

RiskRadar: An NLP-Driven Summarization System for Query-Based Security Insights

by

Md. Shahanur Zilane
20301225
Mohammad Mushfiqur Rahman
20301022
Aniqa Ibnat Jisa
20201136
Qurratul Ayen Elma
20201121
Asaduzzaman Rifat
20301003

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science & Engineering

Department of Computer Science and Engineering
Brac University
October 2024

© 2024. Brac University
All rights reserved.

Declaration

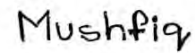
It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:



Md. Shahanur Zilane
20301225



Mohammad Mushfiqur Rahman
20301022



Aniqat Jisa
20201136



Qurratul Ayen Elma
20201121



Asaduzzaman Rifat
20301003

Approval

The thesis/project titled “RiskRadar: An NLP-Driven Summarization System for Query-Based Security Insights” submitted by

1. Md. Shahanur Zilane(20301225)
2. Mohammad Mushfiqur Rahman(20301022)
3. Aniqat Jisa(20201136)
4. Qurratul Ayen Elma(20201121)
5. Asaduzzaman Rifat(20301003)

Of Summer, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science & Engineering on October 20, 2024.

Examining Committee:

Supervisor:
(Member)



Dr. Muhammad Iqbal Hossain
Associate Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor 1:
(Member)



Dr. Jannatun Noor Mukta
Assistant Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor 2:
(Member)



Md. Faisal Ahmed
Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

The evolving complexity and frequency of cyber threat incidents demand the development of robust, user-friendly systems that can educate and assist users, and help them understand and mitigate them as much as possible. This thesis describes Risk Radar, a query-based information retrieval and response system that uses some advanced Natural Language Processing (NLP) methods to provide precise, context-aware responses to cybersecurity data.

The system employs a multi-module architecture, with each module tailored to a specific task, such as query correction, semantic sequence analysis, information retrieval, and response generation. The core NLP models used are BERT for semantic similarity, BM25 for effective retrieval of relevant content, and a distilled BART model for summarization and context-based response generation. A unique rule-based mechanism improves query understanding and maintains contextual continuity across user interactions, addressing the challenges of multi-turn dialogue in technical. The proposed system not only provides detailed responses, but it also includes relevant articles to help users better understand specific incidents or trends. The system's performance is measured by its ability to retain the context of user queries, retrieve and rank relevant content accurately, and generate coherent, informative responses.

The system's real-time implementation dynamically updates the dataset based on daily scraping of cybersecurity articles, ensuring that responses are timely and relevant. To address computational constraints, the model architecture prefers efficient methods like sequence-based rule application and DistilBART over more computationally intensive models like GPT-Neo. This trade-off balances accuracy and resource availability, resulting in a solution that is both practical and efficient. This thesis aims to contribute a scalable, efficient solution for tackling the growing need for real-time, user-oriented cybersecurity information systems.

Keywords: Natural Language Processing; BART; BERT; K-Means Clustering; Information Retrieval; Query Classification; Hybrid Models; BM25; Word2Vec; BERT; GloVe Embeddings; Text Generation; T5 Model; BART Model; Clustering; K-means; Autoencoders; Semantic Analysis; DistilBART

Dedication

We would dedicate this research to our beloved parents. Because without their consistent support and contributions, we would have never made it this far. They are our real heroes.

Acknowledgement

First and foremost, thanks to Almighty Allah for whom we could complete our thesis without any major interruptions. Secondly, a huge thanks to our respected supervisor Dr. Muhammad Iqbal Hossain Sir for being the best supervisor we could ever have. Sir has been extremely kind and understanding and has guided us through this whole research. A special thanks to Dr. Farig Yousuf Sadeque Sir for sharing his invaluable insights and advice on the NLP techniques we used. Thanks to our respected co-supervisor Jannatun Noor Mukta Miss for being with us and supporting us the whole time. Another big thanks to our second co-supervisor Md. Faisal Ahmed Sir, for his nonstop guidance and amazing ideas starting from the idea generation to completing the report. His guidance is really appreciated.

Table of Contents

Declaration	i
Approval	ii
Abstract	iv
Dedication	v
Acknowledgment	vi
Table of Contents	vii
List of Figures	ix
List of Tables	x
1 Introduction	1
1.1 Motivation	2
1.2 Research Problem	3
1.3 Research Objective	3
1.4 Thesis Outline	4
2 Literature Review	5
3 Dataset Description and Analysis	9
3.1 Data Description	9
3.1.1 Cyber Attack	9
3.1.2 Malware	9
3.1.3 Data Breach	10
3.1.4 Vulnerability	10
3.2 Data Collection	11
3.3 Category Construction & Data Preprocessing	11
3.4 Data Visualization & Analysis	13
3.4.1 Frequency of Categories	13
3.4.2 Average Content Length by Category	14
3.4.3 Average Word Distribution by Main Category	14
3.4.4 Distribution of positive and Negative Words	15
3.4.5 Distribution of Unique Initial Labels per Category	16
3.4.6 Time Series Plot of Categories	16
3.4.7 Top 20 Initial Labels by Frequency	17

3.4.8	Top Category and Initial Label in Each Year	17
4	Proposed Methodology	19
4.1	Risk Radar System	19
4.1.1	Module 1: Query Validation and Correction	19
4.1.2	Module 2: Query Sequence Maintenance	20
4.1.3	Module 3: Query-Based Document Retrieval and Re-ranking	24
4.1.4	Module 4: Query Classification Using GloVe Embeddings . . .	25
4.1.5	Module 5: Summarized Text Generation from the Top 10 Contents	26
4.1.6	Module 6: Response Generation Using DISTILBART Model .	27
4.1.7	Module 7: Link Generation and Detailed Summaries	28
4.1.8	Module 8: Response History Storage	29
4.2	Internal Architecture of Risk Radar	30
4.3	Proposed Model Architecture	30
4.3.1	Top 30 Content Fetch Based on Query	30
4.3.2	Summarized Text Generation from top 10 contents	31
4.3.3	Classification Model	33
4.3.4	Bart Training	34
5	Real-Time Implementation	36
5.1	Scraped Dataset and Pre-processing	36
5.2	Integration with Clustering Models	36
5.3	Merging with the Main Dataset	38
5.4	Improving Response Quality	38
6	Result Analysis and Comparison	40
6.1	Execution Time & Precision to fetch Top 30 Content	40
6.2	Query Classification Analysis	43
6.3	Text Generation from the Top 10 Contents	45
6.4	DISTILBART Train Result	47
6.5	K-mean Clustering	47
6.6	Response Result	51
6.7	Limitations	52
7	Conclusion	54
7.1	Further Research Areas	54
	Bibliography	59

List of Figures

3.1	Frequency of Categories	14
3.2	Average Content Length by Category	14
3.3	Average Content Length by Category	15
3.4	Distribution of positive and Negative Words	15
3.5	Distribution of Unique Initial Labels per Category	16
3.6	Distribution of Unique Initial Labels per Category	16
3.7	Top 20 Initial Labels by Frequency	17
3.8	Top Category and Initial Label in Each Year	17
4.1	Sequence capture	21
4.2	DISTILBART	28
4.3	Risk Radar System	30
4.4	Top 30 Fetch Content Based on Query	31
4.5	Text Generation from Top 10 Content Model	32
4.6	Classification Model	33
4.7	Bart Training	34
5.1	K-means clusterred autoencoder	37
5.2	Real-time dataset Implementation	38
6.1	Average Execution Time to fetch Top 30 Content	40
6.2	Model Precision to fetch top 30 content	41
6.3	Time Taken in Classification	43
6.4	Model Precision in Classification	43
6.5	Models Average Execution Time to Summarized	45
6.6	Flesch Reading Ease and Redundancy Score of Summarization	45
6.7	Average Summarized Content Similarity with Query	46
6.8	Training and Validation Loss per Epoch (BART)	47
6.9	Comparison of clustering model	48
6.10	K-Means on 1.3k real time Datapoint (4 clusters)	50
6.11	Average Response Time Comparison	51
6.12	Model Average Precision Resonse Comparison	51
6.13	GPT Table	52

List of Tables

3.1	Top 20 Categories and their Frequency	12
3.2	Category Mapping for Classification	13
4.1	Key Lists for Query Sequence Maintenance	20
6.1	Overall performance of models	42
6.2	Classification	44
6.3	Flesch Reading Ease and Redundancy Score of Summarization for both models	46
6.4	Training result	48
6.5	Comparison of clustering model	50

Chapter 1

Introduction

Today, sensitive information flows seamlessly at remarkable speeds, eradicating the need for cumbersome paperwork and dusty filing cabinets. Data can be transferred across the internet and securely stored in cloud systems with just a few clicks. However, this increased convenience carries a significant risk. As digital storage becomes the norm, it also creates new opportunities for cyber threats, where personal information, business confidential data, and financial details can be accessed, stolen, or exposed.

