universität koblenz
weiter:denken

Fachbereich 4: Web and Data Science

Fraunhofer FKIE

Informationstechnik für Führungssysteme (ITF)

Sub-topic modeling and ranking analysis in document retrieval systems

Masterarbeit

zur Erlangung des Grades Master of Science (M.Sc.)
im Studiengang Web and Data Science

vorgelegt von

Gadiyaram, Sri Sai Praveen

Erstgutachter:	Prof. Dr. Jan Jürjens Institute for Software Technology
Zweitgutachterin:	MSc. Katharina Großer Institute for Software Technology

Koblenz, im Mai 2023

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

	Ja	Nein
Mit der Einstellung der Arbeit in die Bibliothek bin ich einverstanden.	☐	☐

.....

(Ort, Datum) (Unterschrift)

Kurzfassung

Die Suche nach hochrelevanten Dokumenten in den Top-Ergebnissen für eine bestimmte Benutzeranfrage ist eine der schwierigsten Aufgaben im Information Retrieval (IR). Diese Herausforderung wird noch verstärkt, wenn der Benutzer eine bestimmte Absicht hat und der Suchanfrage der Kontext seiner Absicht fehlt. So kann beispielsweise die Benutzeranfrage *"Robotik"* Dokumente aus vielen Bereichen wie Fertigung, Landwirtschaft, Militär usw. abrufen. Eine einfache Schlüsselwortsuche kann den Benutzer mit vielen falsch-positiven Ergebnissen überfordern, wenn der Benutzer die Innovationsdokumente nur in Bezug auf einen bestimmten Bereich, wie z.B. *"Militär"*, untersuchen möchte. Um die Absicht des Benutzers und den fehlenden Kontext in der Benutzerabfrage zu erfüllen, wird ein neuartiger Dokumentenmodellierungsansatz für die Abfrage vorgeschlagen, um hochgradig kohärente abfragespezifische Kontexte (Unterthemen) aus den am häufigsten abgefragten Dokumenten zu extrahieren, was dem Benutzer bei der Eingrenzung des Suchraums immens hilft. Darüber hinaus wird der vorgeschlagene Ansatz anhand von Präzisions- und Umfrageanalysen bewertet.

Abstract

Retrieving highly relevant documents in the top results for a given user query is one of the challenging tasks in Information Retrieval (IR). This challenge is amplified when the user has a specific intention, and the search query lacks the context of their intention. For example, the user query *"Robotics"* can retrieve documents related to many domains such as manufacturing, agriculture, military, etc. A simple keyword search can overwhelm the user with many false positives when the user wants to explore the innovation documents only related to a specific domain, such as *"Military"*. To fulfill the user intent and missing context in the user query, a novel document modeling approach for retrieval is proposed to extract highly coherent query-specific contexts (sub-topics) from the top retrieved documents, which helps the user immensely to narrow down the search space. Furthermore, the proposed approach will be evaluated using precision and survey analysis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Information Seeking (IS) is a process of fulfilling the information need of the individuals by obtaining appropriate information [60]. IS has become a crucial part in today's life and people interact daily with various web platforms, such as Google[1], Youtube[2], ChatGPT[3], etc., to obtain information from various sources according to their needs or requirements. The information source can be of different types namely text, audio, video, etc., and the interacting human query can also be different such as simple text input, image input, audio input etc. Some people just want to gather insights to be informed with the latest news, social activities, entertainment, technology etc., and others wants to play an active seeker role by seeking precise information. This precise information can further used to take crucial decisions. Therefore, irrespective of being an active or a passive information seeker, the individuals find themselves in lack of information and would like to seek information from external sources.

Due to the lack of information, users interpret the information gap as a problem and tries to manage or resolve this problem by interacting with information sources [9]. These user activities are characterized as information seeking behaviors [9] and can be referred as information search behaviors when specifically seeking the information through a web search. In this master thesis, we consider the behaviors related to text interacting information seeking, as the majority of information seeking happens in the form of text search. Moreover, the thesis is explicitly targeted towards the precise and exploratory information seeking behaviors in order to understand and help the users in their information seeking process effectively.

1.1 Motivation

Regardless of the search platform, the quality of the search results depends on the formulation of the information need by the information seekers and the same is described by the researchers as ***Quality-in quality-out*** *principle: "A query that more accurately reflects the user's information need will produce better results"* [20]. This principle not only imply to the syntactic matching techniques but also to the semantic matching algorithms. Therefore, query formulation plays a crucial role in information seeking process, but the quality of the formulated query is hard to determine and is influenced by several factors such as the user's knowledge of information need, search experience, system experience (user interface), etc [42].

[1] https://www.google.com/
[2] https://www.youtube.com/
[3] https://chat.openai.com/

Query Completion (QC) is a popular technique to help the users to better formulate their information need. QC leverages the actively indexed data on the web and query logs generated by millions of users and is observed in almost every modern browser with the facilities such as real-time auto-completion, spelling correction, etc [7, 28]. Researchers in [42] have shown that the information seekers are more involved in their search when using QC and led to higher user satisfaction.

Once the query is formulated, there are some possible reasons for the poor web-search results (low relevancy) [6].

- **Cross-domain queries:** User queries related multiple domains or topics. This results in high false positives and can only be resolved with a well-formed query mapped to a certain topic according to the interest of information seeker.
- **Short queries:** If the query length is too short, then the user's information need is not well expressed. The nature of providing short queries can also be seen as a habit. A recent analysis to understand the user search queries on 306 million keywords used in google search showed that user queries were comprised of a relatively small number of keywords, and the mean keyword length is 1.9 words and 8.5 characters [23].
- **Poor information need:** User is not very sure on what he/she is actually looking for until he/she sees the search results. The user might want to do exploratory search on a certain topic and learn from the results.
- **Poor query formulation:** User is sure on what he or she wants but does not formulate the search query appropriately.

Length of the documents in the information source can also be a major factor influencing the search results. Smaller text documents used in many NLP projects such as IMDB reviews, tweets, requirements etc., are generally mapped to a single class or a topic. On the other hand, longer documents such as news-articles, blogs, e-books, research papers etc., contain several topics in it and can not be logically mapped to a single topic. Reading such long documents in top results can lead to poor user satisfaction when their relevancy is very low (due to the above mentioned factors).

To tackle this problem user should frame a well written query specifying his or her intent. For example, the query *"What are the technological advancements in Robotics related to Unmanned Weapon Systems?"* shows a well-formulated query clearly specifying the information need. Demanding users to provide such a query every time can be very difficult specially in the case of an exploratory task where the users are not sure what they are actually looking for. Furthermore, this can also lead to poor search experience because only certain type of queries are allowed to the users.

Instead of asking the users to formulate a proper query every time, they can be guided using the interactive relevance paths generated from a simple user query. These relevance paths are document representations which are extracted from the top retrieved documents for the original query. This approach assumes that the top documents are relevant to the query and is referred as *Content-driven information seeking* [42]. The outcome of one research suggests that content-driven relevance paths are beneficial in case of exploratory searches and not very useful in known-item searches [42]. Specially in the case of seeking information from very long text documents, these content-driven interfaces can help the user to narrow down the search space and reach the relevant documents easily. An unsupervised context extraction technique from the top-100 documents is proposed in this master thesis to help the users with highly heterogeneous topics.

1.2 Research questions (RQ)

Information Extraction (IE) and Information Retrieval (IR) are two significant techniques in the research to satisfy the user's information need. IE is a technique for automatically extracting or mining pre-specified information from the data. On the other hand, IR is a technique for extracting relevant documents for a given user query. In [28], the authors characterize the complementary nature of these two techniques and the potential of building practical and powerful tools by combining them. For example, IE can be used to better formulate the information need of the user and IR can be used to retrieve high quality search results.

In this master thesis, an interactive system with the combination of these two techniques is proposed and tested to fulfill the user's information need. An unsupervised soft clustering approach is designed to model documents (from multiple languages) as a mixture of sub-topics, which are extracted using the deep inherent information from keywords. The testing of this approach is divided into 3 phases not only evaluating the clustering with some evaluation metrics but also with a user satisfaction survey. Below are the research questions that address the evaluation of the proposed approach mentioned above.

RQ1: ***How effective is the sub-topic modeling approach in creating distinctive clusters from the news articles?***

Any clustering algorithm can generate topics (well-formed clusters) from a collection of documents, but these topics are highly significant to the users only if they are very distinctive to each other. Heterogeneous clusters help the user to take a quick decision and retrieve documents specifically related to the query and the sub-topic cluster. To generate such highly distinctive clusters, a candidate keyword selection approach is introduced and evaluated. This research question aims to test the effectiveness of this keyword selection technique with normal clustering. Both intrinsic and extrinsic clustering evaluation techniques, as well as a survey, are chosen to evaluate the clustering output. The parameters of the clustering pipeline are tuned using the evaluation metrics and also with manual observation.

RQ2: ***Which IR system retrieves more relevant documents for a user query and a sub-topic?***

When a user chooses a particular sub-topic cluster, it is assumed that the retrieval results related to the query and the sub-topic are retrieved and shown to the user. Two different IR systems are proposed in this master thesis to retrieve documents relevant to both the given user query and the chosen sub-topic. This research question targets comparing these two retrieval systems and determining the better one. A survey will be performed, and the collected data will be analyzed to answer the *RQ2*. Survey participants are allowed to read the retrieved documents from the two IR systems and then answer five questions. A statistical t-test is conducted on the collected survey data to determine whether the two IR system results are same or different. More details about the IR systems and survey questions are shared further in the report.

RQ3: ***What is the effect of sub-topic ranking in finding the positive documents from the candidate pool?***

Showing only the documents related to a specific sub-topic can restrict the user from reading the original retrieved results for the given query. This research question addresses the impact of the sub-topic clustering output to find the positive documents against the baseline approach and is evaluated through an exploratory precision analysis. This research questions also aims at the extraction inherent insights from the clusters and its rankings. For example: *Does any specific ranking of sub-topic clusters produces better precision than baseline?*. A popular IR evaluation metric namely Mean Average Precision (MAP) is used to perform the ranking analysis and answer this research question.

1.3 Structure of the thesis

This master thesis is divided further into six sections namely **Background and Fundamentals, Technical background, Related work, Methodology, Experiment setup, Experiment results,** and **Conclusion**.

Chapter 2

Background and Fundamentals

2.1 Fraunhofer FKIE

Fraunhofer FKIE (Fraunhofer-Institut für Kommunikation, Informationsverarbeitung und Ergonomie) is a research institute for providing innovative solutions in information and communications technology, and their main focus is on developing effective and efficient human-machine systems[1]. The users at FKIE are specially interested in reading news articles related to innovation and breakthroughs in *Technology and Military*. The below image, Figure 2.1, shows an example of areas of interest to the FKIE users, and this list is not bounded and can include more domains.

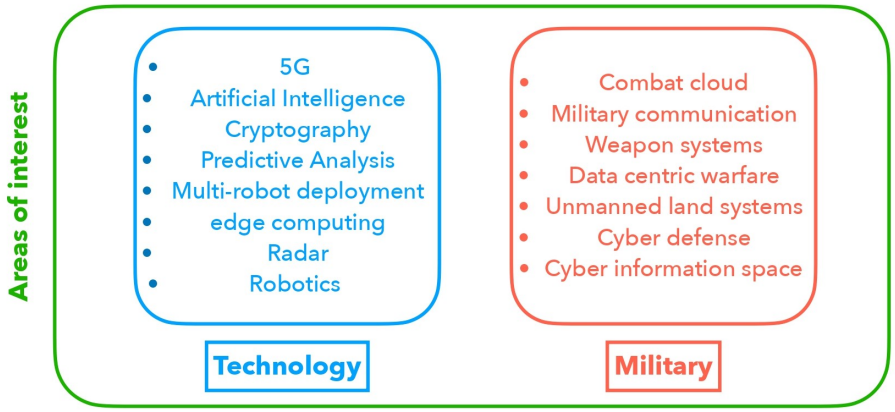


Figure 2.1: Areas of interest for the users at FKIE

2.2 Technical details

2.2.1 Abbreviations

1. **URL:** A URL is a short form for *Uniform Resource Locator* and is used to locate resources uniquely on the Internet [12]. Any resource on the Internet can be accessed with a unique URL. For example, the URL <https://www.linux.org/> represents a resource on the Internet.

[1] `https://www.fkie.fraunhofer.de/en/about-fkie.html`

2. **HTML:** HTML stands for *HyperText Markup Language* and is a markup language for representing documents on the World Wide Web (WWW) and links to other documents or information sources such as images, video, audio, etc [32].

3. **CSS:** Bootstrap

4. **JSON:**

5. **CSV:**

6. **UUID:**

2.2.2 Machine Learning (ML)

1. **Ground-truth:** Ground-truth labels are the information that is more accurate, relevant, and true than the knowledge of the system we are testing [17]. This information is critical to evaluate and compare different systems.

2. **Supervised learning:** Supervised learning algorithms are ML approaches that use labeled data [21] for training the algorithm parameters using specific criteria or a loss function. *Classification* is a supervised technique to learn patterns from the labeled data and classify the unseen data automatically into several classes.

3. **Unsupervised learning:** Clustering is an example of these algorithms, and similar data points are clustered into groups according to the features in the data [40]. These groups are called *Clusters*. Document clustering is a technique to group documents into topics without ground-truth information [22].

4. **Soft clustering:** In *Soft clustering*, the data points are assigned to one or more clusters by the clustering algorithm [22]. This depends mainly on the structure of the data, and especially in news articles, documents are assigned to multiple topics rather than one.

5. **Support Vector Machine (SVM):** SVM is a supervised learning algorithm used for classification and regression. Using the labeled data information, *SVM* selects a maximum margin separating hyperplane(a decision boundary) between the data points. This hyperplane is used later for classifying new data points [50].

6. **Tensorflow Hub:** Tensorflow Hub[2] is a repository of pre-trained ML models from *Tensorflow*[3]. Tensorflow is an open-source platform for ML that provides an ecosystem of tools and libraries and allows developers to build and deploy ML-powered apps and researchers to push state-of-the-art models [24].

2.2.3 Natural Language Processing (NLP)

1. **Token:** In NLP, a token is a word or basic entity in a text document, and Tokenization is the process of splitting a text document into tokens [64]. Document token length is calculated as the number of tokens present in a document.

2. **Noun chunks:** Noun chunks or phrases are the nouns and all the words that depend on these nouns [49]. Consider the sentence, *"Army project may improve military communications by boosting 5G technology"* [1]. The possible noun chunks extracted from this sentence are *"Army project"*, *"Military communications"*, *"5G technology"*.

[2] `https://www.tensorflow.org/hub`
[3] `https://www.tensorflow.org/`

3. **Keywords:** Keywords are the noun chunks that are highly meaningful in a text document and can best describe or summarize a document [8]. An unsupervised multi-lingual keyword extraction approach is used in the proposed approach to extract the most significant keywords from each news article.

4. **Stop-words:**

5. **Lemmatization:**

6. **Lexical matching:** Lexical or syntactic matching is a technique to assign a relevance score between two text data (strings) based on the terms present in the data. This matching technique is not optimal for retrieval, as it does not consider the meaning of the query [35].

7. **Fuzzy-matching:**

8. **Levenshtein distance:**

9. **Semantic matching:** Semantic matching assigns a relevance score between the two text data by considering the semantic information (meaning of the terms).

10. **Text embeddings:** The distributed vector representation of a text in the semantic space is generally referred as text embeddings. These embeddings are generally described in the research as word or sentence embeddings referring to the text being either a word or a sentence repectively. These embeddings can also be generated with short phrases and noun chunks [18, 65].

11. **fasttext:**

2.2.4 Information Retrieval (IR)

1. **Document retrieval system:** IR system specially developed to retrieve the document or text data for a given user query is generally referred to as a Document retrieval system.

2. **BM-25:** BM-25, Best Match 25, is a ranking function based on a probabilistic relevance framework that ranks documents based on the query terms occurring in each document [4]. BM-25 ranking is a lexical or syntactic matching approach and does not consider word semantics.

3. **Semantic search:** Unlike syntactic matching or calculating term frequencies, Semantic search engines try to understand the meaning of the search query and retrieve the matching documents close to the query in the semantic space [26].

2.2.5 Data storage

1. **Document index:** Document indexing or compression is a technique to store documents in an optimized way on the disk for efficient retrieval. This stored data on the disk is now referred to as *Document Index* [70].

2. **Inverted index:** The *Inverted index* is a data structure that contains every word in the corpus and the separate list of documents where the word occurs [70].

3. **Elastic search:**

4. **Semantic search index:** The *Semantic search index* stores the distributed embedding vectors of the documents on the disk and uses them later for retrieval.

5. **FAISS:**

6. **SQLite DB:** SQLite is a lightweight serverless, self-contained, transactional database engine [13]. In this master thesis, labeled data are stored in *SQLite DB* using a library *sqlite3*[4].

7. **Redis DB:**

2.2.6 Evaluation

1. **t-test:** sfs

2. **Intrinsic evaluation:** In case of no labeled data or ground-truth, the clustering output is evaluated through the methods considering only the inherent representation of clustered data [22]. These methods of evaluation are referred to as *Intrinsic evaluation*.

3. **Extrinsic evaluation:** In *Extrinsic evaluation*, the clustering output is evaluated using the external knowledge such as ground-truth or the relevance judgments [22].

4. **Silhouette index:** Irrespective of the clustering algorithm, the output is more distinctive when the distance between the data points within the cluster is minimum and the distance between the clusters is maximum. Silhouette index [59] is an intrinsic clustering evaluation measure and is calculated by using the intra-cluster and inter-cluster distances for each sample.

5. **Precision:** In IR system evaluation, Precision is defined as the ratio of retrieved documents that are relevant to all the retrieved documents [71]. This measure can be used to compare different IR systems and be calculated at different retrieved indices. For example, $P@5, P@10, P@15$ measures precision scores at retrieved indices $5, 10, 15$ respectively.

6. **Cosine similarity:** Cosine similarity is a metric to measure the degree of similarity between two vectors [36]. In the case of IR systems, the similarity is calculated between the user query and document sentence embeddings, and can be further used to rank the documents.

