Result of *k*-NN algorithm

3.2.2.2 t-NN method

The system we developed uses two different approaches: *t*-NN frequency and *t*-NN score. In the first approach, we calculate the frequencies of the keyword numbers of articles that have a similarity score on a threshold value (*t*) determined. In the second approach, we weight the keywords by multiplying the keywords of the articles with similarity scores. Since the working principle of both approaches is similar, in this section we only describe the *t*-NN-Score approach.

3.2.2.2.1 t-NN-Score. In the t-NN-Score approach, as in *k*-NN, we set the similarity score as a threshold to obtain the result set returned by Elasticsearch. So, we retrieve articles which their similarity score is greater than 80. After that we apply 2 different methods with *t*-NN-Score approach. First, we use the term frequency technique at the keywords of these articles. This means count each keyword, calculate its frequency and pick up the top *n* keywords. Secondly, we create a keywords list for each article by multiplying the keyword with its similarity score. Figure 3.11 shows the flow diagram of the algorithm we designed for *t*-NN.

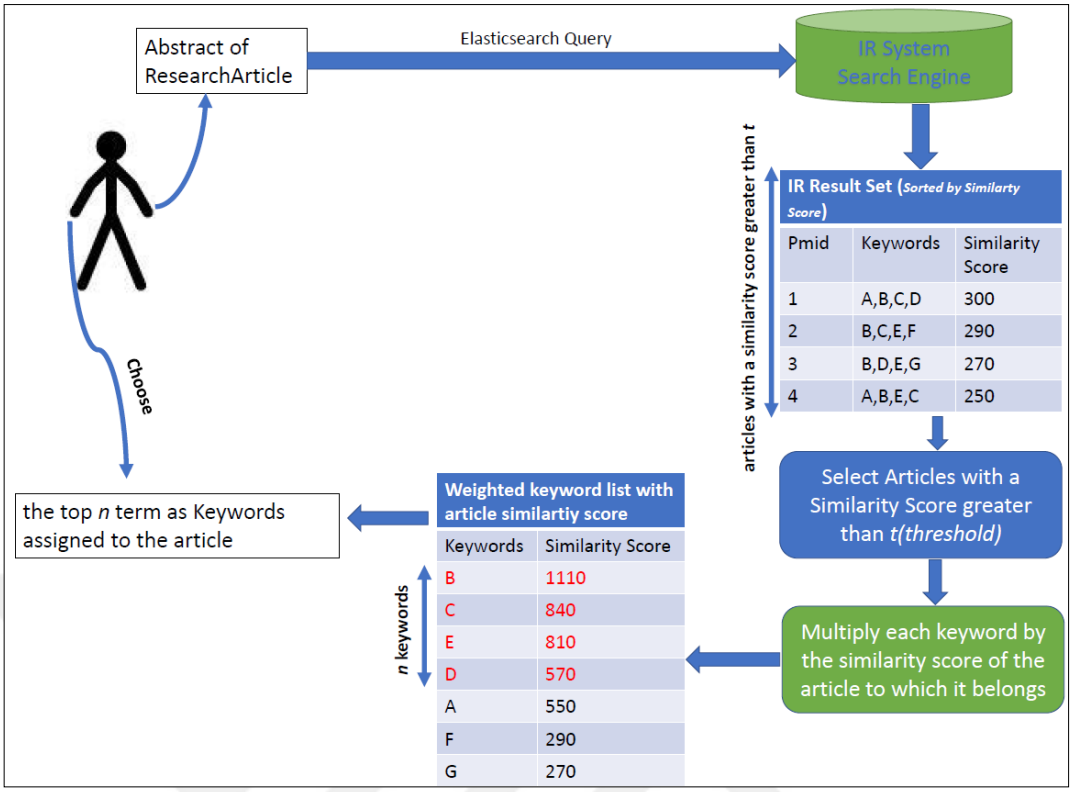


Figure 3.11 Blog diagram of *t*-NN aproach

Figure 3.11 depicts general steps of the proposed *t*-NN approach. We developed a simple user interface that makes it easier for users to use this application. Figure 3.12 shows the user interface.

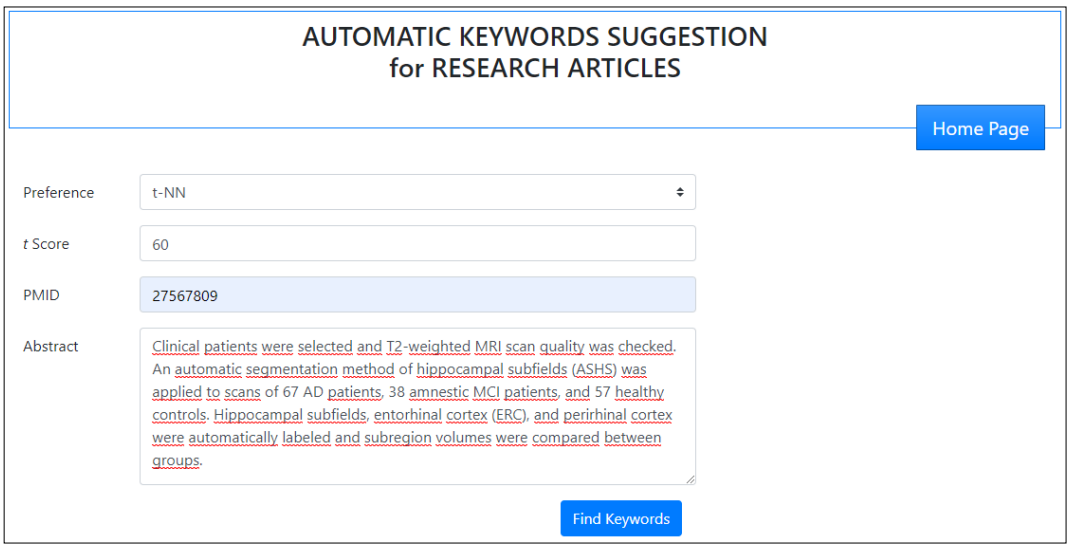
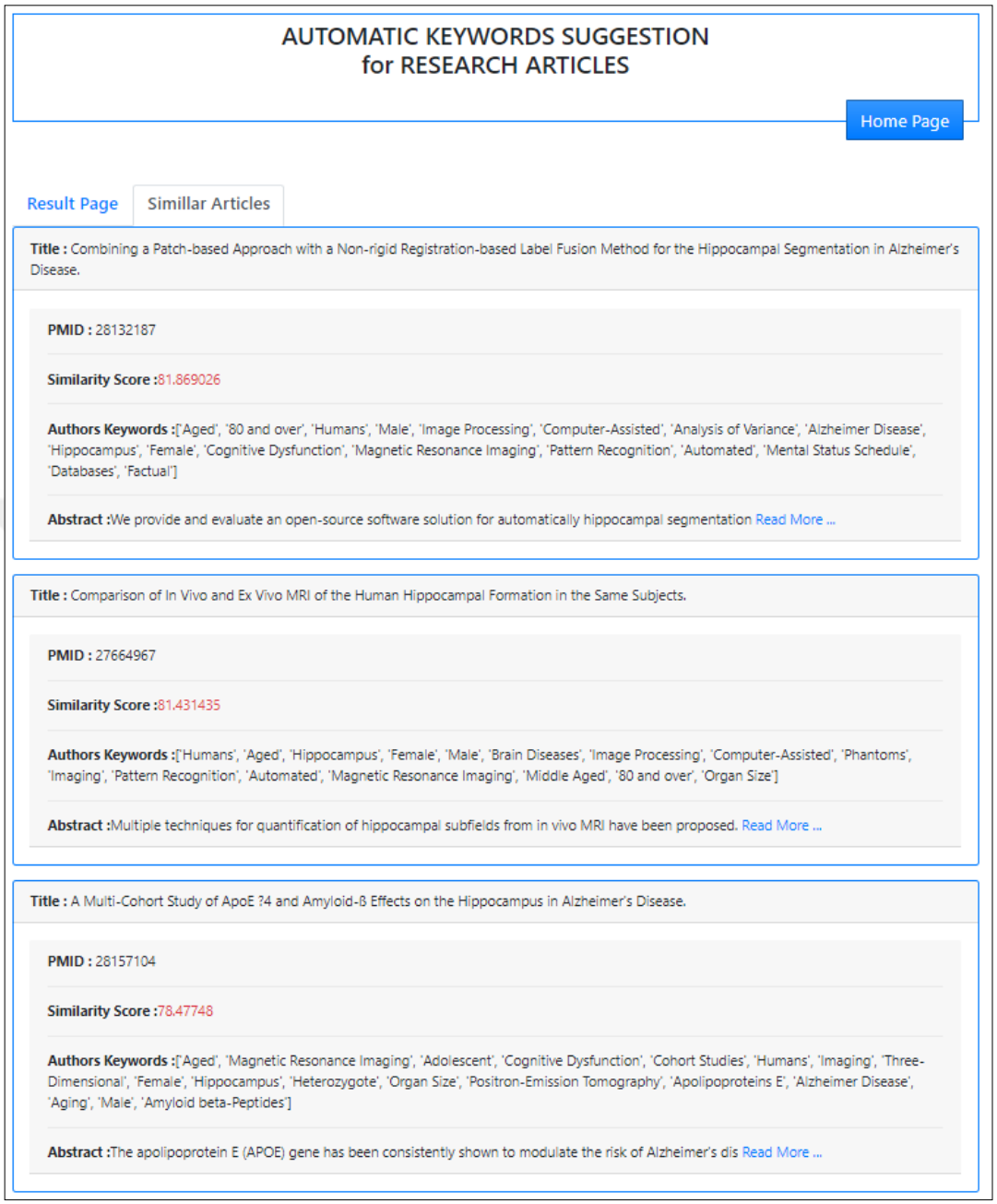