As the new technological advancements are flooding in, so does its risks. Nonetheless, a growing number of individuals do not actually know how to identify such risks nor what their implications are – and therein lies the cybersecurity awareness knowledge vacuum. This gap has led to an increased number of cases concerning ransomware, data breaches and other cyber crimes due to the widely held inexperience around securely shielding sensitive information. So its important we address this issue because the results of Cyber threats can be loseful for both individual as well organization. In this article, we present a smart query-based information retrieval system RiskRadar to bridge the knowledge gap.

Addressing this issue is crucial because the consequences of cyber threats can be devastating, both for individuals and organizations. To bridge this knowledge gap, we have developed Risk Radar, a smart query-based information retrieval system. Unlike traditional threat detection tools, Risk Radar functions as an interactive platform where users can ask questions about cyber threats, and receive explanations tailored to their level of understanding. By providing relevant articles, case studies, and examples of similar incidents, the system helps users grasp complex cybersecurity concepts—even if they have limited prior knowledge. This way, Risk Radar aims to empower users, making the digital world safer and more accessible to everyone, one query at a time.

According to the article[39] there was a 72% increase in data breaches from 2021 to 2023, which is a massive elevation. Security disrupting tactics are changing continuously. With our limited knowledge, we cannot tackle all emerging new problems. Risk Radar provides up-to-date information on a broad range of cybersecurity threats, continuously updating its knowledge base to include the most recent incidents.

Our query retrieval system consists of various ML and NLP techniques and models, such as BART for content summarization and text generation, BERT and BM25 for fetching top 30 content, GloVe is used for classification. Unsupervised models like K-means clustering are used for classification in real time based content retrieval. Additionally, our system consists of the Risk Radar website with a user-friendly interface to serve the users with their queries. The website names The Hacker News has been used to collect all the latest articles written on recent vulnerable incidents to form our main Dataset. The articles, being human-written, enhance the information's credibility to make our system more reliable.

Each output from Risk Radar will include some related articles to the user's query that the user can study and compare to evaluate better. Our system is targeted to improve the quality of life by making everyone aware of the surging cybercrimes and preventing danger at early stages. Our system uses four main labels for our dataset for this thesis: Cyber Attack, Malware, Data Breach and Vulnerability.

Unlike many existing tools that focus solely on technical threat detection, Risk Radar is user-focused, providing tailored educational content based on users' specific queries. It doesn't just identify threats; it helps users understand them with case-based examples and links to relevant incidents, making it easier for non-experts to grasp complex concepts. This makes it more of an educational tool rather than a purely defensive measure.

1.1 Motivation

In a world growing more and more dependent on digital platforms, we have seen both the enormous benefits and the hidden risks associated with this technological shift. From witnessing small businesses thrive online to the frustration of friends and family dealing with phishing scams, we have realized that cybersecurity is more than just a technical field; it is also a deeply personal one. The countless people suffering due to lack of knowledge could be at least made aware of the various repercussions. Therefore, we, along with our thesis supervisor, and co-supervisors, wanted to make the technical complex concepts of cyber security more understandable to everyday users, so that people with no technical background feel empowered to protect themselves.

One of the primary motivations for this research has been the need for real-time, accessible solutions that can rapidly adapt to changing threats. With cyberattacks becoming more sophisticated, simply reacting after a breach is almost insufficient. We need systems that can anticipate and respond to these threats as they arise, providing users with immediate answers. This belief is what motivated the creation of Risk Radar, a smart, real-time query retrieval and response tool. What distinguishes this system is its ability to provide instant feedback to users via a user-friendly interface, and its ability to continuously update itself and learn all the ongoing and most recent cyber threats so that our users do not fall behind. We want individuals to browse safely and comfortably rather than being in constant fear of cyber attacks, data breaches and phishing attacks.

1.2 Research Problem

In the current digital world there exists a huge gap of knowledge in cyber security. With the advancements in digital technology, a lot of our everyday processes has been digitized and made reliant to cloud or web-based system from storing information down to transacting businesses. One of the outcomes is a response to growing online convenience that in fact maybe convenient but at same time weakened security and privacy. Even though the usage of digital tools has been massive, people are apt in missing out how to identify and reduce any potential cybersecurity risks [31][34].

As indicated by the statistics above, this knowledge gap is tantamount to severe consequences. Users click mails or messages they have gotten which leads to someone sending them phishing or malware infections. Cyber hacks such as data breaches, phishing and ransomware continue to increase mostly because users do not know how to spot cybercrimes[33][36]. For example, the Fortinet 2024 Cybersecurity Skills Gap Report states that at least one security incident occurred in a given year for 87% of organizations and most are due to a complete lack of user awareness and even knowledge[19]. Additionally, research from ActZero found that small and medium-sized businesses (SMBs) which are less likely to have dedicated IT security resources compared with larger organizations were hit the hardest[32].

Additionally, as cyber security domain's information is very scattered, a query retrieval system gives access to all the data organized in one place. Reliable source articles are attached with the response which makes it way easier to access proper information. Additionally, individual summaries are presented which facilitates a user to get the idea of the resource without analyzing the whole document. This significantly saves the user's time to get their desired information. Dealing with these issues will necessitate the transition from a strictly technical focus towards an end user engagement model to equip users with learning and tools needed to navigate their own cybersecurity journey. In this work, we present the research study on developing Risk Radar a real-time query retrieval system built by using NLP techniques. As opposed to the conventional threat detection tools, Risk Radar was developed with an educational objective wherein users can enter their cybersecurity questions and have them answered individually and connected back up with real examples or case studies

1.3 Research Objective

. This research aims to develop Risk Radar, a query-based information retrieval and response system, designed to provide users with real-time information on emerging cyber threats. The system will utilize natural language processing (NLP) techniques to analyze user queries and retrieve relevant information about threats like cyber attacks, malware, data breaches, and vulnerabilities. Our approach includes gathering and incorporating recent articles from The Hacker News website into a dataset for training our models. The following are the study's goals:

1. Build a comprehensive dataset by scraping relevant, up-to-date articles from *The Hacker News*.

2. Build a robust query retrieval system for cybersecurity that can retrieve accurate and contextually relevant information in response to user queries.
3. Create a user-friendly interface to improve the user experience, facilitate access to cyber threat information, and simultaneously educate users.
4. Evaluate the system's performance in providing accurate and timely responses.

1.4 Thesis Outline

- **Chapter 2: Literature Review**
Analyzes existing research on cyber threat detection methods and query retrieval systems focusing on NLP-based approaches.
- **Chapter 3: Dataset Description & Analysis**
Provides a comprehensive overview of the data sources, collection process, category construction, preprocessing, and visualization to understand the nature of the data.
- **Chapter 4: Proposed Methodology**
Describes the design and implementation of the Risk Radar System and the architecture of models used for summarization and text generation.
- **Chapter 5: Real-Time Implementation**
Explains the process of keeping the dataset updated daily, merging new data, and potential scalability to include more data sources.
- **Chapter 6: Result Analysis and Comparison**
Analyzes the model performance, comparing different approaches, and highlights the strengths and weaknesses of each method used in the study.
- **Chapter 7: Conclusion**
Summarizes the research findings and suggests future directions, including expanding the dataset and improving model capabilities.

Chapter 2

Literature Review

This literature review explores key research in cybersecurity, information retrieval, and natural language processing (NLP), offering a clear summary of current progress, challenges, and gaps. The findings from these studies are integrated into the following sections to give a well-rounded view of the field.

Alawida et al. (2023) conduct a critical analysis of the dangers linked to large language models (LLMs), including ChatGPT, focussing on issues such as inaccuracies, biases, and plagiarism. It emphasizes ChatGPT's benefits in sectors like education and medical research, while also addressing ethical problems and privacy risks. The study highlights the significance of transformers and multi-head self-attention processes in addressing language issues, offering a detailed comprehension of ChatGPT's limits and possibilities in several domains (Alawida et al., 2023)[17]. Caldarini et al. (2022) similarly examine the constraints of chatbot technology, classifying them into rule-based and AI-based models. They advocate for enhanced evaluation frameworks to reconcile discrepancies between industry norms and innovations, particularly in generative models (Caldarini et al., 2022)[10].