2.2.7 Technology fundamentals

1. **Python:**

2. **Javascript:**

3. **Docker:**

4. **Fastapi:**

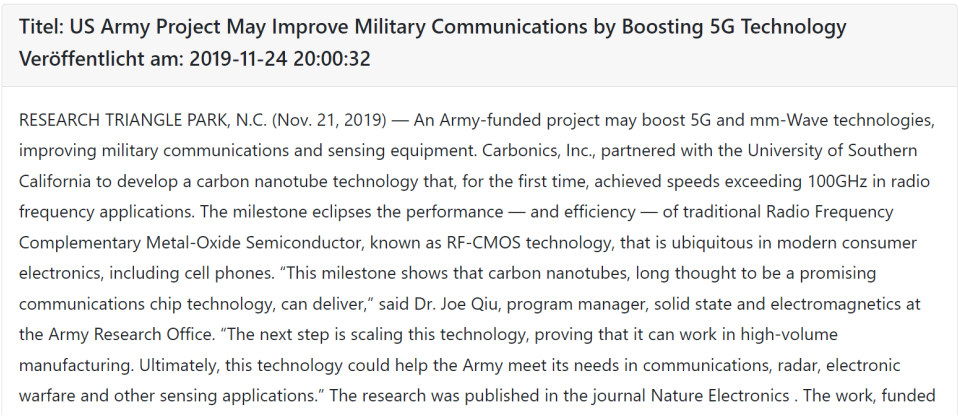
5. **Scrapy:**

6. **FuzzyWuzzy:**

[4] `https://docs.python.org/3/library/sqlite3.html`

2.2.8 Keywords specific to this master thesis

1. **News article:** A news article is a text document published by a news website. An example of a news article (this is only a part of the original article) is shown in Figure 2.2.


Titel: US Army Project May Improve Military Communications by Boosting 5G Technology
Veröffentlicht am: 2019-11-24 20:00:32

RESEARCH TRIANGLE PARK, N.C. (Nov. 21, 2019) — An Army-funded project may boost 5G and mm-Wave technologies, improving military communications and sensing equipment. Carbonics, Inc., partnered with the University of Southern California to develop a carbon nanotube technology that, for the first time, achieved speeds exceeding 100GHz in radio frequency applications. The milestone eclipses the performance — and efficiency — of traditional Radio Frequency Complementary Metal-Oxide Semiconductor, known as RF-CMOS technology, that is ubiquitous in modern consumer electronics, including cell phones. "This milestone shows that carbon nanotubes, long thought to be a promising communications chip technology, can deliver," said Dr. Joe Qiu, program manager, solid state and electromagnetics at the Army Research Office. "The next step is scaling this technology, proving that it can work in high-volume manufacturing. Ultimately, this technology could help the Army meet its needs in communications, radar, electronic warfare and other sensing applications." The research was published in the journal Nature Electronics . The work, funded

Figure 2.2: A sample news article from the document database [1]

2. **Web scraping:** Web scraping is a technique to automatically extraction of data from websites [33]. In the case of text data, most approaches download the structured HTML webpages and extract needed information. In this master thesis, news articles from different websites are scraped.

3. **Candidate pool:** A candidate or retrieval pool is a set of documents from lexical and semantic matching results for a given user query. These documents are very diverse, contain keywords present in the query (or semantically similar), and are further used for clustering.

4. **Sub-topic:** Sub-topics are second-level representations of a document. Generally, news articles are long text documents and can not be represented logically with a single topic or keyword. If the user query provided to the *Retriever* is considered the main topic, then the distinctive topics extracted from the candidate pool are sub-topics.

5. **Context:** In this master thesis, we define a context as a particular domain or field in which the user is interested. For example, in the user query *Cloud*, the retrieved documents are related to different domains or contexts, such as cloud computing, combat cloud, and clouds in the sky. Even though there is some syntactic and semantic matching, the user intention is still unclear from the query.

6. **Labeler:** A person who assigns an appropriate label to the data according to the labeling criteria.

7. **Query type:** Input search queries from the user can be of any form. For example ., abbreviation, single word query, etc. In this master thesis, each form of a possible user query is referred to as a query type. All possible user queries can be categorized into two major query types, namely phrase (three words or less) and sentence queries.

2.3 IR system setup

A document retrieval system was developed to support users at FKIE in retrieving news articles related to technology and military topics. The retrieval setup contains three primary

components: *Web scraper, Document filter*, and *Retriever*, as shown in Figure 2.3. The first component, the *Web scraper*, downloads news articles (HTML pages) from a list of URLs and cleans the raw HTML data from advertisements and noise. Each cleaned news article is considered as a single entity, namely a *Document*. The majority of downloaded documents are a mixture of topics such as military, technology, artificial intelligence, etc., and also contain a small number of typical news topics, namely politics, sports, advertisements, etc. The downloaded news articles are in *German* and *English*.

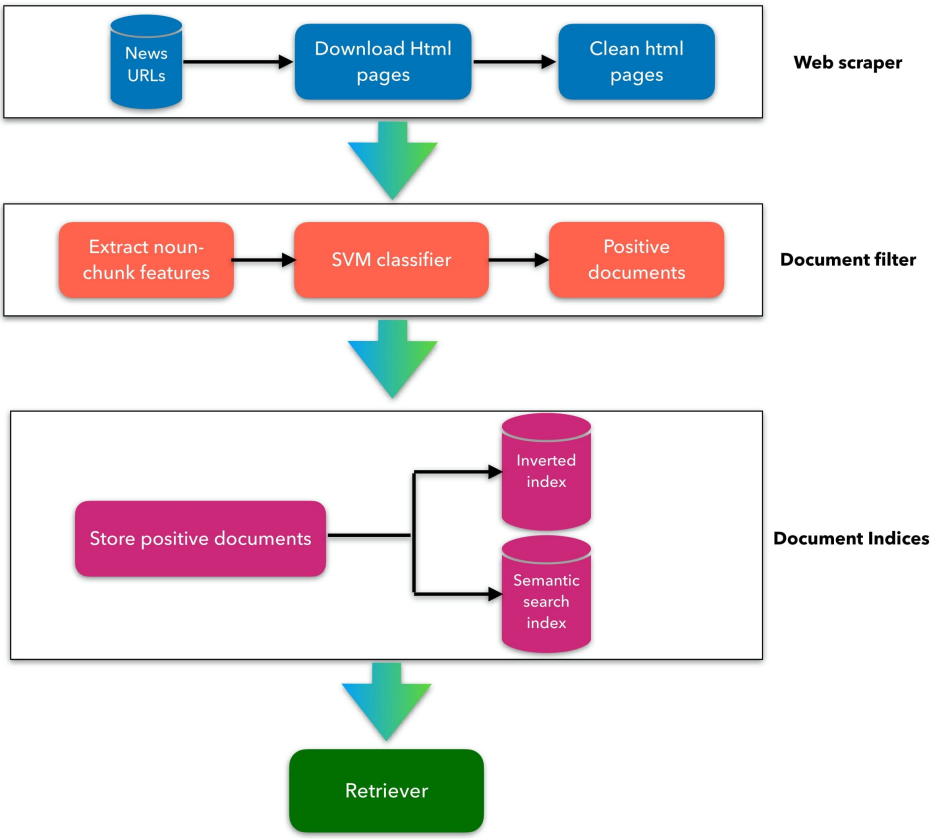


Figure 2.3: Document retrieval system designed at Fraunhofer FKIE

The second component, the *Document filter*, is based on Support Vector Machine (SVM) classifier that filters most of the irrelevant documents related to specific news topics. The documents are classified into two classes namely, *Positive* and *Negative*. *Positive* documents are documents related to technology and military, and *Negative* documents are related to everything else. After several tests at FKIE, it was found that features from noun-chunks in a document are performing better to differentiate *Positive* documents from the *Negative* documents. Noun chunk features based on the pre-trained multi-lingual Universal Sentence Encoder (USE) is used for the task of classification.

In order to facilitate positive documents to the FKIE users, a *Retriever* component is designed to retrieve documents for a given user query using lexical and semantic matching techniques. Therefore, the positive documents from the *Document filter* stage are stored in two different document indices namely *Inverted index* and *Semantic search index*. Finally, the component *Retriever* uses both of the indices and retrieves documents according to the user request through a web user interface, as shown in Figure 2.4.

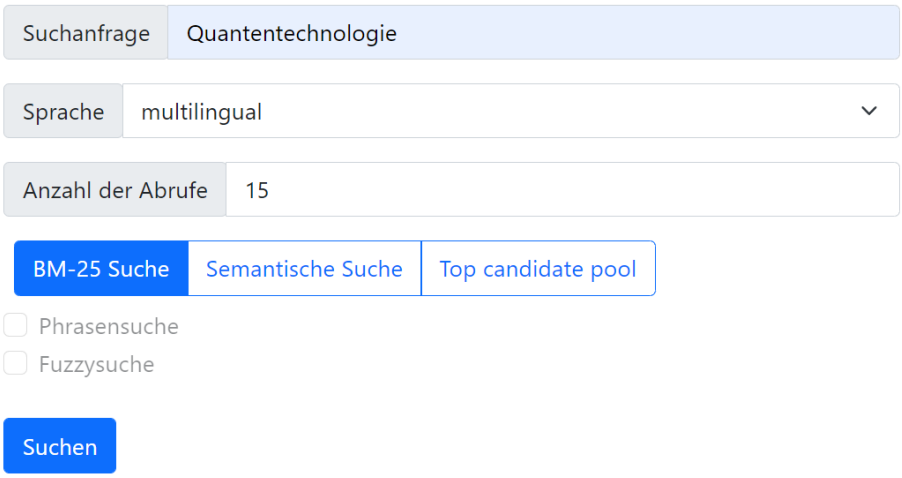


Figure 2.4: User interface to retrieve documents for FKIE users

2.4 Problem description

Semantic matching of query and documents is better suited when the user query is a long sentence query due to the context embedded in the search query. For example, the user query *"What are the technological advancements in Robotics related to Unmanned Weapon Systems?"* provide high-quality results in the top results, as the information request is detailed in the query. Consequently, the search query *"Robotics"* results are mapped to multiple domains and lead to many false positives (according to the user's intention).

Table 2.1: Retrieval algorithms comparison on different query types

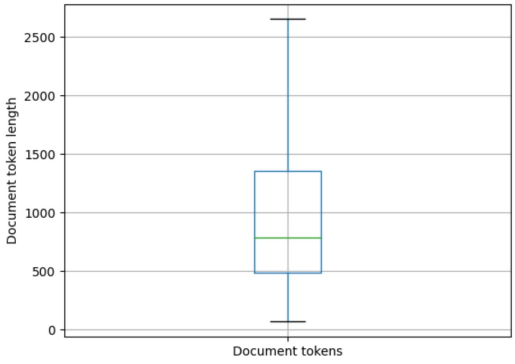
S No.	Query type	Better retrieval technique	Reason	Queries used
1	No meaning queries	BM-25	Lexical matching	Person or object names[5]
2	Multi-lingual queries	Semantic search	Semantic matching	Artificial Intelligence vs Künstliche Intelligenz
3	German composite words	Semantic search	Semantic matching	Quantentech-nologie
4	Spelling mistakes	Semantic search	Semantic matching	Kryptografy, Rbot
5	Polysemy	Semantic search	Semantic matching	Combat Cloud, Cloud computing
6	Sentence/long phrase queries	Semantic search	Semantic matching	Schwachstell-enanalyse eigene Waffen-Systeme

In the case of keyword queries, it was observed that semantic and lexical matching are prone to high false positives and have no unique advantage. In [35], the authors observed a similar

[5]User query with no innate meaning of the word namely out of vocabulary words: for example John Dowe, Wester etc.

challenge in their research. On the one hand, lexical matching does not consider the inherent meaning of the word causing a vocabulary mismatch problem, and semantic matching fails to retrieve the relevant documents in the top results as it matches too many keywords semantically. A manual observation of retrieved results is carried out with a set of sample queries to evaluate the retrieval algorithms, and the results are shared in Table 2.1 on page 20.

The users at FKIE provide only one or two phrase queries, and his or her intention is to explore information to specific topics such as *"Technology"* and *"Military"*. Without labeled data, learning user intention from a single word or phrase query is a huge challenge. One further challenge is that a wide variety of sources can also result in high noise or false positives, and the user is less likely to find the relevant documents in the top results. Unlike tweets or requirements, news articles are long documents with the 50% (percentile) token length of 788 and consist of keywords from multiple domains. Document token length details is shown in Figure 2.5.



(a)

S No	Statistical parameter	Value
1	Minimum token length	69
2	Mean token length	1280.64
3	Maximum token length	122657
4	Token length at 25%	489
5	Token length at 50%	788
6	Token length at 75%	1355

(b)

Figure 2.5: (a) Bloxplot showing the document token length distribution(after removing the outliers) (b) Important statistical details about the token length

Information related to innovation and technological breakthroughs is hard to find in the news articles. However, the probability is not zero, as positive news articles are gathered during data collection for classification. Nevertheless, their low distribution makes it challenging to create a dataset sufficient for supervised approaches. After considering the challenges with positive documents for the user intention, a supervised solution is hard to achieve, in order to match the performance of a full sentence query. Real-time user feedback and continuous reinforcement algorithms can fulfill the lack of labeled datasets, but they need feedback from diverse users regularly. Otherwise, the search results can be highly inclined to a particular user and lead to biased results.

A template-based search query is an option to improve the context of a search query. For example, we have a pre-defined template such as *Innovations in XXX related to the Military*. When the user provides a query: *Robotics*, we replace the *XXX* with the user query, and this results in the final query *Innovations in Robotics related to Military*. An option to update the template according to the user's interest from the user interface can provide tailored results without any extra training. This approach restricts the user to having only a few sets of templates and is also inefficient when a new template needs to be added, or an existing template needs to be updated.

After considering various approaches to fulfill the missing context, we believe that extracting the contexts from top results to the user query in an unsupervised way is more efficient, more explainable and can be better reproducible compared to supervised approaches. This would

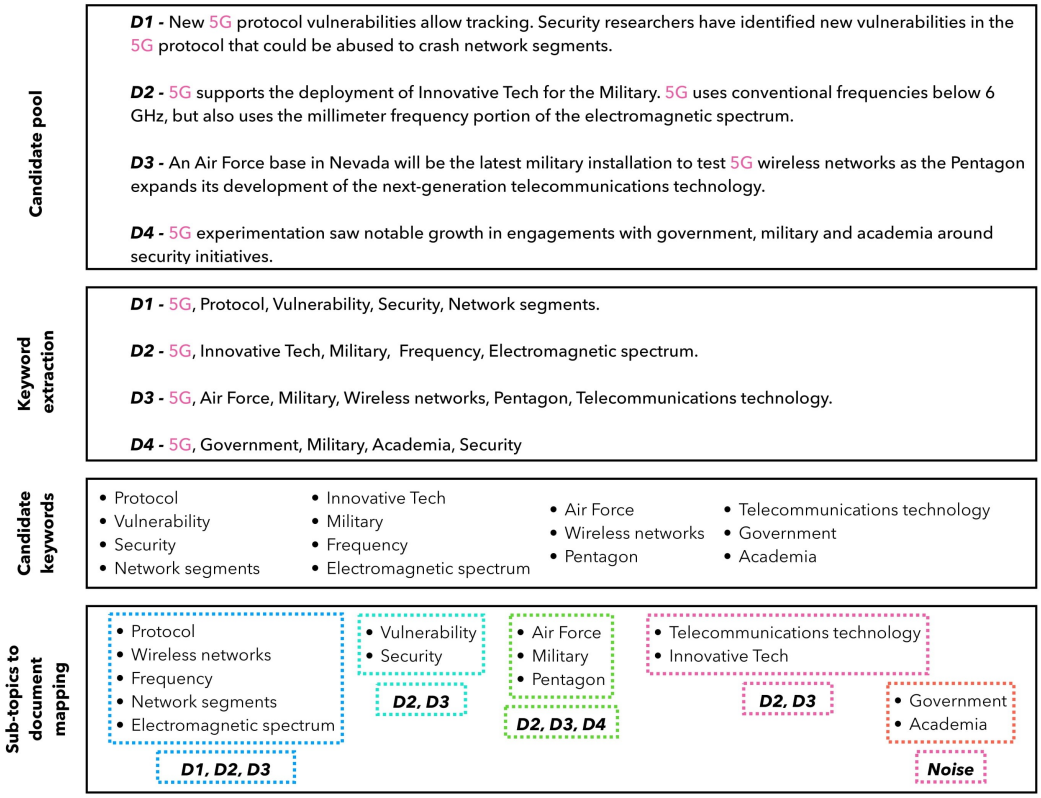


Figure 2.6: Expected sub-topic extraction for the query: 5G

not only help the user to have deep insights into the results pool but also reduce the efforts to reach the highly relevant documents. A sample expected sub-topic extraction pipeline output is shown in Figure 2.6. These contexts are described as *sub-topics*. The proposed approach in this master thesis is aimed at handling the challenges mentioned above. News articles from diverse sources are considered, and the results can be easily transferred to other data sources in the future.

Chapter 3

Technical background

3.1 Sentence encoders

In most of the NLP techniques, text documents are represented in the form of distributed vectors. These vectors are dependent on the word or phrase level representations and are used for exploratory analysis, text classification, clustering, text retrieval etc.. In addition to the phrases, the meta information such as parts-of-speech and entities can further help to improve the performance of these tasks. The advancement of the phrase representations in the last decade have made a huge impact in NLP research. In this section, different phrase representations from the literature are reviewed and the representations that are used in this master thesis are highlighted.