Figure 3.12 *t*-NN user interface

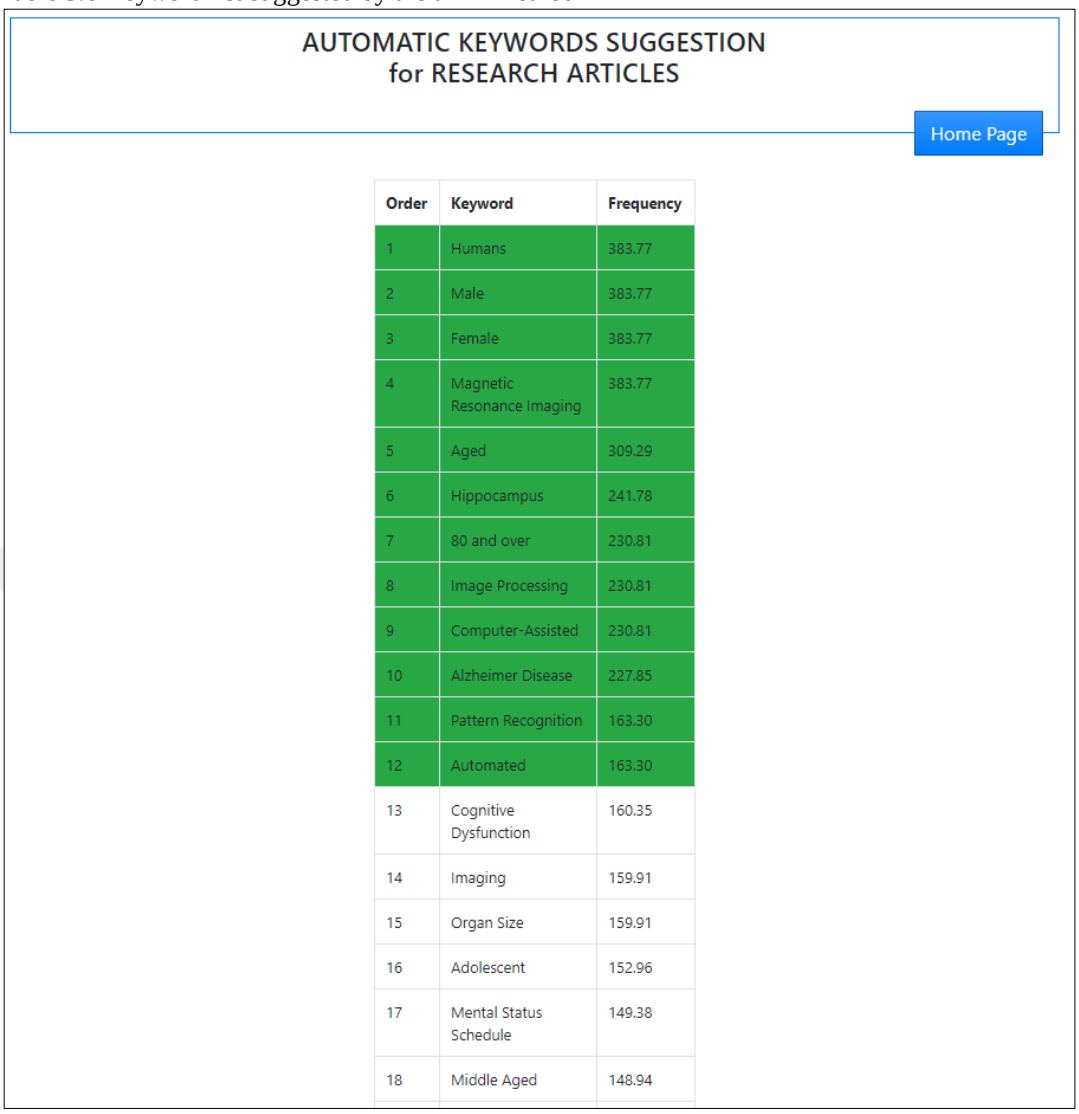
As we see in Figure 4.9, we fill in the fields that appear in the user interface respectively, the value of threshold *t* was set to 60, the abstract of the article and PMID was filled respectively, PMID was set to 27567809. In the system testing phase, we use the PMID to get the author's keyword and compare them with the keywords suggested by our system. From our benchmark tests we have seen that the optimal threshold of *t* variable was 80. However, when we set the value of *t* as 80 it gives the same result to *k*-NN method for this article. In order to compare *t*-NN and *k*-NN approaches in a more accurate way, and for better understanding for the differences between them, one can set threshold value *t* to 60. Finally, when clicked the Find Keywords button, the information retrieval system returns a list of articles that are the most similar to the abstract we have provided with a similarity score greater than the value of threshold *t*. We have shown a part of this list in Table 3.5.