Kumar and Sharma (2022) investigate hybrid optimisation methods such as IAOCOOT for effective text-based information retrieval. Their research illustrates the efficacy of ontology-based query extension, while highlighting the necessity for a more thorough examination of its constraints and opportunities for enhancement in query expansion methodologies (Kumar & Sharma, 2022)[22]. Qammar (2023) analyzes the cybersecurity dangers associated with chatbots such as ChatGPT, emphasizing new vulnerabilities like the generation of malicious code and phishing schemes. This study emphasizes the need for tailored countermeasures and research aimed at safeguarding sensitive information and preventing security breaches (Qammar, 2023)[26].

Shah (2017) examines the advancement of intelligent chatbots designed for educational applications, focussing on middle-class demographics. The research delineates the approaches employed in natural language understanding while acknowledging the necessity for more investigation into demographic problems and complications in NLP applications within education (Shah, 2017)[1]. Alqahtani et al. (2022) introduce the hybrid CMO-COOT technique for query

expansion in information retrieval systems. Their findings indicate enhanced performance in parameters like accuracy and F1-score, while also recognising the necessity for extensive validation and scalability of the technique (Alqah-tani et al., 2022)[9].

Zongxun et al. (2023) examine cyber threat intelligence analysis utilizing the K-CTIAA approach, which amalgamates knowledge graphs with BERT models. This approach exceeds traditional methods; nevertheless, further investigation is necessary to improve knowledge graph construction and optimize thresholds to address current shortcomings (Zongxun et al., 2023)[25]. The analysis by Hambarde (2023) evaluates dense retrieval models and knowledge distillation techniques in information retrieval, emphasizing shortcomings in semantic retrieval and the integration of traditional term-based systems (Hambarde & Proença, 2023)[21].

The research on cybersecurity knowledge graphs highlights the need for open-source datasets and flexible construction approaches for analyzing vulnerabilities and threats (OUCI, n.d.)[14]. Aleedy et al. conducted research on customer service chatbots using deep learning and Natural Language Processing (NLP) techniques where they analyzed collected transcripts. The case study developed by Leitao et al. (2019) underlines the importance of utilizing these technologies in real environments as well Full size image In addition, they point out the need perceptual metrics for estimating a conversational capability and show that existing performance measures (e.g., BLEU) are inadequate to evaluate chatbots (Aleedy et al., 2019)[6].

The study "Cyber Information Retrieval Through Pragmatics Understanding and Visualisation (2023)" emphasizes the crucial role of pragmatic understanding and visualization in enhancing information extraction. It also calls for the development of more user-centered search engines to improve the overall cybersecurity information retrieval process. Subsequent research ought to focus on improving scalability and adapting the search engine to incorporate supplementary datasets (Cyber Information Retrieval Through Pragmatics Understanding and Visualisation, 2023)[15]. Leiva et al. (2022) introduce the P-DAQAP platform for probabilistic query resolution in cybersecurity, showcasing its ability to handle incomplete and ambiguous data. There is a need to focus future studies on improving both sample method and explainability of data (Leiva et al., 2022)[12]. Nevertheless, more research is required for cooperative and application-level matters as well. Lin (2021) does indeed present a theoretical model on how dense and sparse retrieval approaches can be combined; but exact implementation details cannot be found there (Lin, 2021)[13].

Zhu et al. (2023) investigate the integration of large language models (LLMs) into information retrieval systems, emphasizing progress in query rewriters and retrievers. The study emphasizes scalability and computational efficiency as essential domains for further exploration (Zhu et al., 2023)[29]. Zhong et al. (2018) provide a cyber security data triage system designed to address the skill gap in Security Operations Centres (SOCs), emphasizing the necessity

for further discussion on scalability and the possible incorporation of machine learning techniques (Zhong et al., 2018)[4].

Supplementary assessments of papers provide improved comprehension. Wai Khin and Yee (2018) provide a query classification system that uses a NoSQL graph database to improve information retrieval; nonetheless, the system has challenges with extended search durations and typographical errors (Wai Khin & Yee, 2018)[2]. Hambarde and Proença (2023) present a comprehensive examination of progress in information retrieval, including dense retrieval models like BERT and hybrid methodologies, while emphasizing challenges in handling large documents due to processing costs (Hambarde & Proença, 2023)[21].

In their study from 2023, Li and colleagues present the SAILER model for legal case retrieval, showcasing its enhanced effectiveness while recognizing the computational and scalability challenges the model faces (Li et al., 2023)[24]. Ai et al. (2023) investigate difficulties related to dependability and growth, highlighting the potential of big language models for retrieving information (Ai et al., 2023)[16]. Kumar and Sharma (2023) pursues hybrid optimisation solutions for query expansion and reveals the computational complexity to be a major hindrance[23].

Zhu et al. (2023) provide a comprehensive overview of large language models in information retrieval, highlighting computational challenges and the need for improvements in long text processing[30]. Vuong et al. (2023) provide SM-BERT-CR for legal case retrieval, demonstrating enhanced performance while highlighting the significant computational expenses associated with its use (Vuong et al., 2023)[28].

Sun et al. (2023) present a comprehensive assessment of cyber threat intelligence mining for proactive cybersecurity defense, emphasizing the use of machine learning and natural language processing techniques such as BERT, while acknowledging persistent obstacles related to data quality and computational complexity (Sun et al., 2023)[15]. Hadi et al. (2023) investigate the significance of ChatGPT in cybersecurity, emphasizing potential privacy and bias concerns that require more examination (Hadi et al., 2023)[20]. Sánchez-Zas et al. (2023) provide an ontology-driven system for real-time risk management, emphasizing challenges related to scalability and computing demands (Sánchez-Zas et al., 2023)[27]. Li et al. Tan et al. (2024) also consider the advancement of generative models in IR, whilst acknowledging scalability and computational efficiency as serious limitations over established approaches (Li et al., 2024)[35]. Esteva et al. Advancements in computational efficiency and real-time updates are crucial, as demonstrated by CO-Search (Esteva et al., 2021[8]), a semantic search engine for COVID research.

Khin and Yee (2018) provide an online inquiry classification system aimed at enhancing content retrieval, although acknowledge its limits in addressing lengthy enquiries and typographical mistakes (Khin & Yee, 2018)[3]. Ram

et al. (2024) introduce a space-efficient framework for analyzing spacecraft health data using natural language queries, although encounter difficulties with the size of the training dataset and computational complexity (Ram et al., 2024)[38]. Muhammad et al. (2024) provide a hybrid query-based text summarisation method that improves information extraction, although running into issues with imprecise queries and computational complexity (Muhammad et al., 2024)[37].

This thorough analysis of the literature demonstrates important developments in cybersecurity, information retrieval, and natural language processing. The appraised studies contribute new ideas to improve these systems, in terms of effectiveness, reliability and security. However, all of these studies also reveal major limitations and challenges that need to be addressed -computational complexity, scalability, ethical considerations as well as clinical utility. These challenges point to exciting opportunities – not just lessons, but also visions of potential future work and development. This combination of massive language models, hybrid optimisation techniques and cutting-edge cybersecurity architectures is a game-changing innovation in the technical landscape providing unparalleled possibilities to excel. Now, researchers and practitioners are at the forefront of an era calling for creative solutions, and interdisciplinary collaboration initiatives to be global citizens committed to averting carnage. The road ahead will no doubt be fraught with hurdles, but it is also full of ways to continue learning and building smarter, safer systems.

Chapter 3

Dataset Description and Analysis

3.1 Data Description

Our dataset is categorized into four labels: **Cyber Attack**, which includes incidents involving hacking or network intrusion; **Malware**, focusing on harmful software; **Data Breach**, concerning unauthorized access to sensitive information; and **Vulnerability**, identifying weaknesses in software or systems that could be exploited.

3.1.1 Cyber Attack

Cyberattack refers to an attempt by cybercriminals to gain unauthorized access to corrupt or steal data from computer systems, networks, or devices. These attacks can include distributed denial of service (DDoS) attacks, ransomware, and phishing.

The 2017 WannaCry ransomware attack, which took advantage of an anomaly in Microsoft Windows, is an example. More than 230,000 computers across 150 countries were affected by the attack, which encrypted user files and demanded Bitcoin ransom payments in order to unlock them. Critical services were disrupted due to the enormous impact on individuals, businesses, and hospitals [18].

The increasing digitization of organizational processes has led to an increase in both the severity and frequency of cyber attacks. Cybercrime is a serious issue since it affects not only big companies but also small enterprises and private citizens. Cyberattacks can result in severe financial loss, harm to one's reputation, and delays to daily operations.