3.1.1 Word embeddings

Word embeddings are vector representations of words or phrases. One of the earliest approaches namely "Bag of Words" (BoW) method represents a text document as a feature vector denoting the normalized frequency of words in that document [68, 56]. BoW is a simple technique of representing the text in a high dimensional space and every word represents a dimension in this space. This representation of text in continuous space is referred as Vector Space Model (VSM) [2]. In matrix notation, this can be interpreted as Term-Document matrix. For example,

Bow image – term-document matrix

Considering only the word frequency can dominate the words which appear very often in a document and also does not consider the relation between documents in the corpus. To overcome this, an effective BoW approach namely Term frequency-Inverse document frequency (Tf-Idf) is proposed. In this modeling technique, how often a word appears in all documents in the corpus is considered along with term frequencies (or normalized) [37]. The terms which very often in many documents are now penalized and results in a better weighting score.

tf-idf formula

The above two approaches ignore the word meanings completely and also depend on the vocabulary of the corpus. This can lead also to computational issues when the length of vocabulary is in the range of millions. With the advancement of deep learning techniques, the documents representations are significantly improved and overcame the limitations of the above

approaches. This is possible with the help of better semantic word representations which considered word meanings and embed vectors in continuous vector space rather than a discrete vector denoting a single dimension [47, 53].

In [47], the researchers proposed an unsuperived training approach with two model architectures to generate high quality semantic word vectors. These two architectures are namely Continuous Bag-of-Words (CBOW) model and Continuous Skip-gram model and consider a text document as continuous series of words (as a window of words). These architectures are generally referred as *Word2vec* Moreover, these words are referred to either context words or a current word at a given time. These architectures also consider a window which spans the length of the context words. The CBOW architecture predicts the probability of the current word given the surrounding context words and the skip-gram architecture predicts the probability of context words given the current word [47]. Accordingly, the words sharing the same context in the corpus are represented close to one another in the continuous vector space. Both these architectures utilizes a 2-layer feedforward neural network to generate the distributed representations for all words in the corpus. While training the neural network, instead of a non-linear hidden layer a projection layer is used, which is shared for all words. An example showing the working of CBOW is shown below.

CBOW example

The above architectures have shown a great results representing the semantics of the words. For example, the vector equation *"king - man + woman = queen"* signifies the advancement of word representations in the vector space. However, only the local context is given importance and global context (at corpus level) is ignored in creating these word representations in both CBOW and skipgram architectures [53]. Local context here indicates the relationship between the words that exist at a time in a document or word co-occurrences. In [53], researchers have tested an approach namely *GloVe:* Global Vectors for word representation, an unsupervised algorithm to generate distributed word representations using local and global word-word co-occurrences. Glove uses a global log-bilinear regression model combining global matrix factorization and local context window techniques [53]. Glove embeddings have outperformed CBOW and other models in tasks such as Named Entity Recognition (NER) and word similarity [53].

Polysemy is a concept of word or sign having multiple meanings. For example, the word *bank*, which represents multiple things such as river bank or the bank where we keep money. Polysemy is one of the main drawback in both *Word2vec* and *Glove* approaches, as it ignores the possibility of word having different semantics. *ELMo:* Embeddings from Language Models is one of the earliest approaches to address this problem and generates contextual word embeddings [55]. ELMo word representations are a functions of the complete sentence rather than context created by a fixed window length. Furthermore, word embeddings are generated from the internal states of a bi-directional Language Model (biLM) [55]. ELMo also uses character convolutions (taking character-level tokens as input for the biLM) which allows the model to handle the word representations of words that out of vocabulary. In [25], researchers have claimed that the ELMo token representations which are computed by combining the left-to-right and right-to-left representations from biLM are not deeply bidirectional.

To generate deeply bidirectional word representations, an approach named *BERT:* Bidirectional Encoder Representations is proposed [25]. BERT utilizes the transformer architecture with multi-head attention to compute the efficient embeddings in the vector space. Transformer architecture uses self-attention mechanism with encoder-decoder structure to generate input and output representations [63]. BERT uses "masked language model" (MLM) and "next sentence prediction" (NSP) techniques in its pre-training. MLM randomly masks token from the

input sequence and predicts the probability of the missed token based only on its context. This random masking has enabled BERT to produce truely deep bidirectional representations [25]. Moreover, the NSP technique creates sentence pairs and jointly pre-trains the text-pair representations by predicting the next sentence given a particular sentence.

3.1.2 Sentence embeddings

Sentence embeddings are numerical representations of a text document namely a sentence or a paragraph and they play a crucial role in IR specially in semantic search. As a naive approach, one can generate sentence representations as the mean of vector representation of words in a sentence or paragraph (after removing the stop-words). This approach clears lacks the semantic coherence between the words and represents the context poorly in the continuous space. In the recent years, a lot of sentence encoding approaches were proposed, but only two popular techniques are discussed below.

1. **Sentence BERT (SBERT):** BERT encoder provides the input representations for a sequence of input tokens, which can be single sentence or two sentences combined together [25]. BERT uses some special tokens for recognizing certain features in the input sequence. For example, $[SEP]$ token is used to separate two contiguous sentences. BERT works on token level embeddings as a input rather than a single sentence. Therefore, in order to perform semantic similarity tasks, all sentence combinations shall be provided to the BERT network, which creates a massive computational overhead [57]. Authors in [57] have highlighted the disadvantage of BERT to generate independent sentence embeddings.

   A modification to BERT pre-training stage is proposed namely SBERT to generate sentence representations directly for a given sentence. SBERT uses a siamese and triplet networks for updating the network weights to generate sentence embeddings that are semantically similar [57]. Furthermore, pre-training of SBERT model is sufficient and does not need any further inference networks to generate the sentence representations. SBERT sentence embeddings can be directly used in many NLP tasks such as semantic text similarity (STS), text classification, text clustering, etc and the models built on SBERT's architecture are accessed with the help of the python library sentence-transformers[1] .

2. **Universal Sentence Encoder (USE):** USE is a pre-trained DL model from tensorflow that encodes text data such as sentences, phrases, or paragraphs into a distributed semantic vector [18]. Two variants of USE models were proposed depending up on the performance. One model makes use of the transformer architecture and the other uses the deep averaging network (DAN). Similar to the SBERT, USE also enables the transfer learning using the sentence embeddings and no further inference is required. Transfer learning tasks performed using both these architectures have revealed that transformer has performed better than DAN in different transfer learning tasks but requires high computational resources.

   One year later, two new multi-lingual USE (M-USE) models were proposed from tensowflow and the model embeds text from sixteen different languages into a single semantic space [65]. One model uses the transformer architecture and the other uses convolutional neural network (CNN) architecture. M-USE uses a multi-task dual encoder training framework which enables the DL model to train multiple languages simultaneously [65].

[1] `https://pypi.org/project/sentence-transformers/`

In this master thesis, a USE model with transformer architecture from Tensorflow Hub[2] is used. Tensorflow-hub specifies that this USE model is optimized for mutli-word text elements, such as sentences, phrases or short paragraphs and there is no need to explicitly mention the language of the text.

3.2 Dimensionality reduction

Sentence or word embeddings are very high dimensional distributed vectors, specially M-USE embeddings which have a dimension size of 512. Data processing tasks such as visualization, data analysis, feature extraction, clustering etc., on data having high dimensions can be computationally expensive. Therefore, the dimensions of the data are reduced for all the data points in the dataset without losing the crucial patterns or information. This technique is generally referred as dimensionality reduction. Many dimensionality reduction techniques were proposed in the recent years and they can be categorized into two types. Algorithms that preserve the pairwise distance structure among all the data points in the dataset [45]. One example in this category is Principal Component Analysis (PCA). PCA assumes that the data is linear and does not perform well in case of data having non-linear relationships. The other type of algorithms consider the non-linear relationships in the data and preserves local or global structures in the data. One example in this non-linear algorithms is Uniform Manifold Approximation and Projection (UMAP).

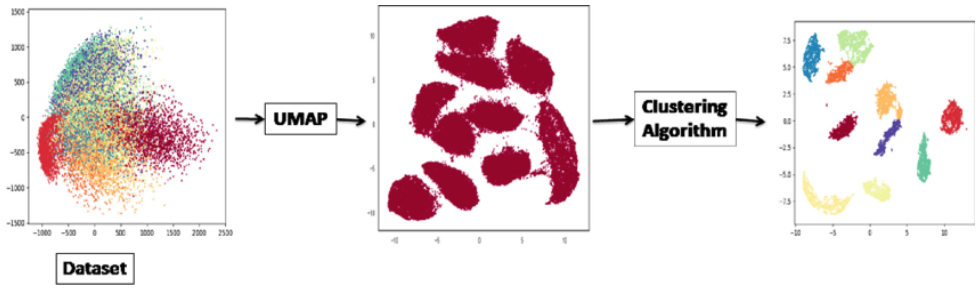


Figure 3.1: Improved clustering pipeline after reducing the data dimensions [3]

In [3], researchers observed an increase in performance of clustering methods after the dimensionality reduction on the data using UMAP. They have tested few popular clustering techniques such as k-means, Gaussian Mixture Models (GMM), Agglomerative hierarchical clustering, HDBSCAN and observed an increase in accuracy after UMAP dimensionality reduction. Moreover, the time taken for clustering is drastically reduced. UMAP works based on the Riemannian geometry and algebraic topology and preserves the global structure in the data [45]. One of the main reasons for acceptance of UMAP in ML is its computational efficiency. However, UMAP algorithm is slower than PCA [54], but given the quality of reduced representations and handling the non-linear relationship in the data, UMAP is clearly a viable dimensionality reduction technique in ML. In this master thesis, UMAP is used to reduce the data dimensionality.

3.3 Document clustering

Document clustering (DC) is the task of separating documents into meaningful groups where the documents with similar characteristics belong to a similar groups. In addition to DC, the

[2] https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3

task of topic modeling is also referred to achieve the same outcome. DC play a crucial role in the field of big data and data mining. Generally, clustering is performed on numerical data points in the continuous space. Text documents however are in alphanumeric (also some special characters) format. One of the earliest approaches to represent text documents is BoW method and tf-idf weighting, which is described in the section 3.1.1. Two main disadvantages of these methods is lack of semantic representation and high dimensional representation. As the dimensions depend on the number of words in the corpus, this can lead to a computational overhead. Semantic (contextual) text representations with the help of word or sentence embeddings, etc. encodes the text in a fixed length vectors. There are several clustering techniques tested on text document clustering such as partitioning, hierarchical, density based [2], etc. Below are few clustering algorithms discussed on an abstract level and their advantages and disadvantages are highlighted.

1. **Partitioning clustering:** This type of clustering deals with creating a fixed number of clusters of similar data based on a particular criteria. K-means clustering is one of the popular and simple clustering technique based on distance measurement in ML. K-means algorithm partitions the data of n samples into k clusters using a centroid-based iterative approach [2]. Being a parametric clustering approach, the number of clusters k needs to be well designed according to the data. One major drawback is the cluster shape, as the algorithm expects a spherical or circular shape output due to the centroid approach. K-means also does not assume any inherent noise in the data and assigns all data points to a cluster.

2. **Hierarchical clustering:** These clustering algorithms create a hierarchical structure from the data samples. There are two types of hierarchical clustering methods namely top-bottom/divisive approach and bottom-top/agglomerative approach [2]. In top-bottom approach, all data samples are considered to be a single cluster and this cluster is further decomposed into smaller cluster until a certain criteria is achieved. Adversely in bottom-top approach, each data sample is considered as a single cluster and the clusters are merged to form larger clusters until a certain criteria is met [69].

   Agglomerative hierarchical clustering is one of the popular hierarchical algorithms in ML and unlike k-means, there is no need to specify the number of clusters before clustering. However, an extensive experiments on comparing both the clustering algorithms shows that partitioning clustering performs always better than agglomerative clustering [69]. The authors also suggested partitional clustering for large document collections.

3. **Density based clustering:** Similar to hierarchical clustering, density based clusters are non-parametric and moreover separates data samples into clusters which have a high density areas until a certain criteria is met. Density can be interpreted as the number of points located within a certain region. Density-based spatial clustering of applications with noise (DBSCAN) is one of the popular density clustering algorithm. DBSCAN characterizes every data point as either a core point or a border point or a noise point [16]. Two crucial parameters in the DBSCAN algorithm are ϵ (epsilon) and $minPts$ (minimum number of points). A data point is defined as a core point when it contains neighbouring datapoints higher than $minPts$ within a circle of radius eps.

   A density-based cluster is expressed as a maximally connected component of the data points that lie within a distance less than eps from a core point (as described above) [16]. Border points are data points inside a cluster which do not follow the core point property. Data points that are not part of a cluster and does not follow the core point criteria are noise points. There are more parameters in this algorithm which define the final clustering output but the number of clusters is not a parameter. This helps the algorithm to assign number of clusters according to the given data.

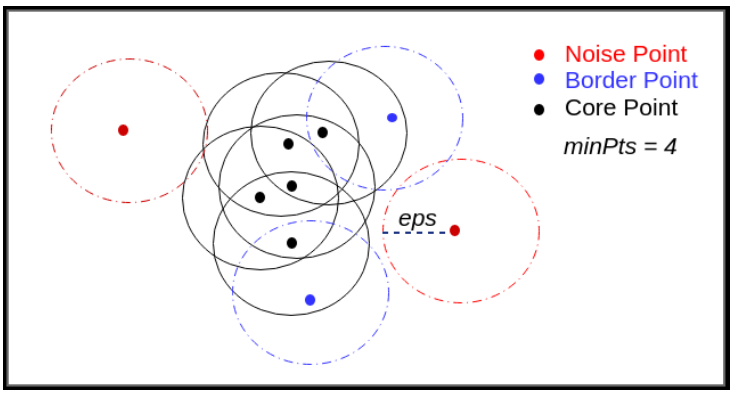


Figure 3.2: DBSCAN classification of data points [62]

Density based clustering clearly have many advantages compared to other clustering algorithms with efficient noise handling, non-parametric, flexible clusters (no specific shape and size). However, DBSCAN has limitations such as difficulty of parameter selection and varying density clusters [43]. To overcome this limitation, an algorithm namely Hierarchical DBSCAN (HDBSCAN) was proposed. This clustering algorithm extends DBSCAN by removing the concept of border points and varying different values of eps. A hierarchy of different DBSCAN clusterings are generated through different values of eps [43]. The hierarchy is condensed and used to find clustering output which provides stability of eps. To achieve a new parameter named "minimum cluster size" is introduced. In this way, HDBSCAN overcomes the limitation of handling varying densities and there is no need to explicitly select the parameter eps. However, HDBSCAN is computationally slower compared to DBSCAN.

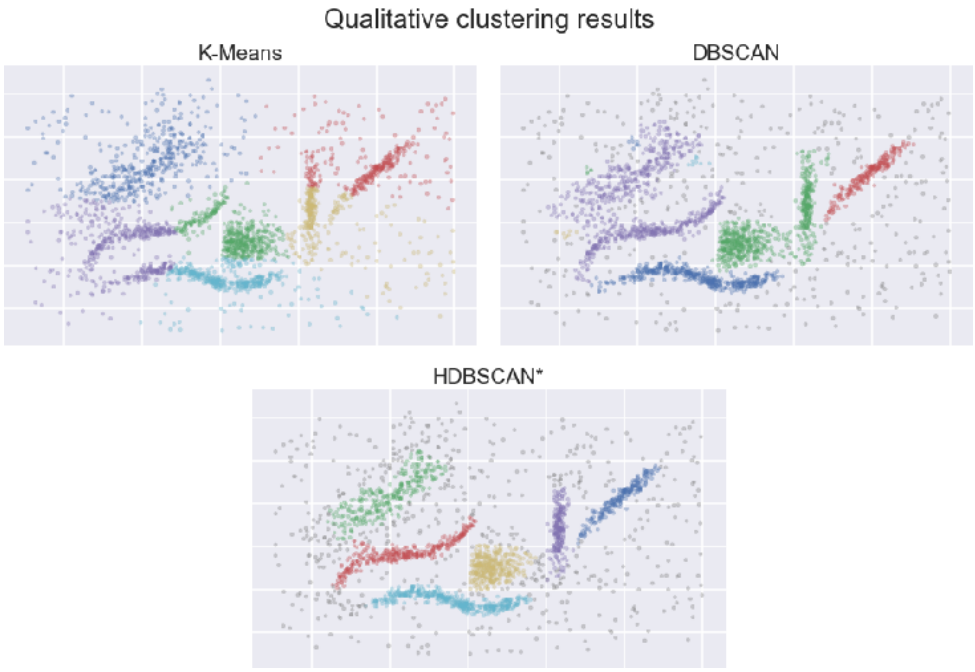


Figure 3.3: Comparision of clustering results from three algorithms namely k-means, DBSCAN, HDBSCAN [43]

Clustering algorithms can be further characterized into two types namely hard clustering and soft clustering depending upon the clustering output [22]. When the clustering algorithm

strictly assign each data point to a single cluster, it is referred to as Hard clustering. In DC, one document assigned to one cluster is an example of hard clustering output. When the clustering output assigns a data point to several clusters, then it is referred as Soft clustering. For example, when a document is assigned to several clusters, then it is soft clustering.

3.4 Topic modeling

Topic modeling (TM) is a technique of extracting inherent patterns or structures from a large collection of text documents [5]. TM is an unsupervised ML approach to express a text document as a mixture of topics. For example, a topic is general theme such as sports, politics, business, movies, health et,. Latent Dirichlet Allocation (LDA) is one of the popular TM algorithms which generates a soft-clustering output. LDA is a generative probabilistic model which represents each document as a finite mixture of topics and each topic as finite mixture of words [14, 5]. LDA uses BoW representation where a document is a finite set of words. One major limitation of LDA method is the lack of semantic representation.

To overcome this limitation a new approach namely top2vec is proposed. With the help of the distributed semantic representation of words and sentences, a text document can be represented in the continuous space. This gives an advantage to learn topics in the continuous vector space [5]. top2vec encodes the text documents into the vector space using sentence embeddings. These high dimensional embeddings are reduced to low dimensions using UMAP technique and further clustered using HDBSCAN clustering. During clustering, the high dense areas in the semantic space are grouped. This results in expressing a topic as a cluster centroid. In [5], the results show that top2vec finds topics that are more informative and representative than traditional topic modeling algorithm LDA.

Chapter 4

Related work

Many researchers have considered different techniques from Machine Learning (ML) to improve the retrieval results based on the availability of labeled data. The research can be categorized into two types: supervised and unsupervised.