Table 3.5 Result set of the most similar articles for $t > 60$



When we apply the developed *t*-NN method to the articles shown in Table 3.5, it calculates the frequencies of the keywords by multiplying the similarity scores of the keywords. After this calculation, the recommended keyword list we obtained is shown in Table 3.6.

Table 3.6 Keyword list suggested by the *t*-NN method



Order	Keyword	Frequency
1	Humans	383.77
2	Male	383.77
3	Female	383.77
4	Magnetic Resonance Imaging	383.77
5	Aged	309.29
6	Hippocampus	241.78
7	80 and over	230.81
8	Image Processing	230.81
9	Computer-Assisted	230.81
10	Alzheimer Disease	227.85
11	Pattern Recognition	163.30
12	Automated	163.30
13	Cognitive Dysfunction	160.35
14	Imaging	159.91
15	Organ Size	159.91
16	Adolescent	152.96
17	Mental Status Schedule	149.38
18	Middle Aged	148.94

We have seen that the optimal number of keywords to be recommended in our benchmark tests is 11. Therefore, we recommend the first 11 words of the keyword list shown in Table 3.6 to users as keywords. We also compare the keywords that the author has assigned with the keywords proposed by our system and calculate how many of them are common by calculating the Jaccard Similarity measure as shown in Figure 3.13**.**

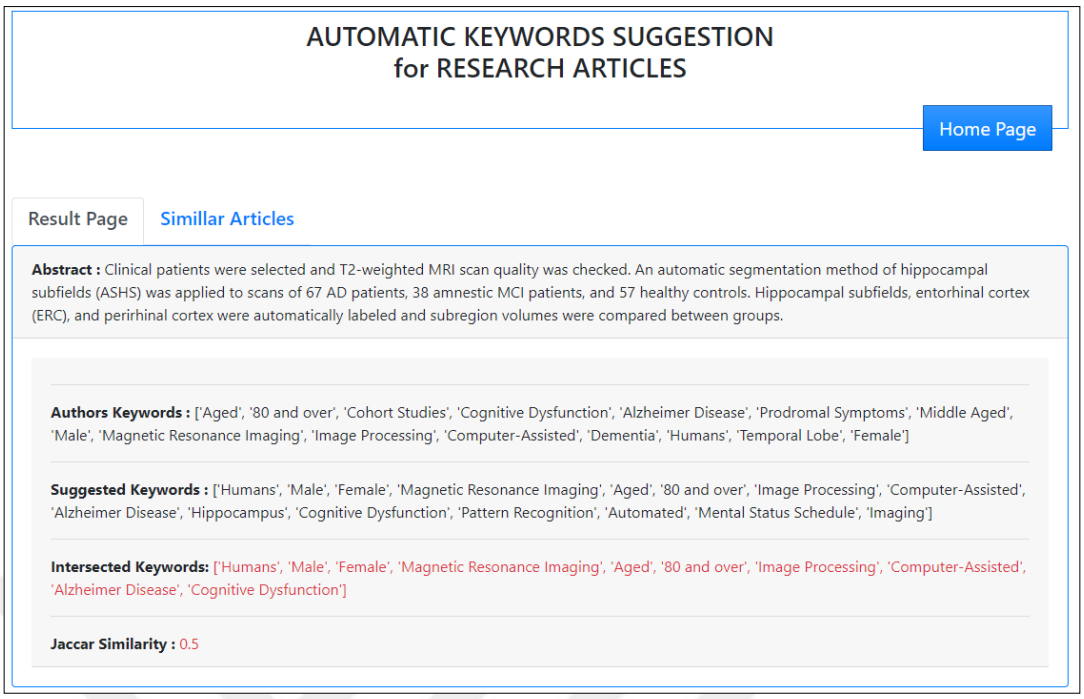


Figure 3.13 Result of *t*-NN algorithm

CHAPTER FOUR
EXPERIMENTATION AND RESULT

In our experiments we have used Elasticsearch as a distributed NoSQL database. Elasticsearch's mechanism is able to calculate similarities and return the most similar documents. Our second tool is Kibana, we can easily monitor the Elasticsearch using Kibana. Another tool is chart.js, it is a JavaScript library that allows us to draw different types of charts by using the HTML5 canvas element. Since it uses canvas, we have to include a Polyfilla to support older browsers. The one suggested by the author is Explorer Canvas, so we may want to stick with it (De Rosa, 2015). Finally, we used the flask web framework which is a powerful python framework used for user interface.

As a result, we have completed our application using these technologies and we have made many experiments. A few of the best results from our experiments are described in detail below.

4.1 Experiment *k*-NN

In this experience, we have conducted many experiments to find the optimal values of *k* and keyword number (*n*) variables in the *k*-NN method. In this experimentation, we basically considered 3 different situations. Firstly, we selected the value of *n* variable as the number of keywords of the author and observed the change of *k* value from 1 to 29. Secondly, this time we select the value of *n* variable as 11 and we examine the change in the value of *k* from 1 to 29. Finally, we selected the optimal value of the *k* variable from the results obtained in the first two cases. Then we observed the change of *n* value. As a result of these 3 experiments, we chose optimal *k* ne *n* values. We explained these 3 experiments in detail below.

4.1.1 Optimization of **k** *value for* **n** *is equal to 11*

In this experiment, we performed tests to find the best value of *k* in the *k*-NN approach. In these tests, we first selected the number of keywords (*n*) 11 and randomly selected 1000 articles from 10000 articles in the Elasticsearch test database. Then we tried the *k* value in order from 1 to 29. To put it in detail, firstly: *k* is 1 and *n* is 11, then we apply *k*-NN approach for 1000 papers. We explained in detail how *k*-NN method works in Chapter 4 *k*-NN method. This process continues until the value of *k* is 1,3,5,7… 29. As a result, the results of the evaluation metrics we obtained, that is, the figure of change of precision, recall, F-measure and other evaluation metrics are shown in Figure 4.1.