3.1.2 Malware

Malware, a shorthand for "malicious software," is software intended to compromise, damage, or access a computer system without authorization. It includes

ransomware, spyware, Trojans, worms, and viruses.

One such instance would be the Emotet malware, which started out as a banking Trojan before developing into a modular malware platform that could steal private data. Emotet has been used to spread other malware, which has resulted in widespread data theft and even monetary losses for companies [7].

Malware threats have increased in frequency due to the rise in remote work and increased online activity. Malware can easily infiltrate systems through phishing emails or compromised websites. Due to its tendency to hide for extended periods of time, it poses a serious risk and, if not discovered quickly, could have long-term consequences.

3.1.3 Data Breach

When unauthorized people obtain sensitive, private, or protected data, it's called a data breach. This could involve the disclosure of private data, including passwords, usernames, and financial records.

A popular example is the 2017 Equifax data breach, in which hackers gained access to 147 million people's personal data, including addresses, Social Security numbers, and birth dates, by taking advantage of a bug in a web application. Equifax suffered severe financial and legal repercussions as a result of the hack [5].

Data breaches are a serious concern due to the growing amount of personal and financial data stored digitally. In addition to endangering people's privacy, they also put businesses at serious risk of fines and damaged reputations. The rise in data breaches, which increased by 72% between 2021 and 2023, highlights the need for preventive measures and awareness tools like Risk Radar.

3.1.4 Vulnerability

A vulnerability is a weakness or flaw in software, hardware, or organizational processes that cybercriminals can exploit to gain unauthorized access or cause harm.

A practical example is the Log4Shell vulnerability discovered in December 2021, which affects the popular Apache Log4j library. This flaw enabled attackers to remotely execute code on affected systems, affecting millions of devices worldwide, including those from large corporations such as Apple, Microsoft, and Amazon [11].

Vulnerabilities are frequently the starting point for cyberattacks. In today's rapidly changing digital landscape, new vulnerabilities emerge as software and hardware systems become more complex. Organizations must be proactive in identifying and patching vulnerabilities to avoid becoming targets of cyber attacks. This is where tools like Risk Radar come in handy, as they keep

users informed of emerging vulnerabilities and assist them in taking preventive measures.

3.2 Data Collection

For this research, we gathered article data from The Hacker News[40] website to create a comprehensive dataset focused on cybersecurity incidents. Initially, we scraped 12,680 entries, each represented as a unique line in a CSV file. These entries included detailed information from the original articles covering a wide range of cybersecurity topics.

There were approximately 5000 distinct labels initially in the raw data, reflecting the variety of topics discussed in the articles. Each article had 2-3 labels assigned to it. To streamline the data, we first condensed these labels down to 1,280 by assigning the most relevant label out of the 2-3 labels for each article. Next, we applied an algorithm to analyze the frequency of each label. Thus, we could identify the most common and relevant categories to narrow down to four main labels for our dataset—Cyber Attack, Malware, Data Breach, and Vulnerability. The resulting dataset consists of the following columns:

- **SLno**: A unique serial number assigned to each entry for easy identification.
- **Title**: The headline of the article.
- **Link**: The URL linking to the original article on The Hacker News website.
- **Double_Category**: A field representing combined or closely related categories (2-3 labels).
- **Priority_Category**: The narrowed-down category assigned to each article out of the 2-3 available labels.
- **Final_Category**: The ultimate category assigned to each article among the four main labels.
- **Content**: The main body of the article, containing detailed information on the cybersecurity event or topic.
- **Date**: The publication date of the article, providing context for the timeline of the reported events.

The dataset has been further preprocessed and cleaned. Click here to access the dataset: <https://tinyurl.com/y4rta7rv>

3.3 Category Construction & Data Preprocessing

Several preprocessing steps were ensured to maintain the dataset’s quality and integrity. These steps focused on eliminating irrelevant data, standardizing

the content, and refining category assignments for a structured and reliable dataset.

The handling of missing and duplicate data was an essential part of the process. Any rows with null values—particularly in the *Final_Category* or *Content* fields—were removed, along with duplicate titles. This step ensured the dataset’s accuracy by eliminating incomplete or redundant entries.

A crucial refinement involved simplifying the multiple categories listed under *Double_Category*. Each entry originally contained 2-3 labels, so the most relevant category was identified and designated as the *Priority_Category* for each article. This step enabled the dataset to focus on the primary nature of the content. Further reduction was applied by analyzing the frequency of each label to generate a *Final_Category* column. The top 20 most frequent categories, listed below in descending order, were identified:

Category	Frequency
Cyber Attack	1,802
Vulnerability	1,548
Cyber Threat	1,407
Malware	1,232
Cybercrime	891
Data Breach	768
Mobile Security	762
Cyber Espionage	761
Data Security	587
Cybersecurity	572
Privacy	466
Cybersecurity Training	416
Software Security	411
Data Protection	401
Ransomware	377
SaaS Security	351
Network Security	317
Threat Intelligence	302
Social Engineering	293
IT Security	274

Table 3.1: Top 20 Categories and their Frequency

From these top categories, four were selected as *Final_Category* labels—Cyber Attack, Malware, Data Breach, and Vulnerability—due to their relevance and frequency. A thorough review of each article ensured that one of these four labels was assigned accurately to the corresponding content.

To safeguard the original dataset, a backup copy was created with only essential columns: *Title*, *Date*, *Link*, *Double_Category*, *Final_Category*, and *Content*. This backup ensured that any preprocessing changes would not impact the original data.

Some articles were either too short (1-2 words) or excessively long (up to 33,578 words). To ensure meaningful analysis, a minimum word count threshold of 200 was established. Articles failing to meet this requirement were discarded, while longer ones were retained for further processing.

For efficient machine learning classification, each category was mapped to a numerical ID as follows:

Category	ID
Cyber Attack	0
Malware	1
Vulnerability	2
Data Breach	3

Table 3.2: Category Mapping for Classification

This mapping simplified the classification process by making the data more accessible to machine learning algorithms.

A text cleanup function was implemented to standardize the content. This function removed non-word characters (e.g., slashes, commas, extra spaces), converted text to lowercase for uniformity, and eliminated URLs to maintain focus on textual data. The cleaned text was saved in a new column named *Clean_Content*, and the *Double_Category* field was renamed to *Initial_Label* for further training purposes.

Initially, the dataset consisted of 12,679 entries. After applying these pre-processing steps, the dataset was refined to 12,218 entries, ensuring a well-structured foundation for training the risk detection and response system, *Risk Radar*.

3.4 Data Visualization & Analysis

3.4.1 Frequency of Categories

Frequency of Categories diagram demonstrates the bar chart representation of how frequent each of the labels are in our preprocessed dataset: 4659 cyber attacks, 3610 vulnerabilities, 2524 data breaches, and 1315 malware attacks.