4.1 Supervised approaches

Many researchers used ML algorithms with special loss functions based on relevance between the query, and documents and some of the popular pairwise ranking methods are RankBoost [27], RankNet [15], Rank-SVM [30] (using click-through data). Recent state-of-the-art supervised approaches are neural re-ranking methods and are based on complex Deep Learning (DL) architectures. Distributed word embeddings combined with the performance of non-linear neural networks have shown remarkable results in improving the performance of retrieval systems by considering semantics [48, 29, 51].

4.2 Unsupervised approaches

These approaches use no-labeled data and re-rank the retrieved results based on the user query and top retrieved documents [58, 6]. One common challenge in these approaches is the user query, which is mostly comprised of only a few keywords [6, 31]. To tackle this problem, many researchers have tested Query Expansion (QE) approaches that partially fill the missing meaning and context in the query. QE techniques include clustering search results, query filtering, word sense disambiguation, and relevance feedback, etc., [6]. Relevance Feedback is a method of retrieving search results using the original query given by the user and then using the top-k documents for query expansion [6]. Researchers have clustered search results in many different ways, such as at the document level, keyphrases, query-specific clustering, etc. [11, 34, 67, 52, 38, 39]. Typical distance-based clustering algorithms such as k-means are used in some research and also Hierarchical clustering is also tested [11, 46, 66], as it is flexible to change the threshold level for cutting the clustering dendrogram in a bottom-up approach. A common drawback in most clustering approaches is mapping a document to a single cluster, which is not logically valid, as a document can contain keywords from different domains.

4.3 Thesis contribution

The approaches based on clustering at the word level [11, 46] consider only a single language of retrieval results or corpus and hence cannot be directly implemented on a multi-lingual corpus and does not have any special keyword selection stage. With the advantage of contextual embeddings from sentence encoders, the authors in [5] made a breakthrough in document clustering with an efficient and explainable topic-modeling approach.

In [41], authors have used a particular candidate selection approach to filter some phrases from the keyword extraction and a specific noun chunks selection. This pipeline is explicitly used to extract innovation insights from research projects. As the user intention is related to *Innovation* at FKIE is proposed as a unique query-specific candidate keyword selection clustering. Moreover, the documents are semantically mapped to a specific topic, and multiple languages are modeled using a single multilingual pre-trained sentence encoder. News articles from multiple languages can be easily integrated into the document indices, and no changes are needed in the clustering pipeline. The proposed approach can be further extended to analyze any corpus containing long text documents for a given phrase or keyword.

Chapter 5

Thesis Methodology

5.1 Proposed methodology

One way to extract different contexts from the candidate pool is to perform any clustering algorithm. This results in very generic clusters closely related to a given query and does not provide any new insights to the user. To generate diverse and distinctive clusters, we need to use the latent information at the word or phrase level rather than at the document level [14]. As the documents contain multiple occurrences of the query and are also highly similar in semantic space, we need to reduce the impact of the given user query to generate a clear distinction between the documents. If we ignore the query-related keywords while clustering then it leads to a high number of clusters which are similar to each other (repetitive clusters) and can definitely impact the user satisfaction. Figure 5.1 illustrates the proposed approach on an abstract level to tackle the above issue and generate highly heterogeneous clusters.

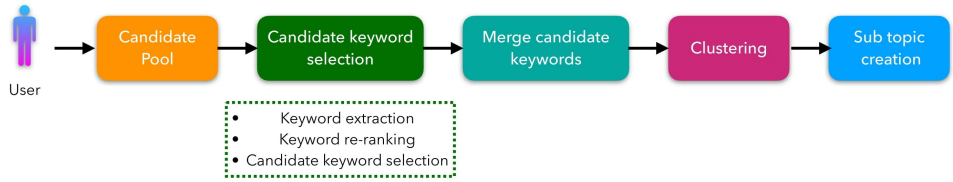


Figure 5.1: Proposed approach on an abstract level

The proposed approach, shown in Figure 5.1, does not assume fixed templates or specific user intentions. Major components in the pipeline are: *Candidate selection, Merge candidate keywords, Clustering, and Sub-topic creation*. This pipeline's first step is retrieving a candidate or retrieval pool for the given query. Subsequently, to extract keywords with high diversity and low noise (stopwords), a Candidate selection module is proposed. This component consists of three significant steps namely *Noun-chunk extraction, Keyword extraction, Keyword re-ranking*, and *Candidate keyword selection*.

Keyword extraction is extracting the most meaningful noun phrases in a text document. In the second stage, *Keyword re-ranking*, cosine similarity is calculated from the phrase embeddings between the keywords and the query. Using these similarity scores, keywords are then re-ordered in descending order. After this stage, the certain keyword, that are not similar to the query have a low cosine similarity score are precisely extracted, as they have a high potential

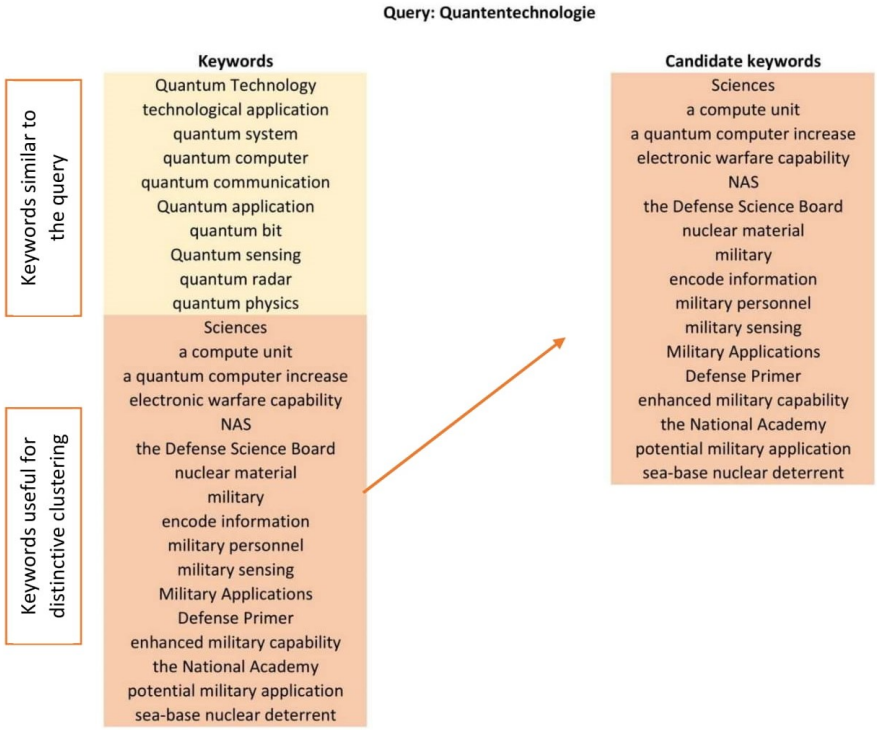


Figure 5.2: Candidate keyword selection step from a single document

for creating distinctive clusters or sub-topics. Specific keywords are selected and used for clustering using a cut-off threshold and this process is referred to as *Candidate keyword selection* and the resulting phrases after this stage are called Candidate keywords, shown in Figure 5.2.

The second component in the pipeline, *Merge candidate keywords*, merges candidate keywords from each document in the candidate pool and duplicates are removed. These keywords are then clustered semantically and modeled with the documents again. This process is also referred as Document to sub-topic modeling and is designed independent to the query and to handle multiple languages.

After clustering, sub-topics are extracted using a centroid approach. A mean phrase vector (centroid vector) is calculated from all the keywords inside a cluster and the closest keyword vector to the centroid vector is considered a cluster label. This process is named *Cluster labeling* and the cluster labels are considered sub-topics. After clustering, the individual clusters are considered as sub-topics. Sub-topics and documents inside a sub-topic can be further ranked before showing to the user. The pipeline ends with this last component, *Sub-topic creation*.

5.2 Candidate pool selection

Sub-topics related to the user query are extracted from the top retrieved results and to extract a wide variety of these sub-topics, a large set of retrieved documents for the query is required. This set is referred as *Candidate pool* and is comprised of retrieved results from both semantic and lexical matching. A diverse and large candidate pool is very crucial for generating a high variety of query related sub-topics. The length of a candidate pool directly influences the sub-topics output and two candidate pools of length around 30 and 100 are considered in

this approach. These document pools are constructed from an equal mixture of documents retrieved from lexical and semantic matching.

Algorithm 1: Algorithm to generate a candidate pool

Input: query - string, pool_size - integer
Output: candidate_pool - list $[candidate\_pool\_i]$, $i = 1, 2, \cdots, n$, where each element is a string

```
Function Get_candidate_pool(query, pool_size):

    pool_size_half = int(pool_size/2)

    /* get_elastic_search_results - retrieves documents which have high lexical
       similarity with the query.                                              */
    top_docs_lexical = get_elastic_search_results (query, pool_size_half)

    /* get_semantic_matching_results - retrieves documents which have high
       semantic similarity with the query.                                     */
    top_docs_semantic = get_semantic_matching_results (query, pool_size_half)

    /* removes common documents which are present in both sets                 */
    candidate_pool = remove_duplicates (top_docs_lexical + top_docs_semantic)

    return candidate_pool
```

It is assumed that the top retrieved documents are relevant to the query and the user and thus these retrieved documents are used for sub-topic extraction. This assumption can lead to poor sub-topics when the retrieved documents are entirely not related to the query specially in the case of semantic matching. In lexical matching, the retrieved documents contains the query keywords and ensures that the documents are at least partially relevant. However, in semantic matching, only cosine similarity is used as the selection criteria. For example, in order to create a candidate pool of 30 (CP-30), a top-15 documents from lexical matching results and top-15 documents from semantic matching results are combined. There is a clear possibility that the top semantically matched results are not entirely related to the user given query. Consequently, the cosine similarity of these top semantically matched results needs to be evaluated before creating the candidate pool.

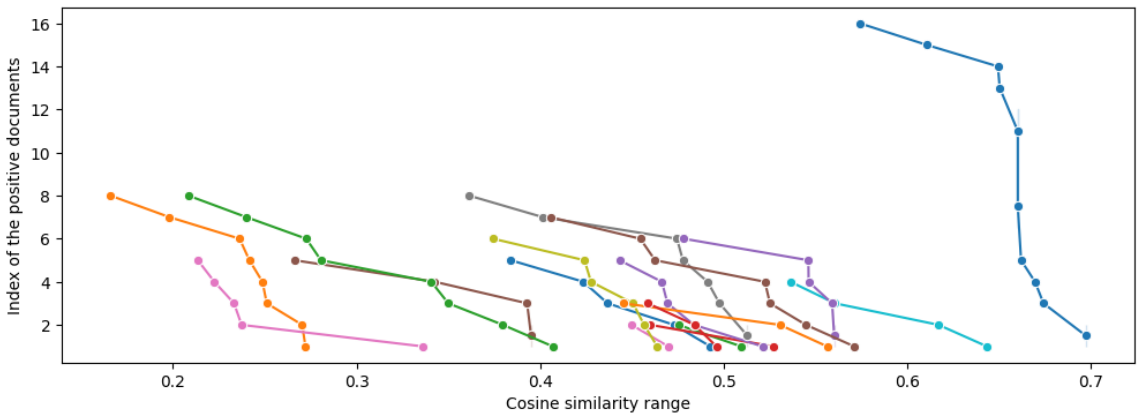


Figure 5.3: Cosine similarity between the query and retrieved documents

To perform this analysis, the cosine similarities between the query and retrieved documents is

observed. In retrieved documents, only relevant documents and the document which has the highest cosine similarity are considered. In Figure 5.3, each query is shown in an unique colored plot and it is clear that the spread of cosine similarity differs from query to query. Queries have a high cosine similarities with the retrieved results ranging at 0.5 and some queries even at a range up to 0.7. On the other hand, there are some queries with cosine similarity range below 0.3. Considering the maximum cosine similarity in the retrieved documents ($CS_{\max}$) as the reference, an appropriate lower similarity threshold ($CS_{\min}$) needs to be determined. So that only an optimal set of retrieved documents are selected for the candidate pool. This information is represented in the below equation with the help of a cut-off parameter cp.

$$CS_{\min} = (cp * CS_{\max})$$

From the above equation, the value of cp can be determined in multiple ways. One approach tested in this master thesis is the average min-max similarity ratio. The ratio is calculated from the mean of the minimum to maximum cosine similarities over multiple queries. Let us consider the maximum cosine similarity in the retrieved results set of query q is max_{q} and the minimum cosine similarity of the relevant document is min_{q}. Now the approximation of the value cp over N queries is described as:

$$cp = (\sum_{q=1}^{N} min_{\mathrm{q}}/max_{\mathrm{q}})/N$$

Applying the above approximation on the data from Figure 5.3 resulted the value of cp as 0.78. However, a slightly lower value of 0.75 is chosen at the end, as there is an expected human bias during data labeling. This approximation can even be modified further by an multiplying factor according to the human bias in the dataset. This optimal threshold selection is designed to prevent selecting the irrelevant documents for a given user query. This selection technique can be used in any semantic matching retrieval methods.

Algorithm 2: Algorithm to retrieve semantically similar documents with optimal selection

Input: query - string, pool_size - integer

Output: top_docs_semantic - list $[top\_docs\_semantic\_i]$, $i = 1, 2, \cdots, n$, where each element is a string

```
Function Get_semantic_matching_results(query, pool_size):

    MIN_THRESHOLD_SEMANTIC = 0.27
    CP = 0.75

    /* top_docs_semantic_search - contains semantically similar documents,
       doc_sim_list - contains respective cosine similarity score              */
    top_docs_semantic_search, doc_sim_list = get_semantic_search_results (query,
     pool_size)

    max_cosine_sim = max (doc_sim_list)
    min_cosine_sim = min (doc_sim_list)
    max_diff_cosine_sim = get_max_diff_sim (doc_sim_list)

    /* Optimal cutoff similarity selection from three individual cutoffs       */
    final_cutoff_sim = min (MIN_THRESHOLD_SEMANTIC, (CP * max_cosine_sim),
     max_diff_cosine_sim)

    top_docs_semantic = []
    for idx ← 0 to pool_size do

        doc = top_docs_semantic_search[idx]
        sim = doc_sim_list[idx]
        // Selecting documents that have high cosine similarty than the optimal
           cutoff similarity.
        if sim > final_cutoff_sim then
            top_document_semantic.append(doc)

    if len (top_docs_semantic) < 10 then
        /* Selecting only top 10 in case of poor similarity distribution between
           the query and documents                                             */
        top_docs_semantic = top_docs_semantic_search[:10]

    return top_docs_semantic
```

Furthermore, it is also observed that few query to document similarities are highly abnormal. For example, cosine similarity distributions below 0.3 in Figure 5.3. To handle these exceptional cases, a robust candidate pool selection approach is proposed. In case of no results at the end of the selection, top-10 semantically matched documents are selected. After the successful selection of documents from the semantic matching, the documents from lexical matching are combined and duplicates are removed. This final set of documents without any redundancy is referred as the candidate pool.

Algorithm 3: Algorithm to calculate similarity at the maximum difference

Input: sim_list - list $[sim\_list\_i]$, $i = 1, 2, \cdots, n$, where each element is a float
Output: max_diff_sim - float

```
Function Get_max_diff_sim(sim_list):

    diff_list = []
    sim_list_len = len(sim_list)

    for idx ← 1 to sim_list_len do
        // store the difference in similarities at recurrent indices.
        diff_list.append(sim_list [idx] - sim_list [idx - 1])

    max_diff = max(diff_list)
    // Get the index where the similarity difference is maximum.
    max_diff_index = diff_list.index(max_diff)

    max_diff_sim = sim_list[max_diff_index]

    return max_diff_sim
```

5.3 Candidate keyword selection

This section details the steps involved in selecting certain keywords and their criteria involved in selection. The main objective of this stage is to select very diverse noun-chunks from each text document. Candidate selection is the most important step in the whole pipeline and the quality of the clustering output is directly dependent on the output of this step.

5.3.1 Noun-chunk extraction

News articles are very long text documents and the most of the crucial information in a document lies in nouns. A noun is one of the parts of speech element which identifies a person, place or a thing and a noun-chunk is a noun which is a phrase (group of words). To extract the noun-chunks automatically from each document, a library named spacy[1] is used. The noun-chunks from spacy are closely analyzed and some inconsistencies are identified which needed further cleaning. To clean these noun-chunks, a special pipeline is proposed to target the following noise elements: *stopwords, punctuation, determiners, duplicates, long noun-chunks*.

- **Remove longer Noun-chunks:** It is observed in the output of spacy noun-chunks that there are some longer noun phrases of length more than 3 words and are mostly consists of no significant information. Consequently, these noun-chunks are removed from the spacy output. For example, *"standort- und zeitunabhängigen Zusammenarbeit"*, *"Every successive cellular generation"*. There might be useful information in these long noun phrases, but extracting that information is very challenging and the ratio of occurrences of these phrases are relatively smaller compared to phrases that have a length below 3. Therefore, these long noun phrases are filtered from the noun-chunks set.

[1] https://spacy.io/

- **Remove stopwords:** Stopwords are the words which carry no significant information and thus are omitted from the noun-chunks. There is no universal set of stopwords in any language and also there are numerous types of stopwords are observed in the news-articles in both english and german languages. Therefore, a robust set of stopwords from various sources[2] is collected in both the languages. Stopwords can appear either as a complete noun-chunk or as a part of a noun-chunk and are removed in any case. Stopwords generally consists of determiners, adjectives, articles, prepositions, etc,. For example: *"der"*, *"die"*, *"das"*, *"the"*, *"from"*, *"front"*, etc,.

- **Remove numeric noun-chunks:** Some noun-chunks have numeric values which contains no useful information and furthermore it is observed that the complete noun-chunk does not convey have any significance either. Thus the numeric noun-chunks are removed from the noun-chunks set. For example: *"mehr als 50 Ländern"*, *"1,95 m Länge"*, *"1,5 Milliarden"* etc,. However, the noun-chunks where the numeric value is attached to a alphabet are ignored i.e., "4G Technology", "2D-Zeichnungen" etc,.