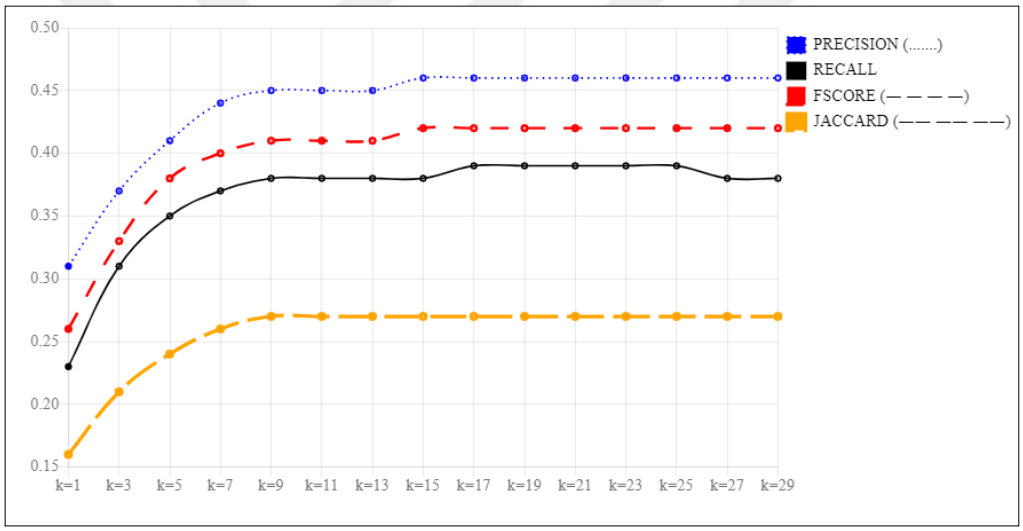


Figure 4.1 Dissimilarity of the *k* value when *n* is equal to 11

The articles in the Medline dataset have an average of 8-13 keywords. Therefore, in this test, we selected the number of keywords (*n* value) as 11. accordingly, the optimal value of the variable *k* in the test result is shown in Figure 4.1 as 21. Because after *k* is 21, the evaluation metrics remain constant.

4.1.2 Optimization of **k** *value for* **n** *is equal to the numbers of author's*

In this test, we selected the number of keywords, the value of *n* variables, as many as the number of keywords assigned by the author. Then, as in the previous test, we

tried the *k* value in this order from 1 to 29. As a result, the results of the evaluation metrics we obtained; The variation curves of the precision recall, F-measure and other evaluation metrics according to the *k* value are shown in Figure 4.2.

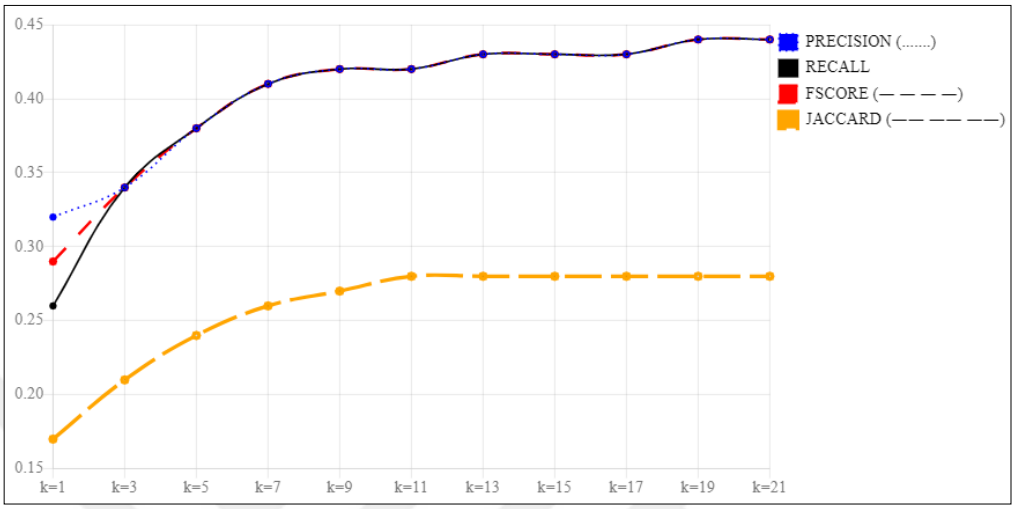


Figure 4.2 Variation of *k* while *n* is equal to the numbers of author's keywords

As a result of both tests we performed in *k*-NN approach, we found that the optimal value of the variable *k* is 21. Therefore, we selected the value of the variable *k* to be used in our next tests as 21.

4.1.3 Optimization of n value for k *is equal to 21*

In this test, we determined the optimal value of the number of keywords (*n* value). Firstly, both of the above tests showed us that the optimal value of *k* variables is 21 Therefore, in this test, we chose *k* as 21. Then we traced how the number of keywords (*n* value) changes from 3 to 29. As a result, the results of the evaluation metrics obtained according to the value of *n*; The variation curves of precision, recall, F-measure and other evaluation metrics are shown in Figure 4.3.

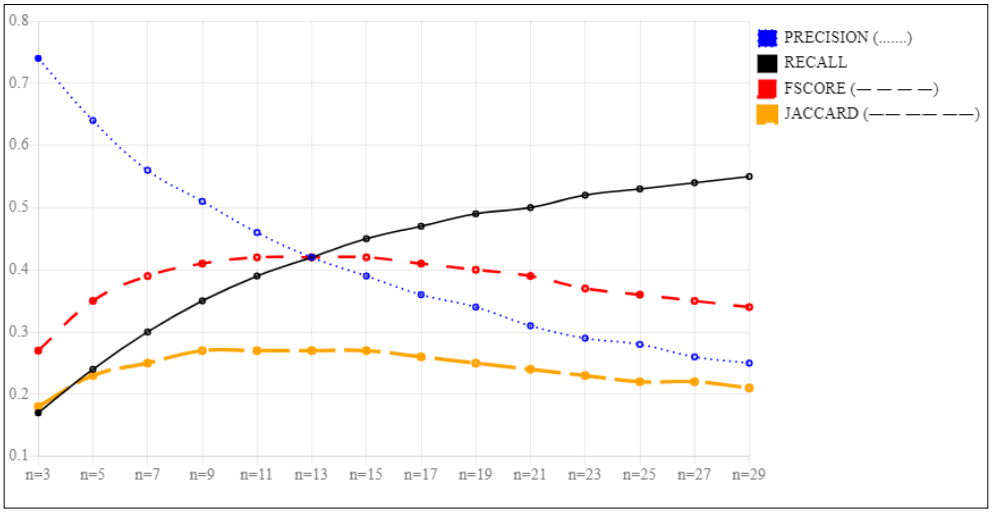


Figure 4.3 Variation of *n* while *k* is equal to 21

Figure 4.3 shows how the number of keywords changes when we run the program and accept the value of *k* as 21. When we look at Figure 4.3, we see that the best value of *n* is 11. As a result, in all tests we performed in *k*-NN method, we found the optimal value of *k* variable as 21 and the optimal value of the number of keywords (*n*) as 11. We used these values in our practice by referring these tests. That is, when our application is published, the user will only enter the abstract of the article. We included *k* and *n* values as 21 and 11, respectively.

4.2 Experiment *t*-NN

In this experiment, *t*-NN method is evaluated with two different approaches. *t*-NN-Frequency and *t*-NN-Score, and we have examined the results. The first approach *t*-NN-Frequency, we examined the optimal values of the threshold value (*t*) and the number of keywords (*n*) by calculating the frequencies of the keyword counts of articles whose similarity scores are above a certain threshold value. The second approach is *t*-NN-Score and in this approach, we have weighted the number of keywords of articles with a similarity score above a certain threshold value by multiplying it with similarity scores. In this section we have explained both approaches in detail.