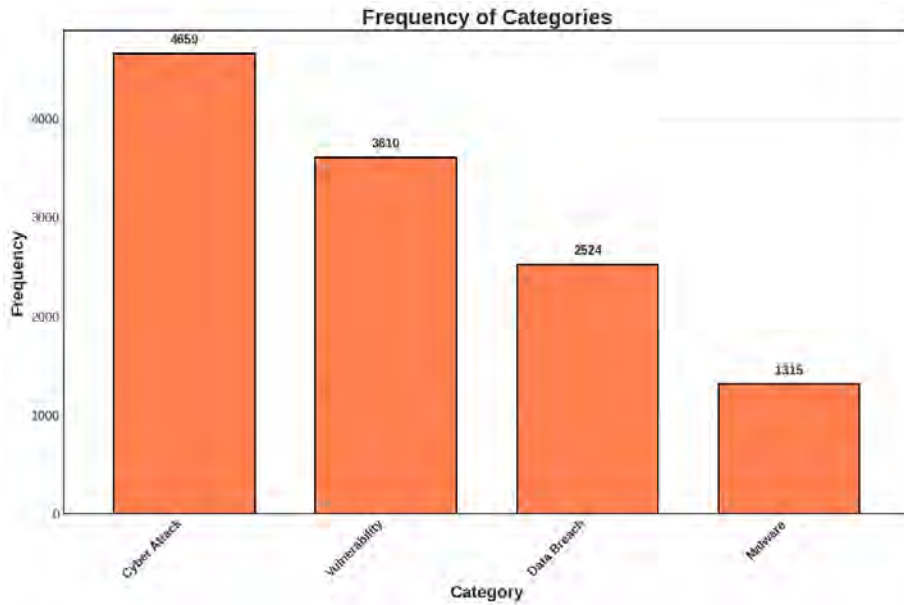


Figure 3.1: Frequency of Categories

3.4.2 Average Content Length by Category

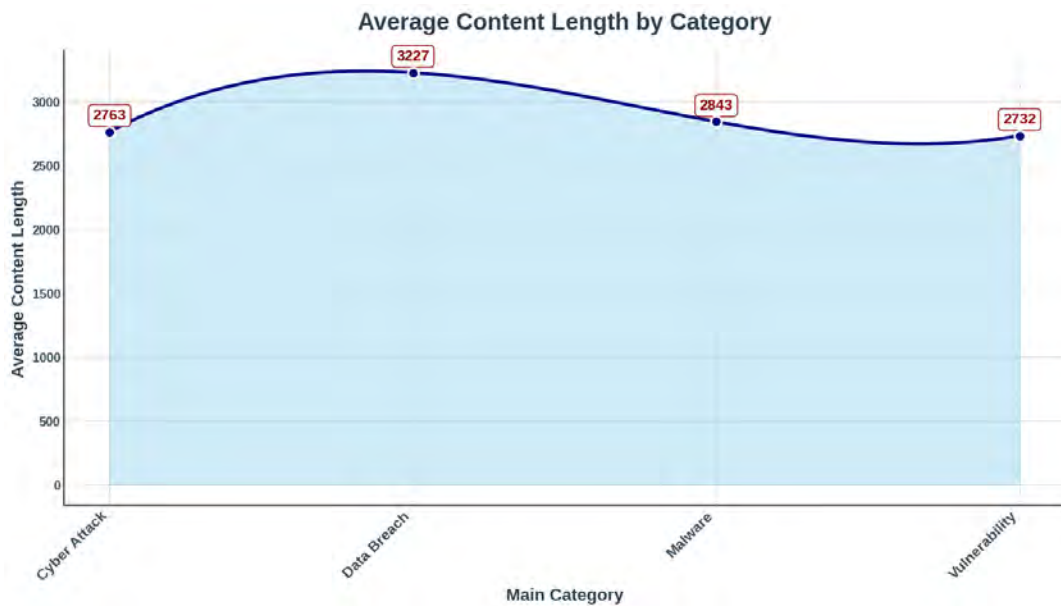


Figure 3.2: Average Content Length by Category

The average content length of Data Breach is the highest because the contents with the maximum lengths are classified as data breach, while that of Vulnerability is the lowest in our dataset

3.4.3 Average Word Distribution by Main Category

The average word distribution is also highest for Data Breach and lowest for Vulnerability.

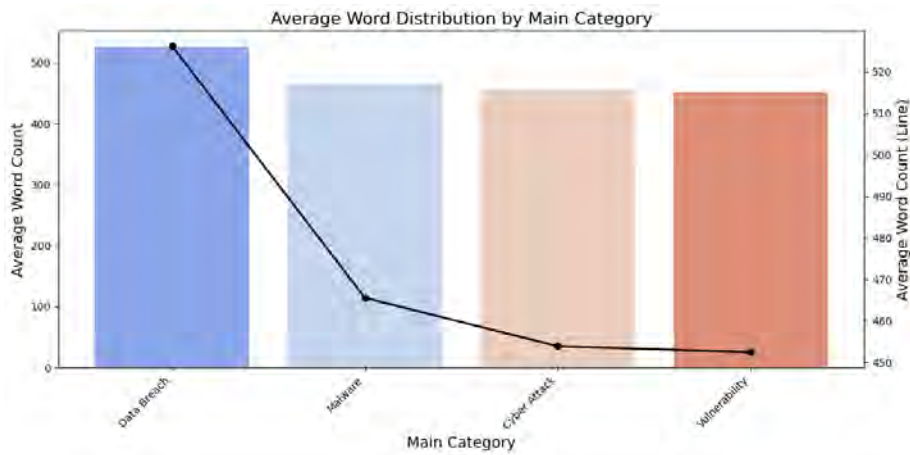


Figure 3.3: Average Content Length by Category

3.4.4 Distribution of positive and Negative Words

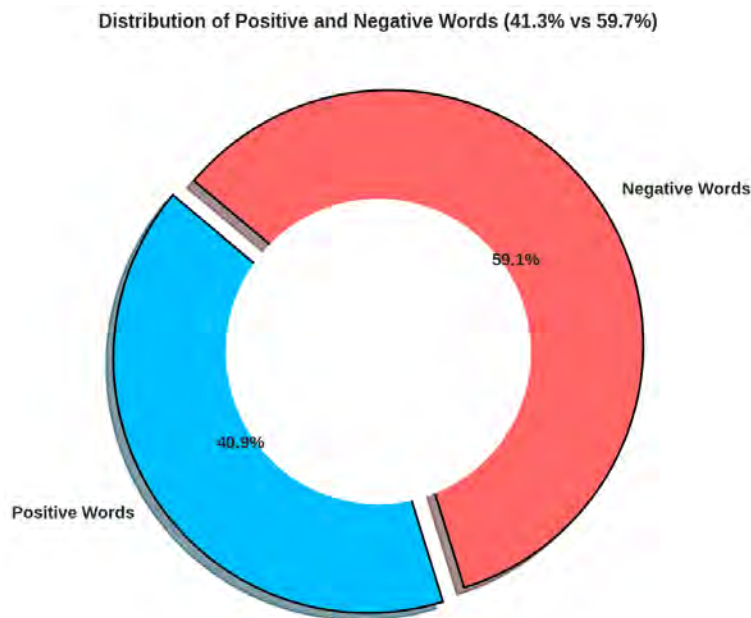


Figure 3.4: Distribution of positive and Negative Words

The distribution of negative words is significantly higher than the distribution of positive words because our dataset contains articles on cyber crimes, attacks, and vulnerabilities. The cyber threat occurrences and discussions often center around identifying and mitigating risks, which involves terminology related to negative events (e.g., breaches, attacks, malware).

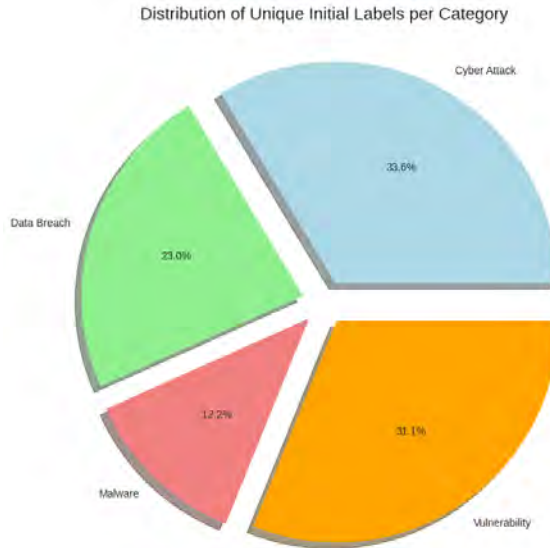


Figure 3.5: Distribution of Unique Initial Labels per Category

3.4.5 Distribution of Unique Initial Labels per Category

Initially, we had 4212 grouped labels(‘Double_Category’) among which Cyber Attack makes up the highest which is 33.6% while Malware make up the lowest which is 12.2%. This indicates that mitigating incidents related to cyber attacks should be our top priority, followed by vulnerability, data breaches, and malware.

3.4.6 Time Series Plot of Categories

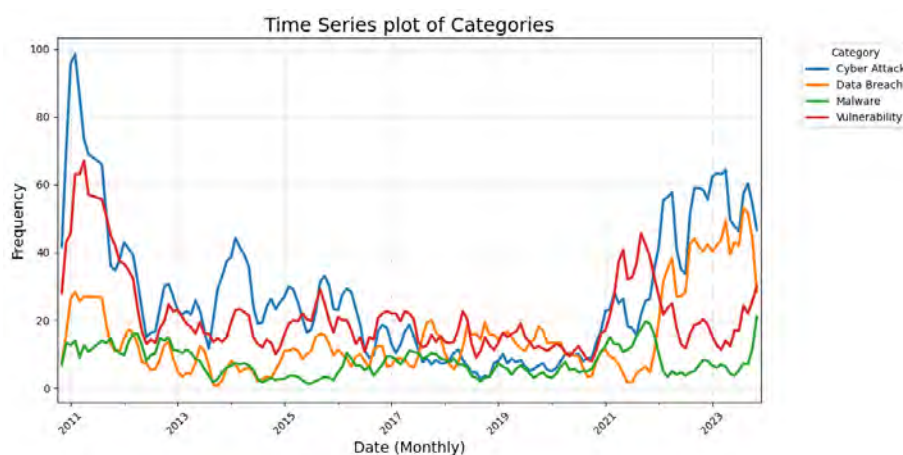


Figure 3.6: Distribution of Unique Initial Labels per Category

This figure demonstrates the frequency of each of the labels changed over time. It plots the monthly frequencies of each year starting from 2011 to 2023. Cyber attacks have mostly been significantly higher than the other labels. In 2023,

cyber-attacks are still the highest, followed by data breaches, vulnerabilities, and then Malware incidents.