- **Remove punctuation:** This step is designed to remove all sorts of punctuation elements and keep the noun-chunks clean and more readable to the user. For example: the original noun-chunk *"Bündnis- und Landesverteidigung"* is transformed to *"Bündnis Landesverteidigung"* after removing the hyphen (-) and stopword (und).

- **Lemmatization:** To lemmatize the noun-chunks spacy module is used again. Both the english and german noun-chunks are lemmatized using spacy. For example: *"Ansätze"* is lemmatized to *"Ansatz"*.

- **Fuzzy redundancy removal:** It is observed that there are a lot of noun-chunks which are syntactically similar to each other. These similar noun-chunks can be categorized into two types namely exact duplicates and close duplicates. Exact duplicate noun-chunks can be easily filtered in python, but identifying the close duplicates and filtering is a challenge. For example: the noun-chunks *"5G network"* and *" 5G 4G network"*. Only one of noun-chunk that is smaller is retained and the other is removed. The main objective of this step is to reduce the similar phrases and increase diversity in the data. This can definitely lead to a loss of information to a certain extent.

  Accordingly, a fuzzy string matching based noun-chunk removal approach is designed to tackle the above problem and close duplicates are removed to a certain extent using the FuzzyWuzzy library. FuzzyWuzzy is an open-source library in python to calculate the syntactical text similarity between two strings. Internally the library uses Levenshtein distance. FuzzyWuzzy assigns a score between 0 to 100 according to the similarity between two strings.

[2] `https://gist.github.com/sebleier/554280`, `https://countwordsfree.com/stopwords`, `https://solariz.de/de/downloads/6/german-enhanced-stopwords.htm`

Algorithm 4: Algorithm to remove close duplicates

Input: noun_chunk_list - list $[noun\_chunk\_list\_i]$, $i = 1, 2, \cdots, n$, where each element is a string

Output: cleaned_noun_chunk_list - list $[cleaned\_noun\_chunk\_list\_i]$, $i = 1, 2, \cdots, n$, where each element is a string

```
Function Remove_close_duplicates(noun_chunk_list):

    // Removing exact duplicates using set in python
    noun_chunk_list = list(set(noun_chunk_list))

    close_duplicates = []
    noun_chunk_list_len = len(noun_chunk_list)

    for i ← 1 to noun_chunk_list_len do
        phrase_1 = noun_chunk_list [i]

        for j ← i + 1 to noun_chunk_list_len do
            phrase_2 = noun_chunk_list [j]

            // fuzzy gives the syntactical text similarity score
            if fuzzy (phrase_1, phrase_2) > 85 then
                // When the score is high, the smaller phrase is recorded
                close_duplicates.append(get_shorter_text(phrase_1, phrase_2))

    // Removing the close duplicates using recorded close_duplicates
    cleaned_noun_chunk_list = list(set(noun_chunk_list) - set(close_duplicates))

    return cleaned_noun_chunk_list
```

5.3.2 Keyword extraction (KE)

The cleaned noun-chunks describe a text document very well. However, not all noun-chunks are of great significance to represent a document specially in case of a news-article. As the length of the news-article is large, the size of extracted noun-chunks set can also be large. This large set of noun-chunks is well processed syntactically and needs to be processed now semantically to select only noun-chunks that can greatly influence the meaning of a text document (a news article). These crucial noun-chunks in a text document are referred as *Keywords*. In [8], authors describe keyword extraction as the task of automatically identifying the terms that best represent the most relevant information contained in the document.

Keyword extraction techniques are majorly classified into either unsupervised or supervised approaches[10]. Supervised KE approaches require a large amount of labeled dataset of both documents and keywords from each document. Annotating keywords manually from each document is a very expensive and tedious task [8] depending on the length of the documents. Unsupervised approaches on the other hand does not require any labeled information, but often have poor accuracy[10]. Following the research, there are a lot of KE approaches and are categorized on an abstract level as shown in .

In this master thesis, an approach to select keywords based on the contextual embeddings is chosen. This approach is inspired from the recent research related to using contextualized sentence embeddings for keyphrase extraction namely EmbedRank [10]. In this research, it is shown that the EmbedRank approach has performed better than the state of the art graph-based KE approaches. The cleaned noun-chunks generated from the earlier stage are used to generate keywords. To generate phrase embeddings of the noun-chunks and the original news article, a multi-lingual pre-trained Universal Sentence Encoder (USE) from tensorflow is used. USE used in the thesis is a transformer based architecture that can embed a text input from 16 different languages into a single semantic space [65]. The target languages in our dataset namely english and german are included.

Transformer architectures compute context-aware embeddings of tokens in a sentence considering the order and identity of the tokens and these token level representations are further averaged to calculate senetence embeddings. There is a possiblity that the sentence level embeddings as mean of token embeddings can result in poor representations of long text documents. It is observed in the news-articles that the length of the text documents is longer and different contexts are discussed in different paragraphs. Therefore, a document is divided into multiple paragraphs and mean of all paragraph embeddings is considered as *Document representation vector*. A paragraph length of 500 tokens is considered in this approach. For example, a text document with a token length of 1600 is divided into 4 paragraphs of length 500, 500, 500, 100 and the final representation vector is a mean representation vector of these paragraph vectors. The best possible parameter for the paragraph length is not explicitly tested in this master thesis and can be considered as a future work.

Phrase embeddings for each cleaned noun-chunks are computed using the same USE model. With help of noun-chunk embeddings and the document representation vector, the relevance of a noun-chunk in a document is expressed as the cosine similarity between the vectors in the semantic space. The higher the similarity, the better the relevance of the noun-chunk in document. Consequently, all noun-chunks are ranked according to their similarity of relevance with the document. Top-25 noun-chunks with highest relevance similarity are considered as *Keywords*.

5.3.3 Candidate keyword selection

Keywords extracted from a document from the earlier stage are syntactically diverse as they are derived from the noun-chunks generated by removing the close duplicates. Let us consider the case to use these keywords to distinguish documents in an IR setup where the user provides a search query and expects relevant documents. Given the user query and the retrieved candidate pool, a manual preliminary analysis has shown that there are certain keywords which are not similar to the user query and carry a significant value in improving the information search behavior. Therefore, the task of removing the query related keywords and retaining the highly distinctive keywords is referred as *Candidate keyword selection*.

Let us consider a text corpus C that contains n documents where each document D is a news article. These documents are saved inside two different indices namely inverted index and semantic search indices to retrieve documents for a given search query.

$$C = \{D_1, D_2, D_3, \ldots, D_n\}$$

Once the user provides the IR system a search query q and a candidate pool CP of size m is generated, which is a document set.

$$CP_q = \{D_i, D_j, D_k, \ldots\}$$

Taking one document D_i into account, a keywords (k) set is extracted using the above steps namely noun-chunk and automatic keyword extraction. A news article is now transformed to a set of key phrases and can be expressed as below.

$$D_i = \{k_1, k_2, k_3, \dots\}$$

The objective of this step is to efficiently select certain keywords that are similar to the query. This similarity selection must also consider the close semantic and multi-lingual nature of the keywords. Accordingly, cosine similarity is considered and multi-lingual USE model is used to encode the query and the keywords into semantic embeddings. The similarity between query and document keywords is a function of distance in semantic space. The keywords with high cosine similarity must be selected to have a highly diverse keywords to the query. A parameter namely Candidate keyword selection cks is proposed to remove the query similar keywords.

The selection parameter must consider the distribution of similarity between query and keywords in a document. Moreover, it should be independent to a document. A percentile selection is adopted as a selection criteria in order to avoid a static similarity threshold. For example, the parameter csk takes the value 30 which signifies that the keywords with similarity under similarity of 30 percentile are retained and the keywords above 30 percentile are removed.

To the best of my knowledge, the testing of this selective approach to improve clustering and IR performance is one of the earliest attempts in the IR system research. Further criteria and parameters for better keyword selection other than query similarity shall be explored in the future.

5.4 Clustering

Candidate keywords from each document in the candidate pool are combined and duplicates are removed to create a final set of candidate keywords. These keywords are further clustered to generate distinctive sub-topics. This stage has three main steps namely *Phrase embeddings extraction, Dimensionality reduction, and Hierarchical clustering*. To achieve semantic clustering, multi-lingual pre-trained sentence encoders are used to generate phrase embeddings for each candidate keyword. These densely distributed embeddings are usually highly dimensional (512) and clustering in high dimension space is complex to capture patterns and can be resource intensive. Therefore, the embeddings are compressed with a dimensionality reduction technique namely UMAP. Therefore, the dimensionality of the embeddings is reduced without losing underlying information in the data using the UMAP algorithm [45] from the umap-learn library. Below are few key parameters in the UMAP algorithm.

These embeddings are further clustered in low dimension using a hierarchical clustering algorithm. Noise is expected in the candidate keyword extraction phrase and all the keywords are not important for modeling. So, clustering algorithms such as k-means, Gaussian Mixture Models, etc., are not suggested in this case, as they consider all data-points while clustering. Consequently, HDBSCAN and DBSCAN clustering algorithms are preferred as they innately consider the noise in the data and avoid assigning a cluster for every data point. One major advantage of these algorithms is that the number of clusters is not a parameter and the algorithm creates clusters effectively based on the data. HDBSCAN algorithm [44] with its varying epsilon and merging clusters has shown robust clustering results by finding varying density clusters and the same algorithm is considered in this master thesis. This clustering pipeline is already tested and shown great results with documents in recent research [5]. Below are few key parameters used in the HDBSCAN algorithm.

5.5 Sub-topic creation

5.5.1 Sub-topic labeling

After clustering, sub-topics are extracted using a centroid approach. A mean phrase vector (centroid vector) is calculated from all the keywords inside a cluster and the closest keyword vector to the centroid vector is considered as a cluster label. This process is named as *Cluster labeling* and the cluster labels are considered sub-topics. After clustering, the individual clusters are considered as sub-topics. Sub-topics and documents inside a sub-topic can be further ranked before showing to the user. The pipeline ends with this last component, *Sub-topic creation*.

Thereafter the keyword set M_q is clustered into r groups (s) and is defined as a sub-topic set S_q.

$$S_q = \{s_i, s_j, s_k, \ldots, s_r\}$$

Each sub-topic (s) is again expressed as a set of keywords (k). The number of keywords in a sub-topic cluster can vary from cluster to cluster.

$$s_i = \{k_x, k_y, k_z, \ldots\}$$

The mapping between the document and keywords is already known from above, now we can express each document as a set of sub-topics, and also each sub-topic is expressed as a set of documents.

$$D = \{s_i, s_j, s_k, \ldots\}$$
$$s_i = \{D_x, D_y, D_z, \ldots\}$$

Chapter 6

Experiment setup

6.1 System specifications

According to the time taken for executing certain tasks, the experiment is carried out in two different systems. All the code development, exploratory analysis, and statistical testing are performed on a HP ELITEBOOK system with a AMD Ryzen 5 PRO 4650U processor, 16 GB of RAM and 500 GB of disk space. Web scraping, IR system hosting, data storage, clustering and further benchmark testing are carried out on a large virtual machine (VM) which hosted at Fraunhofer FKIE. The technical specifications of the VM are four processor CPU (Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz), 48 GB RAM, and 370 GB disk space.

6.2 Testset description

For two main reasons, a dataset specific to this research problem is hard to find in the current IR data repositories. Foremost, the search query needs to be a phrase rather than a sentence. Furthermore, the documents need to be labeled with a specific intention rather than just coherence with the query. The interest at Fraunhofer FKIE is to retrieve the documents related to "Innovation and Technology", and a new testset is collected for this purpose. Below are a few specific areas of interest in news articles that describe the user intention: *Innovation, Technology breakthroughs, Future products, Applied research, New procurement strategies, Artificial Intelligence*. These topics are also described as positive document characteristics because a document is considered positive when it is strongly related to any one of the above-mentioned characteristics.

The strategy for the testset collection is to consider documents from lexical and semantic matching. The results from both algorithms help find the diverse contexts related to the user query. Therefore, a candidate pool with a maximum length of 30 documents is considered. Fifteen documents from both lexical and semantic matching are combined to a merged set where duplicate documents are removed. *Relevance labeling*, is a process to assign an appropriate label to the retrieval results inside the candidate pool. Every labeler has to assign a label not only coherent to the query but also considering the FKIE user's intention, i.e., coherence with positive document characteristics mentioned above. Once the labeler assigns a particular label to a document, labeled information is stored in an SQLite DB.

Table 6.1: Relevance labels definition and document distribution

Label-id	Label name	Document count	Label definition
1	Perfect	78	A document that strongly matches one of the positive document characteristics.
2	Partially relevant	147	A document that contains keywords and seems to be relevant, but still lacks innovation or novelty.
3	Irrelevant	306	A document containing the given user keyword still lacks innovation and coherent discussion about the query. Eg: click-baits, advertisements, etc,.
4	Wrong	98	These are false documents and have nothing to do with the user query.

A total of 22 queries are labeled from 5 different labelers. After analyzing the label distribution of the queries, it is observed that the perfect and partially relevant document distributions are very low compared to the other labels, shown in Table 6.1 on page 44. Five queries are removed from this testset due either no perfect documents or very low perfect and partially relevant documents. Rest of the 17 queries are considered for evaluation. Below table Table 6.2 on page 44 shows the queries and respective label details.

Table 6.2: Testset queries used for the evaluation

S No.	Query	Perfect	Partially relevant	Irrelevant	Wrong
1	Architekturanalyse	4	6	15	4
2	Big Data, KI für Analyse	7	11	10	2
3	Edge computing	1	5	11	11
4	IT-Standards	1	7	9	11
5	Kommunikationsnetze	4	5	18	1
6	Methode Architektur	4	8	15	2
7	Militärische Kommunikation	7	5	18	0
8	Mixed Reality	5	10	8	6
9	Quantentechnologie	3	4	16	1
10	Robotik	15	8	6	0
11	Satellitenkommunikation	2	6	21	1
12	Schutz von unbemannten Systemen	7	11	12	0
13	Visualisierung	5	4	10	11
14	Waffen Systeme	6	15	8	0
15	Wellenformen und -ausbreitung	4	13	8	5
16	militärische Entscheidungsfindung	1	6	20	3
17	unbemannte Landsysteme	2	10	1	17

6.3 Preprocessing for efficient IR

As described in the section 2.3, IR system setup consists of web scraping of news articles, filtering articles not related to technology and military, storing the articles in two different document indices to facilitate information retrieval. Sub-topic modeling works on this IR setup to extract a candidate pool and to select candidate keywords which are very crucial steps to generate a good diverse sub-topics. In both steps, a USE model is used to encode the documents and keywords to distributed semantic embeddings. It is observed in the manual analysis of sub-topic modeling that the time taken for encoding a text using an USE model is high.

– redis-db image

Due to this high encoding time, users interacting te search interface shall wait for a while to get the response from the IR system. In order to overcome this, a cache system is developed to store the embeddings from the USE model in a Redis-db. Below

6.4 Clustering evaluation

The main objective of this evaluation is to find parameters of the sub-topic modeling that create clusters which best represent the candidate pool and also heterogeneous (diverse). Due to the lack of cluster labels for the query in the testset, an alternative approach is adopted to evaluate the quality of clusters generated.

6.4.1 Intrinsic evaluation:

In case of no labeled data, the clustering output is generally evaluated using an intrinsic evaluation approach. Silhouette index is a metric used for evaluating the clustering performance and is calculated by using the intra-cluster and inter-cluster distances for each sample [59, 61]. Let us consider the testset T as a set of data points where each data point is denoted as x_i.

$$T = \{x_1, x_2, x_2, \dots\}$$

The testset T is clustered using a clustering algorithm resulting the data points segregated into groups. Let us consider the cluster set C where each cluster is denoted as c_i.

$$C = \{c_1, c_2, c_2, \dots\}$$

The silhouette index $s(x_i)$ for a data point x_i which is an element of cluster c_k is expressed as below.

$$s(x_i) = \frac{b(x_i) - a(x_i)}{max(b(x_i), a(x_i))}$$

$a(x_i)$ is the mean distance between the data point x_i and all other data points inside the cluster c_k. $b(x_i)$ is the mean distance between the data point x_i and all other data points in the nearest cluster c_l [61]. $a(x_i)$ and $b(x_i)$ represent the intra and inter cluster distances respectively. The silhouette score $s(x_i)$ is calculated over all the data points in the testset and the final mean is considered as the silhouette score S of the clustering. Silhouette score ranges from -1 to 1 and higher the score represents better clustering performance.

Silhouette index formula is represented for the data point x_i (highlighted in dark) in the figure Figure 6.1 where red lines represent the intra-cluster distances $a(x_i)$ and the blue lines represent the inter-cluster distances to the nearest cluster$b(x_i)$. In the best case scenario, the value of

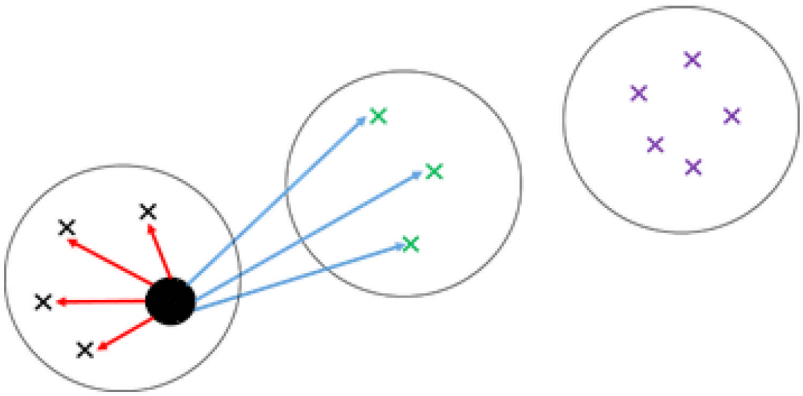


Figure 6.1: Visualization of silhouette index calculation [61].