4.2.1 t-*NN-Frequency*

In this experience, we have conducted several experiments to find the optimal values of the threshold value (*t*) and keyword number (*n*) in the *t*-NN-Frequency approach. In this approach, we basically considered 3 different situations. First of all, we selected the value of *n* variable as the number of keywords of the author and examined the change of the value of *t* from 20 to 200. Secondly, we selected the value of *n* variable as 11 and examined the change of *t* value from 20 to 200. Finally, we selected the optimal value of *t* variable from the results obtained in the first two cases. Then we examined the change of *n* value. In these cases, we selected the optimal *t* and *n* values from the results obtained according to the evaluation parameters and explained them in detail.

4.2.1.1 Optimization of t value for n *is equal to 11*

In this experiment, we performed tests to find the optimal threshold (*t* value) in the *t*-NN-Frequency approach. In these assessments, we first selected the number of keywords (*n* value) 11 and 1000 random articles from 10000 articles in the Elasticsearch test database. Then we tried starting *t* from 20 to 200 in sequence. In more detail, we first chose the *t* value 20 and the *n* value 11, and applied the *t*-NN-Frequency approach for each of the 1000 articles separately. Thus, we observed at how evaluation metrics changed. As a result, the evaluation metrics we obtained precision, recall, F-measure and the other evaluation metrics change are shown in Figure 4.4.

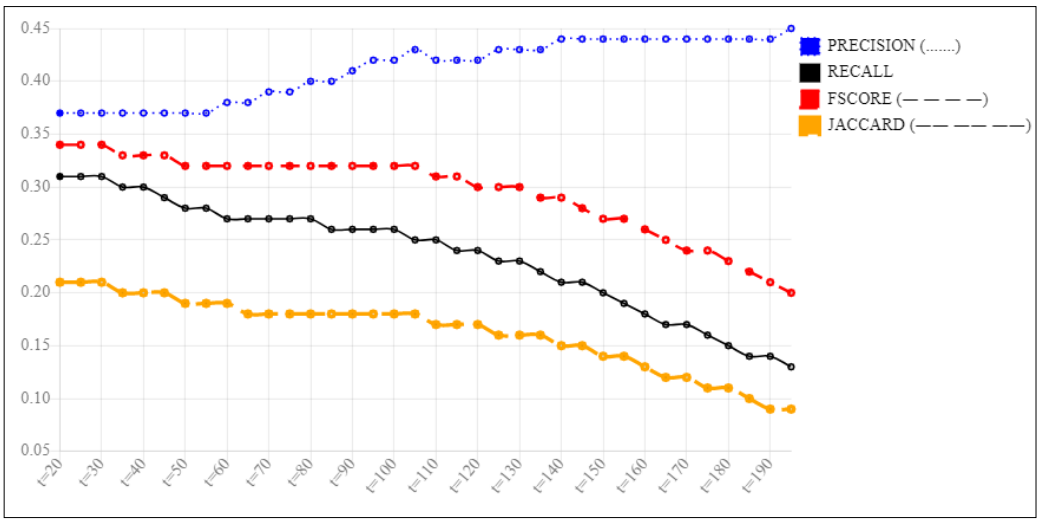


Figure 4.4 Dissimilarity of the *t* value when *n* is equal to 11

In this test, when we select the value of the number of keywords (*n*) as 11, it is seen that the optimal value of the threshold (*t*) variable is 80.

4.2.1.2 Optimization of t value for n *is equal to the numbers of author's*

In this experiment, we performed tests to find the optimal threshold (*t* value) in the *t*-NN approach. In these tests, we first selected 1000 articles randomly from 10000 articles in the Elasticsearch test database, the number of keywords (*n* value) the author assigned to the article. Then we tried starting *t* from 20 to 200 in sequence. In more detail, we first selected the value of *t* as 20 and *n* as the number of words assigned by the author and applied the *t*-NN approach for each of the 1000 articles separately. Thus, we examined how evaluation metrics have changed. As a result, the evaluation metrics we obtained, precision, recall, F-measure and the change of other evaluation metrics are shown in Figure 4.5.

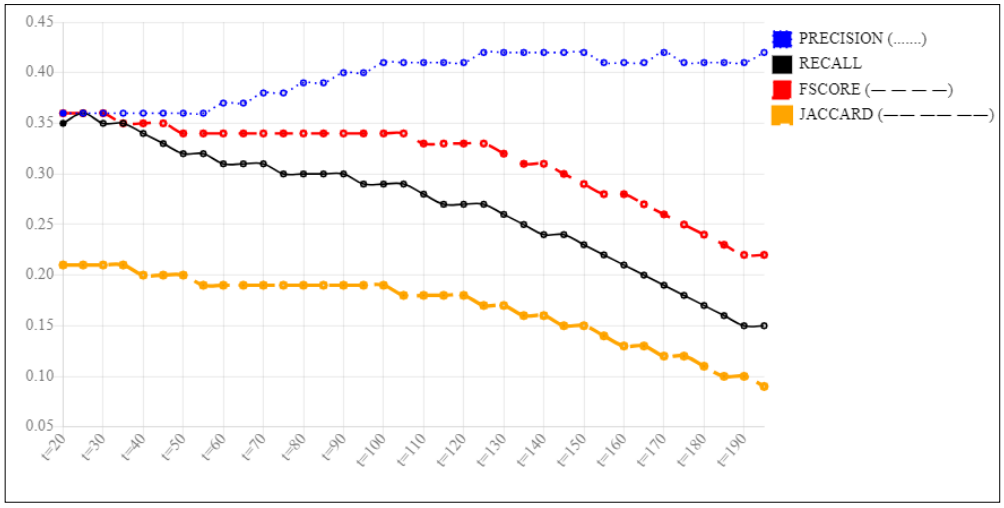


Figure 4.5 Variation of *t* while *n* is equal to the numbers of author's keywords

4.2.1.3 Optimization of n value for t is equal to 80

In this test, we determined the optimal value of the number of keywords (*n*). First, the above two experiments have shown us that *t* has an optimal value of 80. Therefore, in this test, we selected the value of *t* variable to be 80. Then, we observed how the number of keywords (*n*) changed from 3 to 29. As a result of the evaluation metrics precision, recall, F-measure and other evaluation parameters when we examine the optimal value of 15 is shown in Figure 4.6.