3.4.7 Top 20 Initial Labels by Frequency

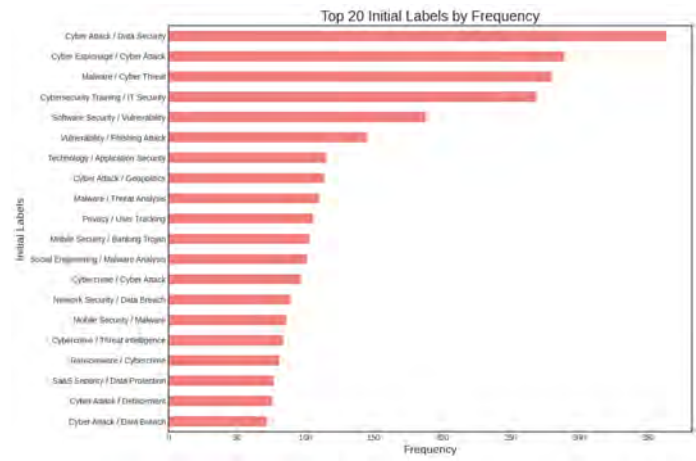


Figure 3.7: Top 20 Initial Labels by Frequency

This figure shows the top 20 initial labels from the “Double_category” column of our dataset. Initially, each content had 2-3 labels which is mentioned in the previous section. The reason for categorizing one article content with 2-3 labels is that one content can be based on more than one category as cyber security incidents often represent multiple threats or consequences in one incident. Hence, labeling with one true category might not always be efficient enough.

3.4.8 Top Category and Initial Label in Each Year

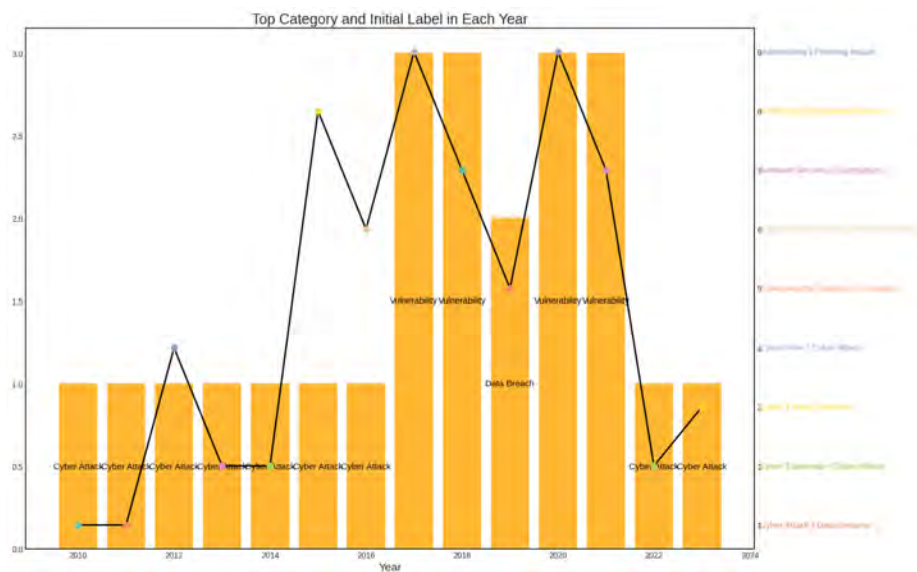


Figure 3.8: Top Category and Initial Label in Each Year

This figure illustrates the yearly occurrences of each of the labels from 2010 to 2023, the dotted points representing the maximum initial labels and the bar charts representing the final label. Cyber attack was maximum during years 2010 to 2016 while years 2017, 2018, 2020, and 2021 saw massive outbreaks of Vulnerability incidents.

Chapter 4

Proposed Methodology

4.1 Risk Radar System

We have divided our system into 8 modules, each of the modules focusing on a specific function.

4.1.1 Module 1: Query Validation and Correction

After a query is submitted, the system assigns a unique ID to the user and records the session's date and time. At the beginning of the session, two variables are initialized:

- 'previous_query': An empty string.
- 'active_query': An empty string, which will later store the user's input.

Once the user provides a query, it becomes the value of 'active_query', serving as the input for this module. In this module, a Python-based language tool configured for US English is used to handle grammar and spelling corrections. The 'correct_grammar_spelling' function adjusts any misspellings or grammatical issues in the user's query, ensuring clarity and correctness. The 'analyze_query' function checks if the query is written in English and whether the sentence structure is valid. If the function fails to detect the English language, it throws an exception with the message: "Could not detect language. Please use English words."

Once the analysis and corrections are complete, the refined query is assigned to the 'current_query' variable, which is then passed to the next module for further processing.

4.1.2 Module 2: Query Sequence Maintenance

In this module, we ensure that the relationships between consecutive queries are correctly identified and maintained. In order to make sure users get unique responses, contents, and links for similar consecutive queries, the sequence needs to be captured and maintained. This helps in understanding the context of user queries over a session, allowing the system to address references and dependencies between queries effectively. To achieve this, we utilize four key lists:

List Name	Description
Pronouns List	Contains pronouns to help identify references to subjects in a query, enabling the system to replace pronouns with the appropriate keywords from previous queries.
Descriptive Words List	Includes adjectives and descriptive words that provide context, such as “low,” “high,” or “danger,” helping the system to understand the nature of the query better.
Irrelevant Words List	Contains words that should be ignored when determining the main keywords of a query. For example, in “Give an example of a cyber attack,” the word “example” should not be treated as a primary subject, while “cyber attack” should be.
Multiple Word Expression List	Consists of compound words or phrases related to cybersecurity (e.g., “cyber security”). This list ensures that such terms are treated as a single keyword instead of separate words, preserving their meaning.

Table 4.1: Key Lists for Query Sequence Maintenance

Using lists allows our module to understand relationships between queries more efficiently. Although a dataset could be used for this purpose, we opted for lists due to their flexibility and simplicity. For the irrelevant words and multiple word expressions (MWE) lists, we kept the entries limited, while the pronouns and descriptive words lists were more extensive. We utilize the Spacy library for part-of-speech (POS) tagging to identify pronouns and track dependencies between words, helping to capture the main topics. However, Spacy struggles with multi-word expressions and sometimes identifies unnecessary words. Hence, we have used our irrelevant words and multiple word expressions (MWE) lists to refine the analysis. We developed some rules to

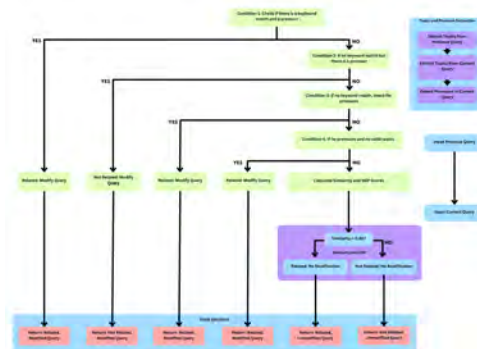


Figure 4.1: Sequence capture

help identify whether a query is related to a previous one and to maintain the continuity of the session. Here are three of them:

Condition 1

– Example 1:

* Input:

previous_query: "importance of cyber security and data breach?"
current_query: "what is cyber security. how can we overcome it?"

* **Analysis:** Since the current query’s keywords (“cyber security”) match those in the previous query (“cyber security”, “data breach”) and contain a pronoun (“it”), the system identifies them as related. The pronoun “it” is replaced with the keyword of the current query which is “cyber security.”

* Output:

previous topic: "cyber security," "data breach"
current topic: "cyber security"
pronoun: "it"
related: Yes
modified_query:
"what is cyber security. how can we overcome cyber security?"

– Example 2:

* Input:


```
previous_query: "importance of cyber security and data breach?"
current_query: "what is cyber security and cyber attack.
how can we overcome that?"
```

* **Output:**

```
previous topic: "cyber security," "data breach"
current topic: "cyber security," "cyber attack"
pronoun: "that"
related: Yes
modified_query: "what is cyber security and cyber attack. how can
we overcome cyber security and cyber attack?"
```

Condition 2

– **Example 1:**

* **Input:**

```
previous_query: "importance of cyber security and data breach?"
current_query: "what is cyber attack. how can we overcome that?"
```

* **Analysis:** Since the current query's keyword ("cyber attack") does not match any keywords in the previous query, they are treated as unrelated. However, the pronoun "that" is replaced with "cyber attack" which is in the current query and passed into the modified query.