$a(x_i)$ is very low in comparison to the value $b(x_i)$ which leads to very well-formed clusters. Therefore, the goal of achieving heterogeneous can be measured using the silhouette score.

6.4.2 Extrinsic evaluation:

To evaluate the quality of clustering with the help of document labels in the testset, a custom target function F is designed. Target function F tests the quality of clusters against the relevance labels from the dataset. The objective is to test whether the relevant documents are clustered into a similar cluster and the same case with irrelevant documents. Without any relation between clustering and relevance labeling, it can not be assumed that the positive and negative documents are clustered automatically because they cover a wide range of keywords in different domains. Therefore, it is more meaningful to evaluate the clustering for negative documents, i.e., Irrelevant and Wrong labeled documents. A target function is designed to address the number of negative documents isolated through sub-topic modeling.

The sub-topic modeling pipeline's output is distinctive clusters with a unique context, independent of relevance to user intention. However, the clusters can be divided into relevant and irrelevant clusters according to the relevance labels in the dataset. Let us consider that N_1, N_2, N_3, N_4 represent functions to get the number of documents in a single cluster with label-ids $1, 2, 3, 4$ respectively, as shown in Table 6.1 on page 44, and C represents the cluster set.

$$C = \{c_1, c_2, c_2, \dots\}$$

Relevant clusters C_r are clusters, that contain at least one document with label-id 1 or documents with majority of label-id 2. This can be determined using the below expression.

$$C_r = \{c_i \in C | (N_1(c_i) > 0) \vee (2 * N_2(c_i) >= (N_3(c_i) + N_4(c_i)))$$

With this expression, relevant clusters are differentiated from others and the focus is only on labels 1 and 2. The clusters that do not satisfy the above condition are logically considered irrelevant clusters.

$$C_i = \{c_j \in C \setminus C_r\}$$

The target function assesses the clustering with a ratio of documents in irrelevant documents to the documents in the candidate pool CP_q to a given user query q. Given N queries, the target function maps the score using the below equation. The function F ranges from 0 to 100 and higher score represents better separation of negative documents from positive documents. Therefore, indirectly representing the diversity of clustering results.

$$F = \sum_{i=1}^{N} (|C_i|/|CP_i|) * 100$$

6.4.3 Harmonic mean:

Both the target functions *Silhouette index* and F are used to tune the parameters of the sub-topic modeling pipeline. *Silhouette index* evaluates the clustering output and the custom target function F evaluates the distribution of relevant labels. In order to perform an automatic parameter selection, an objective function O is proposed which takes the harmonic mean of silhouette and target function scores. As the scores are on different scales, both are normalized to a range [0-1] using min-max normalization. Therefore, the O ranges from [0-1] and the clustering parameters which generate a higher score are considered as the final sub-topic pipeline parameters.

$$O = \frac{(2*S*F)}{(S+F)}$$

Table 6.3 on page 47 shows the parameters which will be tested during clustering evaluation.

Table 6.3: Parameters used in the pipeline for testing

S No.	Hyperparameters	Range
1	Candidate keyword selection	[10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100]
2	Reduced dimensions (UMAP)	[5, 10]
3	Min cluster size (HDBSCAN)	[20, 25, 30, 35, 40, 45, 50, 55, 60]
4	Min samples(HDBSCAN)	[1, 3, 5, 7, 10]

6.5 Survey evaluation

The sub-topics are assumed to help the user by retrieving the documents that are related to the query and sub-topics. This can be referred as the assumption of user satisfaction. This assumption can be evaluated by precision analysis when we have a dataset with two inputs rather one input. Two inputs here denote original query and an additional sub-topic. As Fraunhofer FKIE is keenly interested in a specific topics of technology and military and the user intention is also restricted to the innovation related theme, a manual evaluation with the help of a survey is chosen.

Two IR systems namely *System A* and *System B* are designed to represent the documents during retrieval when a query and sub-topic are provided. This survey evaluates the performance of these two IR systems. Template queries are used to semantically retrieve and rank the text documents. A template *"Innovation in **query** und **sub-topic**"* is used in the retrieval. The "query" and "sub-topic" are used as placeholders in the template and are dynamically replaced by actual values given by the user. In addition to the systems evaluation, the sub-topic modeling output is also tested by taking the user's feedback on the quality of sub-topics. Below are the two IR systems developed and tested in the survey.

1. **System A** is an IR system that retrieves documents from the sub-topic cluster chosen by the user and re-ranks the retrieved documents using the template similarity. Cosine similarity is used as ranking function.

   – system A and B image

2. **System B** is an IR system that retrieves documents semantically using a new search query generated by the template. As the documents are retrieved semantically there is no need to re-rank the documents again.

6.5.1 Survey questionnaire

A website is developed to conduct the survey and collect the data from participants. The feedback collected is stored in an SQLite DB. The participants of this survey are employees at Fraunhofer FKIE. The participants are invited through an email (not chosen by any particular means) and no personal user information is collected during the survey. Only a UUID is used to differentiate users when filling the survey concurrently. The user interface is developed on python, fastapi, bootstrap, HTML and javascript and deployed using docker on the VM.

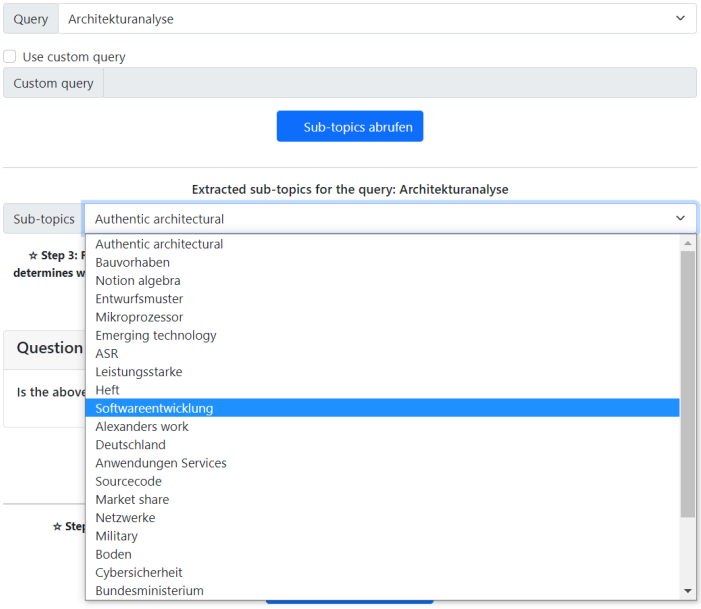


Figure 6.2: Extracted sub-topic list for a given user query

To evaluate the user satisfaction of the retrieved results and sub-topic modeling output, five questions are developed. Survey participants are requested to provide a query from either a drop-down list or a input text box. Subsequently, the sub-topics are retrieved during the runtime and displayed to the user in the form a list. Figure 6.2 shows the extracted sub-topic list for the query "Architekturanalyse". Participants are now asked to share feedback on the quality of this sub-topic list.

Following on the feedback, participants are requested to select a sub-topic and retrieve documents relevant to the query and the sub-topic. Figure 6.3 shows the retrieved IR system results for the query "Architekturanalyse" and the sub-topic "Military". Once again the participants are asked share feedback on the system results by rating the systems quantitatively. Lastly, an optional question regarding the significance of the whole sub-topic approach is given to the participants. Questionnaire used in the survey are briefly described below.

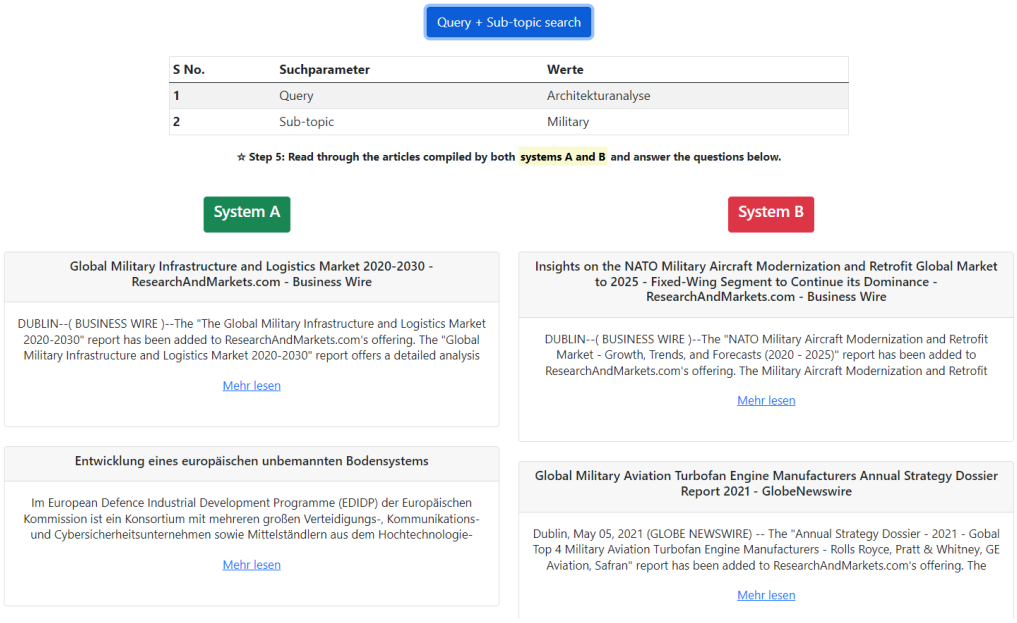


Figure 6.3: Extracted IR system results for a given user query and sub-topic

1. **Is the above sub-topic clustering output distinctive and well-labeled?**

   This question aims to evaluate the effectiveness of methodology tested in the master thesis i.e, well-formed heterogeneous clusters. Participants are requested to share feedback on the clustering output based on two characteristics namely distinctiveness and readability of the sub-topics. Distinctiveness describes the diversity of sub-topics and readability depicts the ease of understanding the sub-topic by its name or label. Below are the options provided to the survey participants and only one option must be selected.

   (a) Distinctive and well-labeled
   (b) Distinctive and not well-labeled
   (c) Not distinctive and well-labeled
   (d) Not distinctive and not well-labeled

2. **Which system results better represent the relevant news articles according to the given query and sub-topic?**

   Relevant news articles are retrieved text documents which have the positive document characteristics (mentioned in the section 6.2). Participants are asked to read the retrieved results from both IR systems A and B and then share the feedback accordingly. Below are the options provided to the survey participants and only one option must be selected.

   (a) System A
   (b) System B
   (c) Neither System A nor System B

3. **Rate the retrieval system A results for the given query and sub-topic (0-10).**

   Participants are requested to share a quantitative feedback by rating the system A results between 0 to 10. Ratings can take fractional values up to 1 decimal place, for example: a

user can provide 3.4 rating but not 4.65.The user ratings directly signify the user satisfaction corresponding to the retrieved results.

4. **Rate the retrieval system B results for the given query and sub-topic (0-10).**

   Participants are requested to share a quantitative feedback by rating the system B results between 0 to 10.

5. **Were the sub-topics helpful for you to find the relevant documents?**

   This question aims to evaluate the usefulness of the sub-topic modeling in IR. This answers the practicality of using the proposed methodology in this master thesis in a real-environment to find innovation related documents. Participants are requested to provide a boolean response of either Yes or No corresponding to the fulfillment of their expectation.

   (a) Yes
   (b) No

6.6 Precision evaluation

Assuming that the cluster labels are not very helpful to the user, the next evaluation technique shows that the clustering output does not deteriorate the performance of the retrieval results. The output of clustering is hard to examine with the baseline IR systems because the order of documents is missing and the performance metrics related to false positives are not addressed. For this purpose, we are extending the sub-topic creation with sub-topic ranking and document ranking. These two rankings help the existing pipeline to create a sequential order of documents and facilitate the evaluation of precision against the baselines. Therefore, this evaluation approach proposes eight different retrieval systems and evaluates the ranked results.

Table 6.4: Proposed IR systems for evaluation

IR system name	Sub-topic ranking	Document ranking
IR0	NA	Uniform distribution
IR1	NA	Query similarity
IR2	Query similarity	Query similarity
IR3	Template similarity	Template similarity
IR4	Document cardinality	Query similarity
IR5	(IR4, IR2)	Query similarity
IR6	(IR4, IR2, IR3)	Query similarity
IR7	Random combinations	Query similarity

The first system, *IR0*, is an arbitrary system where the positive documents are distributed uniformly on the ranking order. *IR1* system is simple query re-ranked results based on cosine similarity between the query and documents. The systems *IR2, IR3, IR4* are results of sub-topic pipeline clustering, where the clusters are first ranked, and later the documents are re-ranked with certain criteria. These three systems simulate the user reading the results linearly or in a sequence. In *IR2*, the sub-topic clusters are ranked by the cosine similarity between the query and centroid vector of the cluster and similarly for document ranking.

The system *IR3* uses a template similarity criteria, where the similarity is calculated between a template and centroid vector rather than the query. For example, the template string can be "Innovation and Technology". In the same way, *IR4* clusters are ranked using the number of documents in the cluster. The last system, *IR7*, is an unreal system just like the *IR0*, but multiple combinations of random ranking of clusters are considered to simulate the random selection of a sub-topic by the user and reading the documents in different sub-topics.

6.6.1 Combination systems

The systems *IR5* and *IR6* are produced by combining sub-topic rankings of other IR systems. The sub-topics are combined in such a way that the top ranking sub-topics are clustered together. This approach adopts the benefits of two ranking criteria into a ranking. As a part of the exploratory analysis, the potential of sub-topic ranking is explored with these combinations. This step is motivated from boosting technique in machine learning where a system of weak learners

–algorithm –figure

6.6.2 IR systems evaluation

In [46], a new evaluation measure for IR systems named expectation score is introduced. Expectation score (E) is similar to Precision (P) but does not consider false positives. E_k represents the number of positive documents at the index k, whereas P_k represents the ratio of positive documents at the index k to k. Similarly, mean expectation score (ME) is proposed to analyze the mean number of positive documents for N queries. This metric is specially useful to compare IR systems which retrieve relevant documents best at a given index k.

$$ME@k = (\sum_{i=1}^{N} E_i@k)/N$$

Furthermore, Mean Average Precision (MAP) [19] is used to evaluate the ranking performance. MAP is calculated through the Average Precision (AP) metric, which is an average of precision scores only at the positive document indices. Let us consider that G is a set of all positive document index with size g, the average precision and mean average precision is formulated as below. MAP is alone sufficient to compare different IR system and the system with highest MAP value is considered as the best IR system.

$$AP = (\sum_{i=1}^{G} P_i)/g$$
$$MAP = (\sum_{i=1}^{N} AP_i)/N$$

6.7 Evaluation summarization

The table Table 6.5 on page 51 shares the evaluation techniques chosen in this master thesis and the respective research questions answered.

Table 6.5: Proposed evaluation techniques

S No.	Evaluation type	Research questions addressed
1	Clustering analysis	**RQ1**
2	Survey Questionnaire	**RQ1, RQ2**
3	Precision analysis	**RQ3**

Chapter 7

Experiment results

7.1 Clustering results

As the testset contains 17 user queries and the clustering analysis is performed on each query and the output is analyzed on the mean values of clustering output. Two candiate pools are generated for each query with sizes 30 and 100 respectively. The small candidate pool *CP-30* and the large candidate pool *CP-100* represents the retrieved results of the original query from both syntactic and semantic matching. *CP-30* contains a maximum of 30 documents each document is labeled. *CP-100* contains around 100 documents and only few documents that are present in testset are labeled. Therefore, more than half of the *CP-100* documents are unlabeled. During the analysis the unlabeled documents are removed from the clustering output and only the labeled documents are used for evaluation. This assumes that the more documents present in the *CP-100* helps to provide better clusters compared to *CP-30*. Furthermore, it is expected to extract better sub-topics from the large collections in order to get deep insights of the data.

7.1.1 Clustering output analysis

Sub-topic modeling pipeline is tested on all the hyper-parameters of clustering that are mentioned in the section 6.4.3. Ideally there should be 1710 possible cases to test the clustering output for both small and large candidate pools. However few testcases have produced very low clustering (only one cluster) and are not to considered in the parameter selection. Ultimately 1588 and 1009 possible combinations of hyper-parameters are evaluated for large and small candidate pools respectively. The clustering output consisting the evaluation metrics are expressed as mean over all 17 queries. Furthermore, the analysis is shared separately for the small and large candidate pools. In the below analysis, "silhouette score" denotes the intrinsic evaluation metric and "targeted negative document ratio" signifies the extrinsic evaluation metric denoting the quality of separation between the relevant and irrelevant clusters.

HDBSCAN clustering does not need any parameter to specify the number of clusters created from the data. On the other hand, there are other parameters which needs to well chosen to generate better clusters. Before evaluating the parameters which significantly effect the clustering output, the relation between the evaluation metrics and number of clusters is analyzed. Figure 7.1 shows the relation between the silhouette score and the number of clusters. The data is generated when all the hyper-parameters possibilities of clustering pipeline are tested over the testset queries.

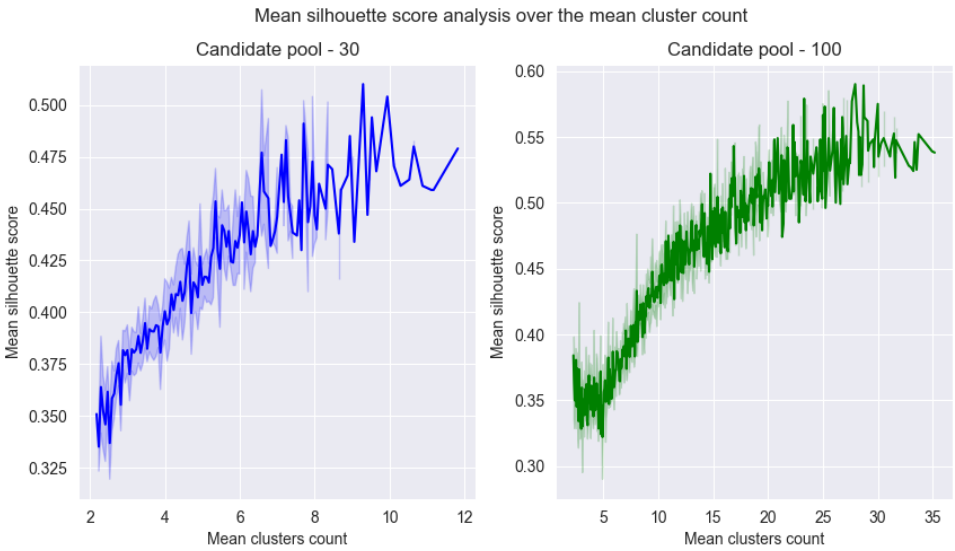


Figure 7.1: Silhouette score analysis over clusters count.