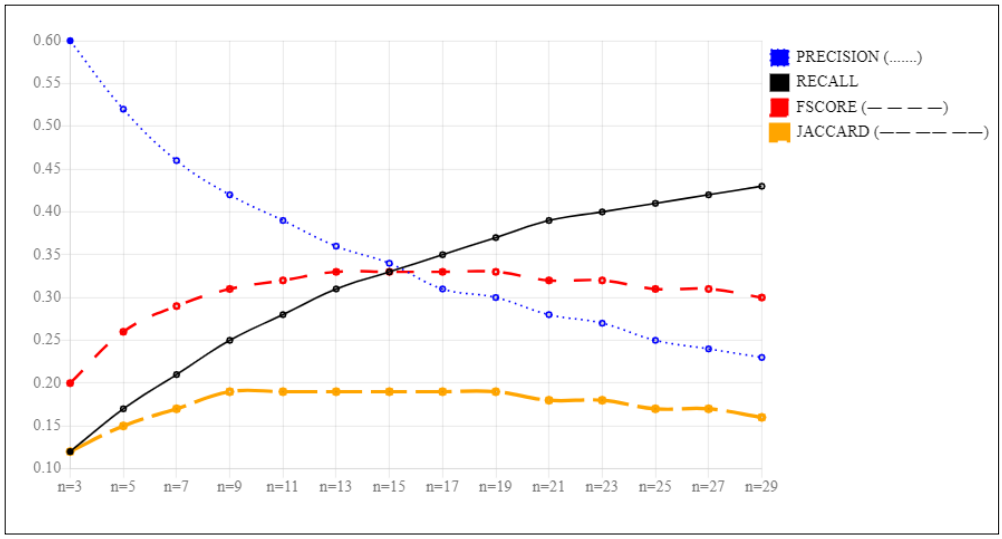


Figure 4.6 *t* is equal to 80 to see how *n* change

Figure 4.6 shows how the number of keywords changes when we run the program and assume *t* is 80. When we look at Figure 4.6, we see that the best value of *n* is 15. As a result, in all tests we performed in *t*-NN method, we found the optimal value of *t* variable to be 80 and the optimal value of the number of keywords (*n*) to be 15. We used these values in the application we developed. The user will only enter the abstract of the article. We used *t* and *n* values as 80 and 15 respectively.

4.2.2 t-*NN-Score*

In this experience, we have conducted several experiments to find the optimal values of the threshold value (*t*) and keyword number (*n*) in the *t*-NN-Score approach. In this approach, we basically considered 3 different situations. These 3 different situations are followed by the same processes with *t*-NN-Frequency. However, instead of finding a frequency, the keywords of the articles are weighted by multiplying the similarity score of the article. We observed *t* and *n* changes. In these cases, we selected the optimal *t* and *n* values from the results obtained according to the evaluation parameters and explained them in detail.

4.2.2.1 Optimization of t value for n is equal to 11

In this experiment, we performed tests to find the optimal threshold (*t* value) in the *t*-NN-Score approach. In these assessments, we first selected the number of keywords (*n* value) 11 and 1000 random articles from 10000 articles in the Elasticsearch test database. Then we tried starting *t* from 20 to 200 in sequence. In more detail, we first chose the *t* value 20 and the *n* value 11, and applied the *t*-NN-Score approach for each of the 1000 articles separately. Thus, we observed at how evaluation metrics changed. As a result, the evaluation metrics we obtained precision, recall, F-measure and the other evaluation metrics change are shown in Figure 4.7.

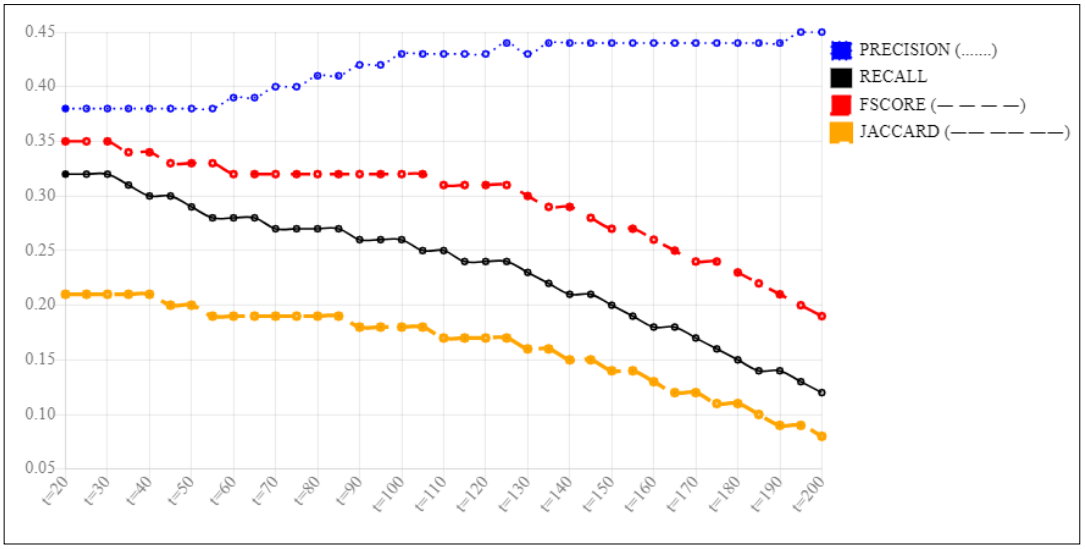


Figure 4.7 Dissimilarity of the *t* value when *n* is equal to 11 for *t*-NN-Score

In this test, when we select the value of the number of keywords (*n*) as 11, it is seen that the optimal value of the threshold (*t*) variable is 30.

4.2.2.2 Optimization of t *value for* n *is equal to the numbers of author's*

In this experiment, we performed tests to find the optimal threshold (*t* value) in the *t*-NN-Score approach. In these tests, we first selected 1000 articles randomly from 10000 articles in the Elasticsearch test database, the number of keywords (*n* value) the author assigned to the article. Then we tried starting *t* from 20 to 200 in sequence. In more detail, we first selected the value of *t* as 20 and *n* as the number of words assigned by the author and applied the *t*-NN-Score approach for each of the 1000 articles separately. Thus, we examined how evaluation metrics have changed. As a result, the evaluation metrics we obtained, precision, recall, F-measure and the change of other evaluation metrics are shown in Figure 4.8.

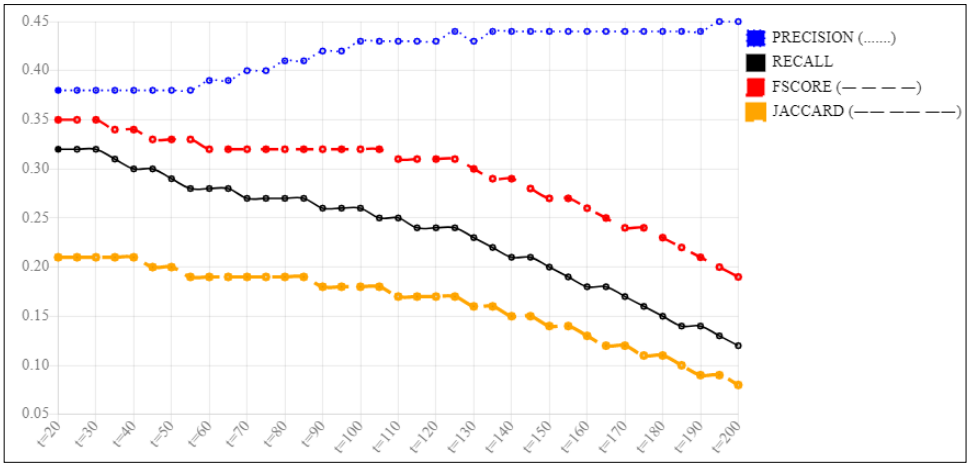


Figure 4.8 Variation of *t* while *n* is equal to the numbers of author's keywords

4.2.2.3 Optimization of n value for t *is equal to 30*

In this test, we determined the optimal value of the number of keywords (*n*). First, the above two experiments have shown us that *t* has an optimal value of 30. Therefore, in this test, we selected the value of *t* variable to be 30. Then, we observed how the number of keywords (*n*) changed from 3 to 29. As a result of the evaluation metrics precision, recall, F-measure and other evaluation parameters when we examine the optimal value of 15 is shown in Figure 4.9.