* **Output:**

```
previous topics: "cyber security," "data breach"
current topics: "cyber attack"
pronouns: "that"
related: No
modified_query:
"what is cyber attack. how can we overcome cyber attack?"
```

If not related, previous_query is reset to an empty string.

Condition 3

– **Example 1:**

* **Input:**

```
previous_query: "give me an example of malware and data breach?"
current_query: "what are the dangers of them?"
```

* **Analysis:** If the current query does not have any keyword/topic and there is a pronoun in the query, then the pronoun is replaced by the previous query's topic and concluded as related.

* **Output:**
previous topic: "malware," "data breach"
current topic:
pronoun: "them"
related: Yes
modified_query:
"what are the dangers of malware and data breach?"

Condition 4

If there are multiple keywords in the previous query and no keywords or any pronoun in the current query, then the queries are related.

– **Example 1:**

* **Input:**
previous_query: "tell me about malware and data breach?"
current_query: "give me an example?"

* **Output:**
previous topic: "malware," "data breach"
current topic:
pronoun:
related: Yes
modified_query:
"give me an example of malware and data breach?"

Condition 5

If the previous query's keywords and current query's keywords don't match and the current query has no pronouns, then the queries are not related.

– **Example 2:**

* **Input:**
previous_query: "tell me about malware and data breach?"
current_query: "give me an example of cyber security?"

* **Output:**
previous topic: "malware," "data breach"
current topic: "cyber security"
pronoun:
related: No
modified_query:
"give me an example of cyber security?"

The outcome of Module 2 is a modified query, with continuity maintained between related queries. This output is then used as the input for Module 3.

We have used rules because writing rules is simpler than building and training a dataset-based model, which could still be prone to errors. Additionally, using an API for this would be costly, whereas our rule-based approach is more economical. Also, this module is adaptable to other domains, such as the medical field, by simply changing the Multiple Word Expression List (MWE). It saves time as no retraining is needed. Rules can be added at any time without retraining, and the response time is faster than a trained model.

4.1.3 Module 3: Query-Based Document Retrieval and Re-ranking

In this module, we utilize BM25 and BERT to efficiently retrieve and rank documents based on user queries. This dual approach combines the strength of BM25 for initial term-based retrieval with BERT’s ability to capture deeper contextual relationships between queries and documents, providing accurate and relevant results.

1. **BM25: Term-based Document Retrieval:** The BM25 algorithm is used to search through the dataset and rank documents according to their relevance to the modified query. It operates like a search engine and is particularly effective for initial document narrowing.

The modified query from Module 2 enters as input. BM25 then searches through the "Clean_Content" field in our dataset, which consists of approximately 12,218 data points. Each document is tokenized using the `simple_preprocess` function. The tokenized corpus is passed into a BM25 object, named `bm25kapi` and it runs the entire corpus at once when the system is initialised. Then BM25 ranks the documents by comparing the tokens of the modified query with the tokens of each document.

A ranked list of documents, including fields "Title", "Clean_Content", "Link", "Initial Label", and "Category ID" is output which is then passed to BERT.

2. **BERT: Context-based Re-ranking** To refine the initial term-based ranking from BM25, we use BERT to evaluate the contextual relationship between the query and the documents, bringing the most contextually relevant documents to the top of the list.

The "Clean_Content" of the documents is tokenized and passed into the BERT model, which generates embeddings for the text. BERT compares the embeddings of the query with those of the 30 top-ranked documents retrieved by BM25. This comparison allows BERT to understand the deeper context of the documents beyond mere term matches. BERT re-ranks the documents and assigns a similarity score to each one, ensuring that documents with a higher contextual alignment to the query appear

at the top.

One challenge we faced was the repetition of similar articles when multiple related queries were made. To address this, we implemented a solution to manage the selection of top-ranked articles:

For Unrelated Queries: When queries are unrelated, the system fetches 30 documents and appends the titles of the top 10 documents to an initially empty list. If a new query is unrelated, the previous list is cleared, and the process starts over with a new set of titles.

For Related Queries: For related queries, the system checks the existing list of titles containing the previous 10 results. After fetching 30 new documents, it verifies if any of the top 10 titles from the new results match those already in the list. If a match is not found, it simply appends the 10 titles to the list. If a match is found, the duplicate titles are removed, and the remaining titles are fetched from the 20 other documents and appended to the list.

Advantages:

- The list maintains that repetitive articles are avoided when queries are related.
- The combination of BM25 and BERT provides a more accurate and context-sensitive ranking where BM25 efficiently narrows down the documents using keyword matching and BERT enhances this by understanding the deeper meaning of the text.
- BM25 builds the tokenized corpus once for all the data points, which reduces time complexity for subsequent queries, making the retrieval process faster.

4.1.4 Module 4: Query Classification Using GloVe Embeddings

In this module, the user's query is classified to identify the most relevant category and label, using GloVe embeddings for similarity analysis. This process ensures that the user receives an approximately precise category for their query, even when the dataset is small. The input to this module is the top 10 data points ranked in Module 3. Each of these data points contains "clean_content," "initial_label," and "category_ID."

Now, the GloVe embedding process is utilized with a 300-dimensional vector space. This high-dimensional embedding helps to capture the semantic similarities between the query and the content of the selected data points. Using 300 dimensions enables better representation, especially since the content length can be large. This allows to accurately vectorize both the modified query and the content.

For each of the top 10 data points, we calculate the vector similarity be-

tween the ‘modified_query’ and the datapoint content. The datapoint that has the highest similarity score to the query is selected. The ”Initial_Label” and ”Category_ID” of the most similar datapoint are then assigned as the output category for the query. The determined Initial_Label is presented to the user, helping them understand the type of threat or issue their query relates to.

The classification approach using GloVe embeddings is chosen because it provides effective results with small datasets. No model can efficiently train on just 10 datapoints, as it would lead to overfitting due to the limited sample size. And GloVe embeddings help in determining semantic similarities directly, making this method more suitable for small-scale, real-time classification.

Advantages:

- **Accurate Classification with Limited Data:** Using GloVe ensures that the small set of data points is effectively utilized to determine the best matching category.
- **Handling Large Content Lengths:** The 300-dimensional embeddings allow for precise vectorization of longer content, ensuring the right category is selected.

4.1.5 Module 5: Summarized Text Generation from the Top 10 Contents

In this module, the larger version of the BART model, BART_Large_CNN, is employed to generate precise, query-driven summaries from the top 10 retrieved articles. Input:

- A modified user query
- Contents of top 10 articles retrieved

Process

The BART compares the article contents based on the input modified query, captures the utmost necessary information specific to the query, and generates corresponding concise summaries with the key information.

The model is fine-tuned using certain parameters to enhance and optimize the summarization process:

- **Generation Limit:** Configured to the maximum limit of 1024 to guarantee the summary adheres to the specified length and maintained at the default value of 30 as the minimum length to avoid excessively short output generation. Therefore, the model will not generate beyond this limit.
- **Number of Beams:** Set to a value of 6 to determine the accuracy and sharpened structure of the generated text response.
- **Batch Size:** Modified to a size of 4 for maintaining optimal processing mechanism.

- **Temperature:** Employed with a value of 0.8 to regulate the ingenuity of the output while preserving logical coherence and appropriate punctuation.
- **Repetition Penalty:** Implemented a penalty of 2.9 to mitigate repetitive information in the final summary. Here, since the necessary information from the contents is extracted based on the query, there is a possibility of repetition. To reduce such repetitions, a penalty of 2.9 is introduced in the system.

After tuning, the adjusted parameters allow BART to extract the query-centric information from the article contents. These values are then saved, and the retrieved contents are stored in a list called `summerize_text`.

Advantages of Query-based Summarization:

Instead of the proposed approach, an indulging generalized summarization of the articles can be used. Initially, a summarization of all the contents was performed using the T5 model, which was also included in an additional column of the dataset named `Summarized_Content`.

However, the user can ask for a query on any topic. While generating the generalized summarization, the model doesn't know which necessary parts to capture and may overlook relevant information to the user's query. For example, suppose an article's content is relevant to both cyber attacks and malware. The model may give relatively more weight to the cyber attack topic and then capture only that information. When a query related to malware is made, from the generalized summarization, the model may fail to generate a response correctly, as it has only information about cyber attacks but not about malware. To avoid such a mishap, the contents' summarization is based on the query rather than the generalized summary of the content, ensuring all the relevant information specific to the query is captured, providing a more comprehensive and precise response.