The objective of this analysis is to observe whether the number of clusters has an impact on the silhouette score globally. This means overall effect of clusters count, not specific to a particular query or a hyper-parameter set. It is clear from Figure 7.1, that a relatively high number of clusters are generating higher silhouette score implying better cluster quality. However a very high number of clusters in both small and large candidate pools (30 and 100) are showing lower silhouette score than the highest value. This signifies that the high number of clusters can possibly lead to many clusters with very few data points such that the quality of clustering is reduced, as the silhouette score is built on the intra- and inter-cluster distances. The possible better clustering can be achieved when the intra-cluster distance is minimum and inter-cluster distances are maximum. Therefore, a small number of clusters leads to overlapping clusters (very close) and therefore not a good choice for clustering parameter selection.

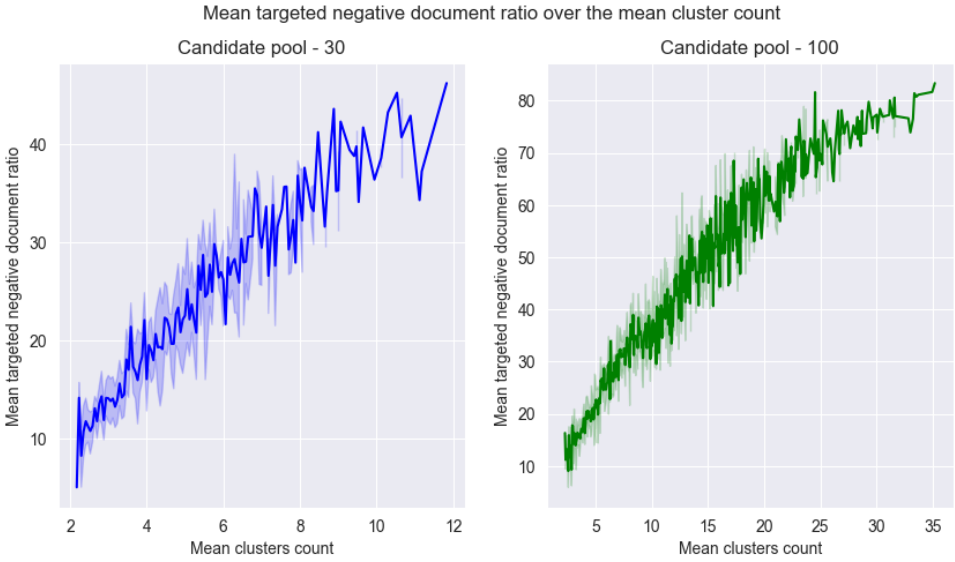


Figure 7.2: Targeted negative document ratio analysis over cluster count.

It is assumed to to derive the heterogeneity or diversity of clusters from the cluster quality and in order to have high diversity of clusters, the parameters which create a high number of clusters but not too many clusters must be selected. One major drawback in this analysis is

to combine all the clustering outputs for all 17 queries in the testset. Some queries generate low number of clusters and some generate high number of clusters. The combination of all queries into one analysis may mislead the analysis in some scenarios. As it is in the interest to observe the overall impact of number of clusters, the analysis can be further considered for the parameter selection without any issue. Table 7.1 on page 55 shows the pearson correlation over all the hyperparameters tested and there is a positive correlation between the clusters count and evaluation metrics chosen.

Table 7.1: Pearson correlation between the clustering observations

Parameter	CP-30	CP-100
Mean silhouette score and mean clusters count	0.71	0.89
Mean targeted negative document ratio and mean clusters count	0.76	0.94

Figure 7.2 shows the relation between the targeted negative document score and the number of clusters. Similar to the silhouette score, high cluster count positively correlates with targeted document ratio. The design of this evaluation metric has a slight inclination to support the clustering output with high clusters count, as it creates many clusters with low data points and leads to a greater separation between clusters. Consequently a low of number of clusters signify poor cluster quality due to lower separation between the documents. Therefore, to create a diverse set of clusters with well separation, this metric has to be relatively higher. Large candidate pool has better silhouette score and targeted negative document ratio compared to the smaller candidate pool. As the labels for all the documents in the large candidate pool are unknown, the actual scores could be even higher than the scores presented.

7.1.2 Candidate keyword selection (cks) analysis

This section explores the impact of cks parameter for keywords selection on the clustering output. This analysis is also very crucial to finalize the parameter selection at the end. The parameter cks ranges from 10 to 100 with values incrementing by 5. These values signify the percentile selection based on the query similarity. For example, a value of 40 denotes the selection of keywords that have similarity below the 40 percentile similarity and the removal of rest of the keywords. Similarly, a value of 100 denotes the selection of all keywords without any removal. The keywords selected using the cks parameter are further clustered using HDBSCAN. One of the main objectives of this master thesis is to test the effectiveness of clustering with keyword selection against the no selection. The data used in this analysis is generated from the clustering result for all the possible parameter combinations in the hyper-parameter set over the testset queries. The x-axis represents the candidate keyword selection and y-axis represents the mean of respective evaluation metrics.

Figure 7.3 presents the relation between the mean silhouette score and cks parameter. Results from the large candidate pool clearly show that a higher cks parameter generates better mean silhouette score. This signifies that lower selection of keywords is better and the best clustering outcome is achieved with no keyword selection (csk = 100). Small candidate pool results partly agree with this outcome, as the highest silhouette score is achieved when no keywords are selected. However, it is observed that the mean silhouette score is decreased at around a selection of 30 and 55. This denotes that there is a change in the structure of clustering output when certain keywords are selected. However there is no downtrend in the data implying that keyword selection does not generate better clusters and only an overall upward trend is observed.

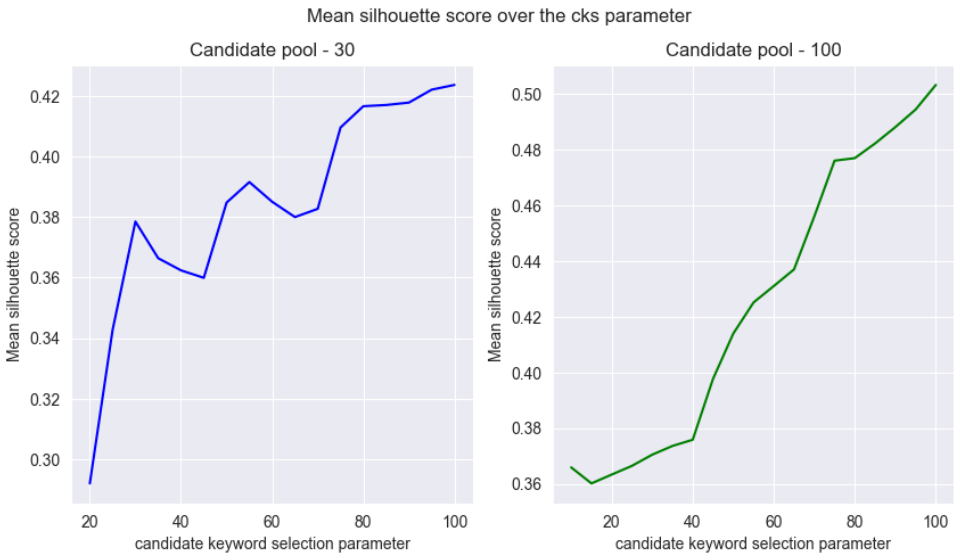


Figure 7.3: Silhouette score analysis over candidate selection parameter.

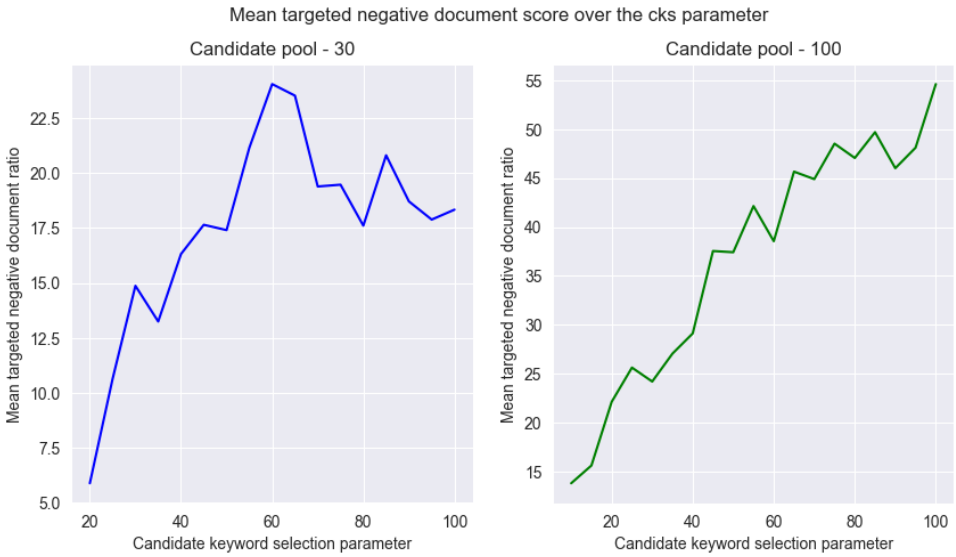


Figure 7.4: Targeted negative document ratio over candidate selection parameter.

Figure 7.4 presents the relation between the mean targeted negative document ratio and cks parameter. The metric here denotes the separation of clusters according to their similarities. The results from the small candidate pool show a highest separation of negative documents at csk = 60 and a downward trend for the values of csk higher than 60. This signifies that the keyword selection has positive impact on clustering achieving better separation. Large candidate pool results have a contrary outcome when compared to the earlier results. There is a clear upward trend implying that no keyword selection i.e csk = 100 has generated better separation of clusters. The results from the large candidate pool does not truly represent the nature of this evaluation metric as it ignores the unlabeled documents after clustering. When the labels of all 100 documents are considered, the results can be different.

Table 7.2: Mean of evaluation metrics over the csk parameter for the small candidate pool (30)

Csk parameter	Mean silhouette score	Mean targeted negative document ratio	Objective function score
85	0.42	20.81	0.48
75	0.41	19.48	0.46
60	0.39	24.06	0.45
100	0.42	18.34	0.45
95	0.42	17.89	0.45

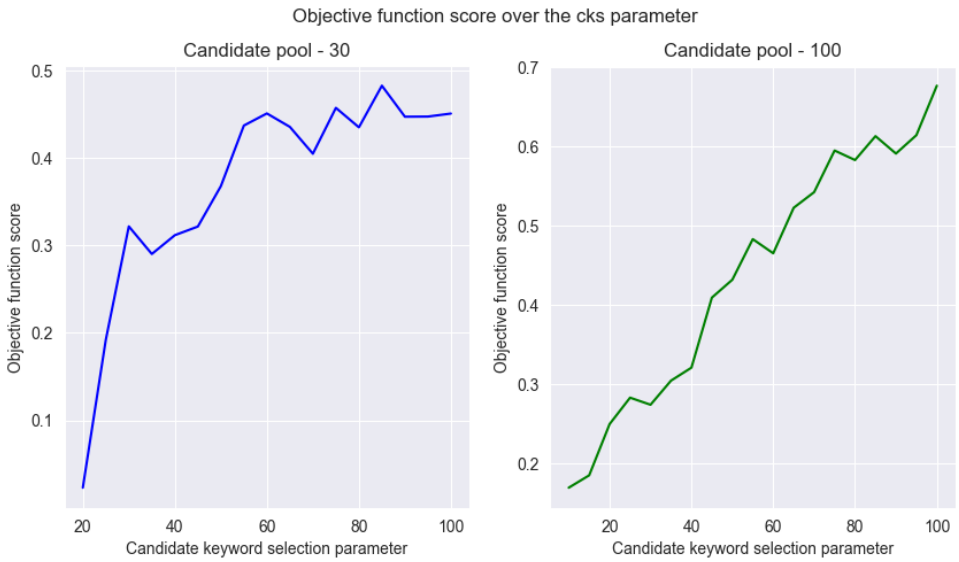


Figure 7.5: Objective function score analysis over candidate selection parameter.

The analysis results from intrinsic and extrinsic measures have shown a different patterns for small candidate pool and similar patterns for large candidate pool. Combining these two measures, a final objective function score which is the harmonic mean of normalized intrinsic and extrinsic measure is calculated. Figure 7.5 presents the relation between the objective function score and cks parameter. The objective function for the small candidate pool has achieved the highest score when the keywords selection is at 85. This shows that the selection of keywords below the 85 percentile similarity generates better clustering.

However, the top 15 percentile keywords may contain around 3 to 4 keywords or even less. Therefore the significance of clustering improvement between the csk values 85 and 100 is very minimal. Moreover, the keywords selection at 60 and 75 also generate very close results compared to the highest score at 85. Table 7.2 on page 56 shares the top-5 csk parameter results ordered by the objective function score for the small candidate pool. Although 85 percentile selection has the highest objective function score, there is no significant difference in the scores when compared with other percentile selections.

Table 7.3: Mean of evaluation metrics over the csk parameter for the large candidate pool (100)

Csk parameter	Mean silhouette score	Mean targeted negative document ratio	Objective function score
100	0.50	54.62	0.68
95	0.49	48.12	0.61
85	0.48	49.70	0.61
75	0.48	48.53	0.59
90	0.49	46.02	0.59

On the other hand, large candidate pool shows an increase in the objective function score with the large candidate keyword selection. Table 7.3 on page 57 shares the top-5 csk parameter results ordered by the objective function score for the large candidate pool. These results certainly convey that there is no benefit of keywords selection as the csk parameter = 100 has the highest evaluation scores.

7.1.3 Missed document analysis

The proposed methodology clusters keywords present in a document rather the documents itself. Documents are modeled as the set of clustered keywords. This leads to a soft-clustering output where a document is mapped to multiple clusters, as it can contain keywords from multiple clusters. The clustering algorithm HDBSCAN considers inherent noise present in the data and does not assign all keywords to a specific cluster. This results in creating noise keywords which does not belong to any cluster. If a document contains only keywords that are part of noise keywords, then the document is missed (removed) during the modeling process. It is very crucial to analyze the missed documents because if more documents are removed then the document modeling has no significance.

Table 7.4: Mean missed document values over the csk parameter

Candidate pool - 30

Csk parameter	Mean missed documents
20	0.94
40	0.75
60	0.40
80	0.18
100	0.09

Candidate pool - 100

Csk parameter	Mean missed documents
20	1.07
40	0.59
60	0.34
80	0.11
100	0.01

Table 7.4 on page 58 shows the average of mean missed documents over the candidate selection parameter. This includes all possible combinations of hyper-parameter set and the queries from tesetset. In case of small candidate pool, only one document is missed on the average for lower values of the csk parameter. Actual missed documents for certain queries might be higher the average scores, as only the mean is considered here. For large csk values, the mean missed documents is very small and no significant leak of documents during the sub-topic modeling. A similar pattern is observed in the large candidate pool where low values of csk parameter have high loss in documents. This data apparently signifies to avoid taking a candidate selection less than 40 percentile, as this can lead to possible loss of documents during modeling.

7.1.4 Parameter selection analysis

This section details the clustering performance for top-5 parameters when considered individually. In the earlier analysis, the results are grouped over the candidate keyword selection (cks) parameter. Table 7.5 on page 59 details the hyper-parameters and respective objective function scores for the small candidate pool. There is a huge variation in the parameter "minimum samples" in top-5 results. This parameter implies the minimum number of data points in a cluster. A value of 1 signifies that a minimum of one data point can create a cluster. This would definitely create a large number of clusters such that clusters have very few data points. This leads to poor clustering even though the evaluation metrics show better performance. The change in the cks parameter between the values 90, 95, and 100 will not have a major impact of the clustering output, as the number of keywords lie between 5 percentile selection can be very low.

Table 7.5: Objective score analysis over the parameters for the small candidate pool (30)

Csk parameter	Reduced dimensions	Minimum cluster size	Minimum samples	Objective function score
100.0	10.0	20.0	1.0	0.896
95.0	5.0	20.0	7.0	0.896
100.0	10.0	20.0	3.0	0.878
90.0	5.0	20.0	1.0	0.848
95.0	5.0	20.0	5.0	0.848

Table 7.6: Objective score analysis over the parameters for the large candidate pool (100)

Csk parameter	Reduced dimensions	Minimum cluster size	Minimum samples	Objective function score
100.0	5.0	20.0	10.0	0.964
100.0	10.0	20.0	10.0	0.946
100.0	10.0	20.0	5.0	0.941
100.0	10.0	20.0	7.0	0.939
100.0	5.0	20.0	7.0	0.935

Comparatively, the parameters such as reduced dimensions and minimum cluster size have very less variation. Table 7.5 on page 59 shares the similar data for the large candidate pool. From this data, a higher minimum samples of 10 is showing better clustering results, on contrary to the small candidate pool. In order to generate a diverse clustering output, the clusters size should be relatively higher rather than very small clusters. Small clusters generated when minimum samples of one or three data points can be considered as noise clusters and not desired. From both the candidate pools, a minimum cluster size of 20 and a reduced dimensions of 5 can be chosen for the final parameter selection. The final parameters selected after this analysis is shown in Table 7.10 on page 61

Table 7.7: Parameters selected after parameter selection analysis

Parameter	Value
Cks parameter	100
Reduced dimensions	5
Minimum cluster size	20
Minimum samples	10

7.1.5 Manual interpretation of results

After the parameters selected from observing the clustering performance for both small and candidate pool, a manual evaluation is performed on the large candidate pool. The clustering results provide deep insights over the candidate pool. However, it is also observed that the effectiveness of the selected parameters in the aspect of generating heterogeneous clusters is not achieved (not as expected). Main reason for the poor clustering results in diversity is due to the no keyword selection i.e, cks = 100. Generating deep clusters and also maintaining the cluster uniqueness is a challenging task. Evaluation metrics are not successful to capture this problem. A manual analysis is carried out on multiple clustering outputs with the parameters mentioned in Table 7.10 on page 61 and different values of cks parameter.