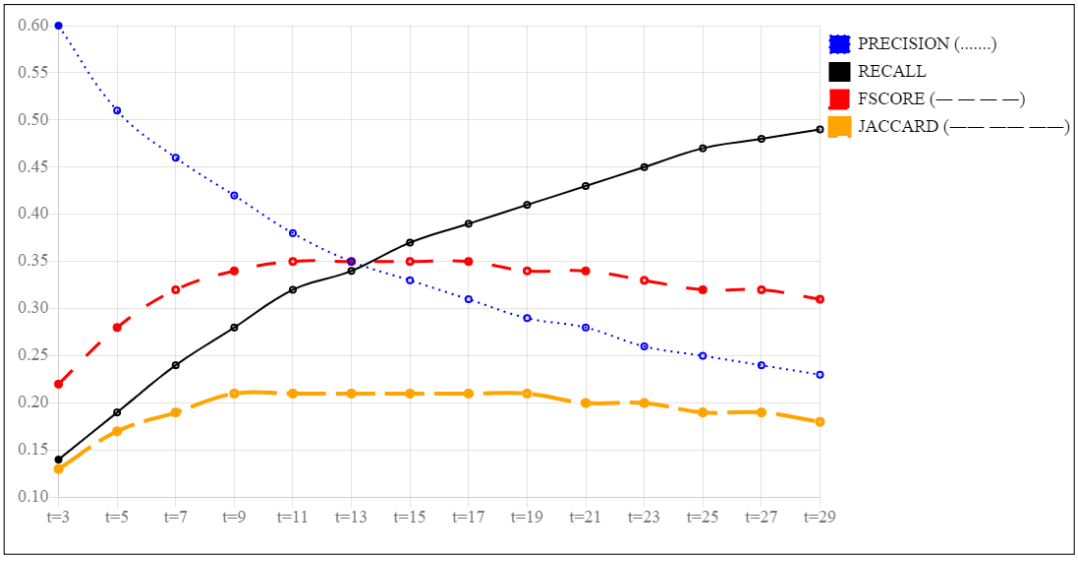


Figure 4.9 *t* is equal to 30 to see how *n* change

Figure 4.9 shows how the number of keywords changes when we run the program and assume *t* is 30. When we look at Figure 4.9, we see that the best value of *n* is 15. As a result, in all tests we performed in *t*-NN-Score method, we found the optimal value of *t* variable to be 30 and the optimal value of the number of keywords (*n*) to be 15. We used these values in the application we developed. The user will only enter the abstract of the article. We used *t* and *n* values as 30 and 15 respectively.

4.3 Evaluation

In Text Mining and Information Retrieval System there commonly used three evaluation metrics: Precision, Recall and F-measure (Goldsteiny, Kantrowitz, Mittal, & Carbonelly, 1999). These metrics are used to evaluate the quality proposed (Li & Zhao, 2016). In this study, we use precision, recall and F-measure rates to evaluate the results of our study. We also used Jaccard Similarity. Definitions of variables used in formulas: A (Author's Keywords) represents the set of keywords which are assigned by authors to articles in Medline dataset, S (Suggested Keywords) represents the set of keywords suggested by our algorithm, $|A \cap S|$ represents the number of the intersection of A and S. The commonly used formulas are as follow.

The concept of precision is generally indicated by the letter P and is calculated as the ratio of the correct results in the fetched information to the total of the fetched information.

$$P = \frac{|A \cap S|}{|S|} \tag{4.1}$$

The concept of recall is also generally indicated by the letter R and is calculated by the ratio of the correct results to the correct results.

$$R = \frac{|A \cap S|}{|A|} \tag{4.2}$$

In light of these definitions above, the F-measure is the harmonic mean of these values.

$$F - measure = \frac{PxRx2}{P+R} \tag{4.3}$$

The J general represents Jaccard Similarity and it is a statistic term used for measurement the similarity of sample sets.

$$Jaccard - Similarity = \frac{|A \cap S|}{|A \cup S|} \tag{4.4}$$

As a result, in our experiments for the models we developed, the best evaluation metrics of k and t variables, depending on the change in the number of keywords, are shown in Table 4.1, Table 4.2 and Table 4.3 respectively for each model.

Table 4.1 The best evaluation metrics results for variable *k*-NN

n	***k***	**Precision**	**Recall**	**F-measure**	**Jaccard Similarity**
11	21	0.46	0.39	0.42	0.27

Table 4.2 The best evaluation metrics results for variable *t*-NN-frequency

n	***t***	**Precision**	**Recall**	**F-measure**	**Jaccard Similarity**
15	80	0.36	0.36	0.36	0.19

Table 4.3 The best evaluation metrics results for variable *t*-NN-score

n	***t***	**Precision**	**Recall**	**F-measure**	**Jaccard Similarity**
13	30	0.36	0.35	0.36	0.20

CHAPTER FIVE
CONCLUSION AND FUTURE WORKS

As we all know, assigning keywords to research articles is a very important process. These keywords should properly describe the articles. Doing this manually is difficult and may cause the article to be misidentified. Before developing this system, we have examined previous studies in that fields. We have seen that these studies are generally called keyword extraction and are based on two different approaches. Some of them are based on collections. These approaches usually calculate the frequencies of words in a collection, or perform keyword extraction by calculating the frequency of cooccurrence words. On the other hand, the rest of the studies, calculate the frequencies of the words on a single document, we have seen the process of keyword extraction.

In our study, we designed and developed an automated keyword suggestion system for research articles. We used collection-based techniques to develop this system. In the system we developed, we used the Medline data set in PubMed system as a collection. However, we used two different collection-based methods based on the use of information retrieval systems, different from collection-based keyword extractions. First, the proposed keyword suggestion system considers the abstract of the research article as a query for the information retrieval system. The information retrieval system then returns a list of articles based on the ordered similarity score of the given query. Next, we selected an article set from this list using two different methods: *k*-NN and *t*-NN. These two methods represent the first *k* articles and articles whose their similarity score is greater than the threshold value of *t*.

To evaluate the proposed system, we conducted a series of experiments using a thousand randomly selected articles from Medline corpus and compared the system results with the authors' keywords. Our results show that our system offers approximately 42% matching keywords in terms of F-measure.

The results of the experiments showed that the proposed approach suggested 50% matched keywords with the keywords given by the authors. However, we can not assume that the mismatched keywords are irrelevant. A potential future study may be an evaluation of the external interest of keywords that are not in the authors' original keyword list.

As a result, thanks to the system we have proposed, researchers will be able to assign appropriate keywords as keywords for their articles. In addition, our tests showed that the system we developed was valid for suggesting keywords to a research article.

REFERENCES

Aggarwal, C., & Zhai, C. (2012). A Survey of Text Clustering Algorithms. C. Aggarwal, & C. Zhai içinde, *Mining Text Data* (77-128). New York : Springer.

Akdal, B., Keskin, Ç., Gül Z., Ekinci, E. E., & Kardas, G. (2018). Model - driven Query Generation for Elasticsearch. *Computer Science and Information Systems*, 853-862.

Andrade, M., & Valencia, A. (1998). Automatic extraction of keywords from scientific text: Application to the knowledge domain of protein families. *Bioinformatics*, 600-607.

Balamaci, S. (2016). *Java app monitoring with ELK - Part II- ElasticSearch*. Retrived March 21, 2016, from https://balamaci.ro/java-app-monitoring-with-elk-elastic-search/

Bansal, S. (2016). *Beginners Guide to Topic Modeling in Python*. Retrieved Ağustos 24, 2016, from https://www.analyticsvidhya.com/blog/2016/08/beginners-guide-to-topic-modeling-in-python/

Cai, Y., & Sun, J. T. (2009). Text Mining. *Springer*, 155-159.

De Rosa, A. (2015). *Creating beautiful charts with Chart.js*. Retrieved January 07, 2015, from https://www.sitepoint.com/creating-beautiful-charts-chart-js/

Fairhead , H. (2017). *History of Computer Languages - The Classical Decade, 1950s*. Retrieved November 16, 2017, from https://www.i-programmer.info/history/57-computer-languages/471-the-classical-decade.html

Garbade, M. J. (2018). *A quick introduction to text summarization in machine learning*. Retrieved September 19, 2018, from https://towardsdatascience.com/a-quick-introduction-to-text-summarization-in-machine-learning-3d27ccf18a9f

Goldsteiny, J., Kantrowitz , M., Mittal , V., & Carbonelly, J. (1999). In text mining and information retrieval system there commonly used three evaluation metrics. *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (p. 121-128). Berkeley, CA, USA: 22nd SIGIR 1999.

Gupta, V., & Lehal, G. S. (2009). A survey of text mining techniques and applications. *Journal Of Emerging Technologies In Web Intelligence*, 60-76.

Gutwin, C., Paynter, G., Witten, I., Nevill-Manning, C., & Frank, E. (1999). Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, 81-104.

Hafez, P. (2017). *How to succeed in quant investing with big data analytics*. Retrieved June 27, 2017, from https://www.ravenpack.com/blog/quant-investing-big-data-analytics/

Hunter, L., & Cohen, K. B. (2006). Biomedical language processing: what's beyond PubMed?. *Molecular Cell* , 589-594.

Hunt-Walker, N. (2018,). *An introduction to the Flask Python web app framework*. Retrieved April 02, 2018, from https://palletsprojects.com/p/flask/

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Springer*, 137-142.

Jones, S., & Paynter, G. W. (2002). Automatic extraction of document keyphrases for use in digital libraries: Evaluation and applications. *Journal of the American Society for Information Science and Technology*, 653-677.

Kemp, S. (2019). *Global Social Media Users Pass 3.5 Billion*. Retrieved July 17, 2019, from https://wearesocial.com/blog/2019/07/global-social-media-users-pass-3-5-billion

Leaman, R., & lu, Z. (2004). Accessing biomedical literature in the current Information landscape. *Methods in Molecular Biology*, 11-31.

Li, W., & Zhao, J. (2016). TextRank Algorithm by exploiting wikipedia for short text keywords extraction. *3rd International Conference on Information Science and Control Engineering* (683-686). Beijing, China.: IEEE.

Li, Z., Zhou, D., Juan, Y.-F., & Han, J. (2010). Keyword extraction for social snippets. *Proceedings of the 19th International Conference on World Wide Web* (1143-1144). North Carolina: WWW 2010.

Liu, Z., Huang, W., Zheng, Y., & Sun, M. (2010). Automatic keyphrase extraction via topic decomposition. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing* (366-376). Cambridge: Association for Computational Linguistics.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Revelopment*, 159-165.

Maisam. (2019). *Python programming language*. Retrieved October 18, 2019, from https://socialmedianewstime.com/2019/10/18/python-programming-language/

Matsuo, Y., & Ishizuka, M. (2003). Keyword extraction from a single document using word co-occurrence statistical information. *International Journal of Artificial Intelligence Tools*, 392-396.

Mehta, S. (2014). *Elasticsearch tutorial - Elasticsearch storage architecture : analysis and inverted indexes*. Retrieved June 01, 2014, from http://siddhumehta.blogspot.com/2014/06/elasticsearch-tutorialinvertedindex.html

Mohapatra, S. (2015). *ElasticSearch basics*. Retrieved February 04, 2015, from https://www.slideshare.net/pinkusatya/elasticsearch-basics

Newman, D., Chemudugunta, C., Smyth, P., & Steyvers, M. (2006). Analyzing entities and topics in news articles using statistical topic models. *International Conference on Intelligence and Security Informatics* (93-104). San Diego, CA, USA: Springer.

Oliphant, T. E. (2007). Python for scientific computing. *Computing in Science & Engineering*, 10-20.

Patrick. (2018). *Testing a Flask Application using pytest*. Retrived May 02, 2018, from http://www.patricksoftwareblog.com/category/flask-tutorial/

PostgreSQL About. (2019). Retrieved September 15, 2019, from https://www.postgresql.org/about/

PostgreSQL Tutorial. (2019). Retrieved November 12, 2019, from http://www.postgresqltutorial.com/

Rajeshkumar. (2018). *Understanding Elasticsearch Keywords and Terminology*. Retrieved September 27, 2018, from https://www.devopsschool.com/blog/understanding-elasticsearch-keywords-and-terminology/

Rak, R., Kurgan, L. A., & Reformat, M. (2007). Multilabel associative classification categorization of MEDLINE articles into MeSH keywords. *IEEE Engineering in Medicine and Biology Magazine*, 47-55.

Rose, S., Engel, D., & Cramer, N. (2010). Automatic keyword extraction from individual documents. M. W. Berry, & J. Kogan içinde, *Text Mining Applications and Theory* (s. 3-19). University of Maryland Baltimore County, USA: A John Wiley and Sons, Ltd., Publication.

Rouse, M. (2019). *Analytics technologies lend enterprise content management a hand*. Retrieved November 07, 2019, from https://searchbusinessanalytics.techtarget.com/definition/text-mining

Sharma, S. (2017). *KISS: Keep It Short and Simple*. Retrieved January 06, 2017, from http://sidhant.io/kiss-keep-it-short-and-simple

Truong, E. V. (2018). *Text Analysis*. Retrieved October 30, 2018, from http://rpubs.com/-bnevt0/AT336106

What is Elasticsearch. (2019). Retrieved October 18, 2019, from https://www.elastic.co/what-is/elasticsearch

Zhang, C., Wang, H., Liu, Y., Wu, D., Liao, Y., & Wang, B. (2008). Automatic keyword extraction from documents using conditional. *Journal of Computational Information Systems4:*, 1169-1180.

Zhang, K., Xu, H., Tang, J., & Li, J. (2006). Keyword extraction using support vector machine. *Advances in Web-Age Information Management* (85-96). Hong Kong, China: Springer.