Table 7.8: Manual clustering output analysis for the Query "5G" with different cks parameters

Cks parameter	Number of cluster	Sub-topics (clusters)
100	34	5G, 5G network, Frequenzspektrum, 5G deployment, China, Defence intelligence, Nokia, Berlin, Mobilfunkmast, Telekom, Gigabit speed, 4G, WiFi, UMTS, Military, Fiberoptic broadband, 5G technology, Vodafone, Experimentation testing, Mobilfunknetzbetreiber, Technologie, Netzwerk, LTE, Smartphones, Netz, Aircraft, 5G service, Data cap, Deutsche Telekom, Telefónica, Apple, Gebiet, Netzausbau, Anbieter
75	23	Anstieg, TelekomKunden, Netzausbau, ATT, Mobilfunkmast, Military, Frequenzspektrum, Defense cybersecurity, Berlin, Nokia Ericsson, Experimentation testing, USA, Fiberoptic broadband, Smartphones, Technologie, 4G network, Carrier, Military aviation, ISP, WiFis reach, Upload speed, Apple, 5G telecommunication
50	14	Deutschland, Fortschritt, Telecom, Mobilfunkantenn, Wireless ISPs, Frequenzspektrum, 4G network, Military, Defense cybersecurity, Technologie, Netzausbau, Nokia Ericsson, Experimentation testing, Military aviation
25	10	Ausbau, Military aviation, Testing experimentation, Trump administration, Verizon wireless, SmartphoneNutzer, Telekommunikationsdienstleister, Berliner Hauptbahnhof, Frequenzspektrum, Technologieführerschaft

The results of one query namely "5G" is presented in the table Table 7.8 on page 60. The sub-topics generated when no keywords are selected are highly redundant and might not be useful to the users which have less significance. For example, the sub-topics "5G" and "5G network"

are very similar and might lead same set of documents. The objective of creating unique pathways from the user query to retrieved documents is not achieved. The 75 percentile similarity selection shows similar clustering output. The practicality of sub-topics in real-world is successful only when the clusters are unique and diverse. On the other hand, the clustering outputs for the cks parameter of 25 and 50 are unique, as we have removed a significant amount of keywords similar to the query are removed. However, there is a high chance of missing documents in this scenario.

Table 7.9: Results from manual cks parameter selection 0f 65 for different queries

Query	Number of cluster	Sub-topics (clusters)
5G	18	5G telecommunication, 4G network, Broadband, ATT Verizon, WiFis reach, Smartphone, Frequenzspektrum, Netzausbau, Fortschritt, Mobilfunkmast, Deutschland, Carrier, Telekommunikationskonzer, Experimentation testing, Technologie, Defense cybersecurity, Aircraft, Military
Quantentechnologie	23	Industriebetrieb, Quantum physic, DAQC, SuperRechner, Vorsprung, Datennetzwerk, Luft Raumfahrt, Forscher, Simulation, AI, Qubit, IBM, Datum, Department defense, AmazonGründer, Deutschland, Entwicklung, Military, Congress, Marineschiffbau, Forschung, Bundesministerium, China

In Figure 7.4, there is a downward trend in targeted negative document ratio at cks = 60 for the small candidate pool. This signifies that there is a better separation between the clusters at cks = 60 and then it decreases for higher cks values. However, we are manually analyzing the larger candidate pool results rather the small candidate pool. Despite this difference, the information from the small candidate pool can be partly helpful, as it uses the labeled information. Therefore, a 65 percentile similarity selection is chosen after a manual analysis on several queries. The results of manual parameter selection of cks = 60 is shared in the table Table 7.9 on page 61. There are still some redundant clusters at this selection, but the results are better than the higher selection values. The choice of this particular 65 percentile selection is not only due to the heterogeneity of clusters but also to have minimum loss of documents during clustering. Furthermore, the results from 65 percentile selection are evaluated during the survey feedback and the results are shared in upcoming sections.

Table 7.10: Final parameters selected after manual analysis

Parameter	CP-30	CP-100
Cks parameter	85	100
Reduced dimensions	5	5
Minimum cluster size	20	20
Minimum samples	10	10

7.1.6 Key statistics from clustering

After finalizing the parameters from Table 7.10 on page 61 and a change of 65 percentile selection for the large candidate pool and 85 percentile selection for the small candidate pool, an analysis is carried out on the cluster meta data. Three clustering observations are collected from the sub-topic modeling and presented in Table 7.11 on page 62. All the data presented below are calculated over 17 queries in the testset.

1. **Mean number of clusters** gives the average number of clusters/sub-topics after sub-topic modeling.
2. **Mean keywords size** gives the average number of keywords used during the sub-topic modeling.
3. **Mean cluster size** gives the average number of keywords per cluster.

Table 7.11: Keyword observations during clustering over 17 queries

Parameter	CP-30	CP-100
Mean number of clusters	7.06	19.41
Mean keywords size	423.59	1057.47
Mean cluster size	70.12	78.96

It is evident from the above table that the large candidate pool considers more data points for clustering and generates around three times more sub-topics compared to small candidate pool. This signifies that the user has the opportunity to view more diverse sub-topics and can access more documents. However, the average cluster size remains almost the same in both the candidate pools.

7.2 Survey results

1. **Question 1 results**
2. **Question 2, 3 and 4 results**
3. **Question 5 results**

7.2.1 Questionnaire results

7.2.2 Statistical analysis

7.3 Precision analysis results

7.3.1 Mean expectation score analysis

7.3.2 Mean average precision analysis

Chapter 8

Conclusion

8.1 Conclusion

8.2 Limitations

8.3 Future work

Bibliography

[1] U.S. Army CCDC Army Research Laboratory Public Affairs. *Army project may improve military communications by boosting 5G technology*. 2020. URL: https://www.army.mil/article/230198/ (visited on 11/26/2019).

[2] Maedeh Afzali and Suresh Kumar. „Text document clustering: issues and challenges". In: *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*. IEEE. 2019, pp. 263–268.

[3] Mebarka Allaoui, Mohammed Lamine Kherfi, and Abdelhakim Cheriet. „Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study". In: *Image and Signal Processing: 9th International Conference, ICISP 2020, Marrakesh, Morocco, June 4–6, 2020, Proceedings 9*. Springer. 2020, pp. 317–325.

[4] Giambattista Amati. „BM25". en. In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 257–260. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_921. URL: https://doi.org/10.1007/978-0-387-39940-9_921 (visited on 01/17/2023).

[5] Dimo Angelov. „Top2Vec: Distributed Representations of Topics". In: *CoRR* abs/2008.09470 (2020). arXiv: 2008.09470. URL: https://arxiv.org/abs/2008.09470.

[6] Hiteshwar Kumar Azad and Akshay Deepak. „Query expansion techniques for information retrieval: a survey". In: *Information Processing & Management* 56.5 (2019), pp. 1698–1735.

[7] Holger Bast and Ingmar Weber. „Type less, find more: fast autocompletion search with a succinct index". In: *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 2006, pp. 364–371.

[8] Slobodan Beliga. „Keyword extraction: a review of methods and approaches". In: *University of Rijeka, Department of Informatics, Rijeka* 1.9 (2014).

[9] Nicholas J Belkin et al. „Interaction with texts: Information retrieval as information seeking behavior". In: *Information retrieval* 93.55-66 (1993).

[10] Kamil Bennani-Smires et al. „Simple unsupervised keyphrase extraction using sentence embeddings". In: *arXiv preprint arXiv:1801.04470* (2018).

[11] Andrea Bernardini, Claudio Carpineto, and Massimiliano D'Amico. „Full-subtopic retrieval with keyphrase-based search results clustering". In: *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*. Vol. 1. IEEE. 2009, pp. 206–213.

[12] Tim Berners-Lee, Larry Masinter, and Mark McCahill. *Uniform resource locators (URL)*. Tech. rep. 1994.

[13] ST Bhosale, Tejaswini Patil, and Pooja Patil. „Sqlite: Light database system". In: *Int. J. Comput. Sci. Mob. Comput* 44.4 (2015), pp. 882–885.

[14] David M Blei, Andrew Y Ng, and Michael I Jordan. „Latent dirichlet allocation“. In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[15] Chris Burges et al. „Learning to rank using gradient descent“. In: *Proceedings of the 22nd international conference on Machine learning*. 2005, pp. 89–96.

[16] Ricardo JGB Campello et al. „Density-based clustering“. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10.2 (2020), e1343.

[17] Jefferson Rosa Cardoso et al. „What is gold standard and what is ground truth?“ In: *Dental press journal of orthodontics* 19 (2014), pp. 27–30.

[18] Daniel Cer et al. „Universal sentence encoder“. In: *arXiv preprint arXiv:1803.11175* (2018).

[19] Gordon V Cormack and Thomas R Lynam. „Statistical precision of information retrieval evaluation“. In: *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 2006, pp. 533–540.

[20] W Bruce Croft and Roger H Thompson. „I3R: A new approach to the design of document retrieval systems“. In: *Journal of the american society for information science* 38.6 (1987), pp. 389–404.

[21] Kushal Rashmikant Dalal. „Analysing the Role of Supervised and Unsupervised Machine Learning in IoT“. In: *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*. 2020, pp. 75–79. DOI: 10.1109/ICESC48915.2020.9155761.

[22] Christopher M De Vries, Shlomo Geva, and Andrew Trotman. „Document clustering evaluation: Divergence from a random baseline“. In: *arXiv preprint arXiv:1208.5654* (2012).

[23] Brian Dean. *Here's What We Learned About Google Searches*. [Accessed 25-Apr-2023]. 2020. URL: https://backlinko.com/google-keyword-study.

[24] TensorFlow Developers. *TensorFlow*. Version v2.8.2. Specific TensorFlow versions can be found in the "Versions" list on the right side of this page.
See the full list of authors on GitHub. May 2022. DOI: 10.5281/zenodo.6574269. URL: https://doi.org/10.5281/zenodo.6574269.

[25] Jacob Devlin et al. „Bert: Pre-training of deep bidirectional transformers for language understanding“. In: *arXiv preprint arXiv:1810.04805* (2018).

[26] Hai Dong, Farookh Khadeer Hussain, and Elizabeth Chang. „A survey in semantic search technologies“. In: *2008 2nd IEEE international conference on digital ecosystems and technologies*. IEEE. 2008, pp. 403–408.

[27] Yoav Freund et al. „An efficient boosting algorithm for combining preferences“. In: *Journal of machine learning research* 4.Nov (2003), pp. 933–969.

[28] Robert Gaizauskas and Yorick Wilks. „Information extraction: Beyond document retrieval“. In: *Journal of documentation* (1998).

[29] Jiafeng Guo et al. „A deep relevance matching model for ad-hoc retrieval“. In: *Proceedings of the 25th ACM international on conference on information and knowledge management*. 2016, pp. 55–64.

[30] Thorsten Joachims. „Optimizing search engines using clickthrough data“. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2002, pp. 133–142.

[31] Ashish Kankaria. „Query Expansion techniques“. In: 2015.

[32] Eric Kasten. „HTML“. In: *Linux J.* 1995.15es (July 1995), 3–es. ISSN: 1075-3583.

[33] Moaiad Ahmad Khder. „Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application.“ In: *International Journal of Advances in Soft Computing & Its Applications* 13.3 (2021).

[34] Oren Kurland and Carmel Domshlak. „A rank-aggregation approach to searching for optimal query-specific clusters“. In: *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 2008, pp. 547–554.

[35] Saar Kuzi et al. „Leveraging semantic and lexical matching to improve the recall of document retrieval systems: A hybrid approach“. In: *arXiv preprint arXiv:2010.01195* (2020).

[36] Alfirna Rizqi Lahitani, Adhistya Erna Permanasari, and Noor Akhmad Setiawan. „Cosine similarity to determine similarity measure: Study case in online essay assessment“. In: *2016 4th International Conference on Cyber and IT Service Management*. IEEE. 2016, pp. 1–6.

[37] Matthew Lavin. „Analyzing documents with TF-IDF“. In: (2019).

[38] Xiaoyong Liu and W Bruce Croft. „Cluster-based retrieval using language models“. In: *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. 2004, pp. 186–193.

[39] Xiaoyong Liu and W Bruce Croft. „Representing clusters for retrieval“. In: *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 2006, pp. 671–672.

[40] Batta Mahesh. „Machine learning algorithms-a review“. In: *International Journal of Science and Research (IJSR).[Internet]* 9 (2020), pp. 381–386.

[41] Francesca Maridina Malloci et al. „A text mining approach to extract and rank innovation insights from research projects“. In: *International Conference on Web Information Systems Engineering*. Springer. 2020, pp. 143–154.

[42] Gary Marchionini and Ryen White. „Find what you need, understand what you find“. In: *International Journal of Human [# x02013] Computer Interaction* 23.3 (2007), pp. 205–237.

[43] Leland McInnes and John Healy. „Accelerated hierarchical density based clustering“. In: *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2017, pp. 33–42.

[44] Leland McInnes, John Healy, and Steve Astels. „hdbscan: Hierarchical density based clustering.“ In: *J. Open Source Softw.* 2.11 (2017), p. 205.

[45] Leland McInnes, John Healy, and James Melville. „Umap: Uniform manifold approximation and projection for dimension reduction“. In: *arXiv preprint arXiv:1802.03426* (2018).

[46] Martin Mehlitz et al. „A new evaluation measure for information retrieval systems“. In: *2007 IEEE International Conference on Systems, Man and Cybernetics*. IEEE. 2007, pp. 1200–1204.

[47] Tomas Mikolov et al. „Efficient estimation of word representations in vector space“. In: *arXiv preprint arXiv:1301.3781* (2013).

[48] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. „Learning to match using local and distributed representations of text for web search“. In: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 1291–1299.

[49] nlpcloud.com. *Noun Chunks / Noun Phrases API*. 2020. URL: `https://nlpcloud.com/nlp-noun-chunks-noun-phrase-extraction-api.html` (visited on 12/01/2020).

[50] William S Noble. „What is a support vector machine?“ In: *Nature biotechnology* 24.12 (2006), pp. 1565–1567.

[51] Rodrigo Nogueira and Kyunghyun Cho. „Passage Re-ranking with BERT“. In: *arXiv preprint arXiv:1901.04085* (2019).

[52] Stanislaw Osinski and Dawid Weiss. „A concept-driven algorithm for clustering search results“. In: *IEEE Intelligent Systems* 20.3 (2005), pp. 48–54.

[53] Jeffrey Pennington, Richard Socher, and Christopher D Manning. „Glove: Global vectors for word representation“. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[54] *Performance Comparison of Dimension Reduction Implementations*. `https://umap-learn.readthedocs.io/en/latest/benchmarking.html`. [Accessed 25-Apr-2023].

[55] Matthew E. Peters et al. *Deep contextualized word representations*. 2018. arXiv: `1802.05365 [cs.CL]`.

[56] Wisam A Qader, Musa M Ameen, and Bilal I Ahmed. „An overview of bag of words; importance, implementation, applications, and challenges“. In: *2019 international engineering conference (IEC)*. IEEE. 2019, pp. 200–204.

[57] Nils Reimers and Iryna Gurevych. „Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks“. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: `https://arxiv.org/abs/1908.10084`.

[58] Shaurya Rohatgi, Jian Wu, and C Lee Giles. „PSU at CLEF-2020 ARQMath Track: Unsupervised Re-ranking using Pretraining.“ In: *CLEF (Working Notes)*. 2020.

[59] Peter J Rousseeuw. „Silhouettes: a graphical aid to the interpretation and validation of cluster analysis“. In: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.

[60] Reijo Savolainen. „Elaborating the conceptual space of information-seeking phenomena.“ In: *Information Research: An International Electronic Journal* 21.3 (2016), n3.

[61] Meshal Shutaywi and Nezamoddin N Kachouie. „Silhouette analysis for performance evaluation in machine learning with applications to clustering“. In: *Entropy* 23.6 (2021), p. 759.

[62] Aayush Srivastava. *DBSCAN Clustering Algorithm — blog.knoldus.com*. `https://blog.knoldus.com/dbscan-clustering-algorithm/`. [Accessed 27-Apr-2023].

[63] Ashish Vaswani et al. „Attention is all you need“. In: *Advances in neural information processing systems* 30 (2017).

[64] Jonathan J Webster and Chunyu Kit. „Tokenization as the initial phase in NLP“. In: *COLING 1992 volume 4: The 14th international conference on computational linguistics*. 1992.

[65] Yinfei Yang et al. „Multilingual universal sentence encoder for semantic retrieval“. In: *arXiv preprint arXiv:1907.04307* (2019).

[66] Meng Yuan, Justin Zobel, and Pauline Lin. „Measurement of clustering effectiveness for document collections“. In: *Information Retrieval Journal* (2022), pp. 1–30.

[67] Oren Zamir and Oren Etzioni. „Web document clustering: A feasibility demonstration“. In: *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. 1998, pp. 46–54.

[68] Rui Zhao and Kezhi Mao. „Fuzzy bag-of-words model for document representation“. In: *IEEE transactions on fuzzy systems* 26.2 (2017), pp. 794–804.

[69] Ying Zhao and George Karypis. *Comparison of agglomerative and partitional document clustering algorithms*. Tech. rep. MINNESOTA UNIV MINNEAPOLIS DEPT OF COMPUTER SCIENCE, 2002.

[70] Nivio Ziviani et al. „Compression: A key for next-generation text retrieval systems“. In: *Computer* 33.11 (2000), pp. 37–44.

[71] Keneilwe Zuva and Tranos Zuva. „Evaluation of information retrieval systems“. In: *International journal of computer science & information technology* 4.3 (2012), p. 35.