Output: The list named `summerize_text` is then forwarded to the next module, which will be processed further to generate the ultimate response to the user's query.

4.1.6 Module 6: Response Generation Using DISTIL-BART Model

The input to this module consists of two elements: a list of summarized texts generated from Module 5 using the BART-large-CNN model and the modified user query obtained from the earlier modules.

The summarized texts are consolidated into a single input string named `input_text`. Each index from the summarized list is concatenated into `input_text`, forming a comprehensive text block. This `input_text`, combined with the `modified_query`

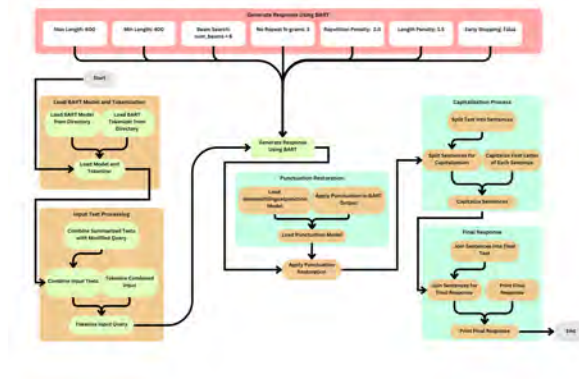


Figure 4.2: DISTILBART

from the user, is used as a prompt for our trained BART model.

The trained BART uses parameters maximum length: 600 and minimum length: 400 for the response. Additionally, it uses beam number=6 to give a deeper, and better structured response. To avoid repetitions of n-gram is utilized of size 3. Repetition penalty is 2 and length penalty=1.5 for detailed response. The response is decoded and stored in a variable. Additionally, to add punctuation, Deep Multilingual punctuation is used. This imports a punctuation model which adds punctuation to the previous response, followed by a capitalization algorithm. This generates a detailed and coherent final response that corresponds to the user's query.

This module generates a well-organized response using the DISTILBART model with punctuation model and capitalization algorithm. The response is delivered directly to the user, providing a synthesized explanation or guidance based on the input query and related summarized content. However, if the user query is not related to the domain of cybersecurity, this model may generate a less accurate response.

Advantages:

1. Using a BART model generates contextually powerful and coherent responses, utilizing pre-summarized content to provide users with comprehensive answers.
2. By generating responses based on summarized texts, it ensures that the response is both precise and informative, providing users with valuable insights into their queries.

4.1.7 Module 7: Link Generation and Detailed Summaries

The combined response generated from Module 6 and a list of summarized texts from the previous steps are input into this module.

At first, the combined response is provided to the users along with similar article links for direct access to the source materials. If users wish to see details, they have the option to view detailed information, which will contain a list of individual article links and corresponding summaries. These summaries are sourced from the previously generated list of summaries, allowing users to understand the main idea of each article before visiting the link.

Thus, the output of this module is a response with article links for immediate reference, and, if chosen by the user, a detailed list of individual summaries and links to help users explore the original content.

Advantages:

- This feature provides flexibility for users, offering a concise response while also enabling further exploration of the topic through detailed articles.
- Users can check out the summary and choose which article to access even before reviewing the content. This makes the system more user-friendly and adaptable to different needs while saving users time.

4.1.8 Module 8: Response History Storage

After each query and response interaction, the details are stored in a CSV file. The system captures the interaction details into a CSV file, creating a dataset with the following columns:

- `S1_no`: Serial number for each interaction.
- `user_query`: The original query submitted by the user.
- `summarized_text`: The summarized content related to the query.
- `response`: The final generated response.
- `label`: The identified category or label of the query.

This CSV file serves as a repository of previous interactions (query + response), allowing the system to check for similar queries in the future. After this, the `modified_query` is emptied and the content is passed on to the `previous_query` variable.

Advantages:

1. The stored history allows for quick responses if a similar query is received, as the system can retrieve the previous response directly from the CSV file without needing to regenerate it.
2. This feature significantly reduces the response time for recurring queries, enhancing the system's efficiency.

However, the stored responses may become outdated over time, potentially providing users with obsolete information. To address this, the CSV file is cleared out monthly to ensure that the stored responses are recent and relevant, maintaining the accuracy and reliability of the information provided.

4.2 Internal Architecture of Risk Radar

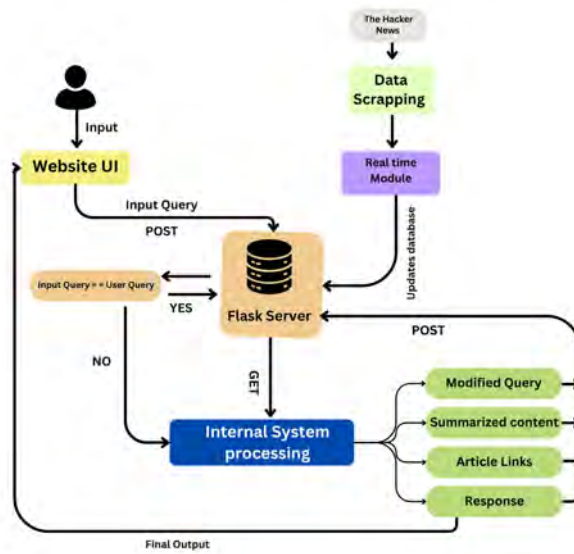


Figure 4.3: Risk Radar System

The process starts with a user entering a query via the website interface, which is sent as a POST request to the Flask server. The server verifies if the input query corresponds to a stored user query and gets the required information. If not, it interacts with the internal system processing module to get updated information. A data-scraping module updates the database in real-time. The Flask server holds the modified query, summarised content, article links, and response. Ultimately, the created response is sent back to the user interface for presentation.

4.3 Proposed Model Architecture

4.3.1 Top 30 Content Fetch Based on Query

This model employs the BERT-Large-Uncased architecture to generate the top 10 re-ranked documents based on a user query. The model configuration includes a maximum embedding length of 1024 tokens, with 24 layers, and truncation enabled to handle longer sequences effectively. Although embeddings typically utilize 512 tokens, setting the maximum length to 1024 helps prevent the loss of essential content and miscalculation of similarity scores for articles exceeding 512 tokens.

The pipeline begins with the BM25 module, where the query is tokenized using a simple preprocessing method. BM25 then processes the query using the term frequency-inverse document frequency (TF-IDF) to calculate BM25 relevance scores for each document and retrieves the top 30 relevant contents. In cases where the current query is related to a previous query, title filtering

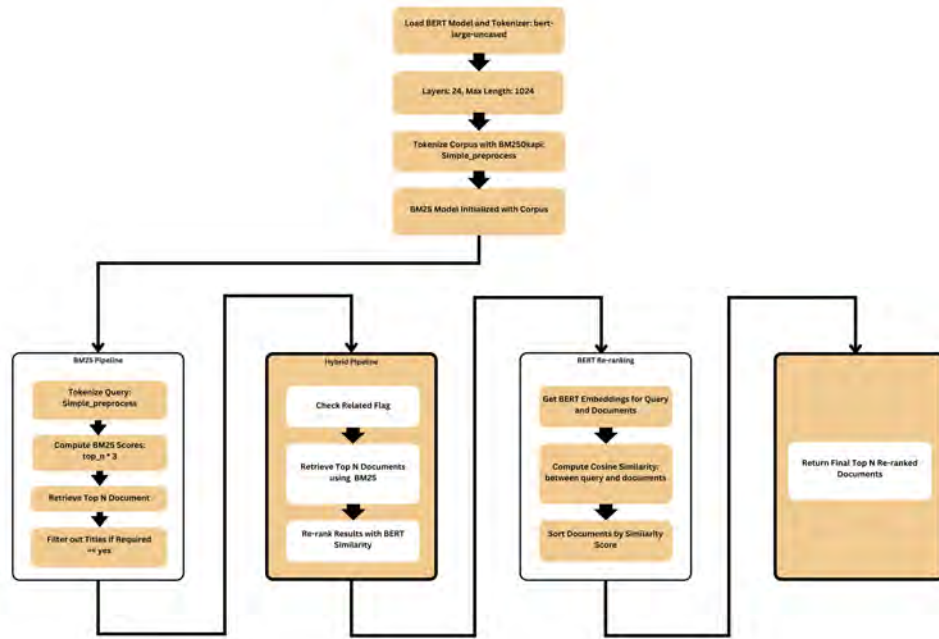


Figure 4.4: Top 30 Fetch Content Based on Query

is applied to refine the preliminary results, and are then forwarded to the Hybrid Pipeline, where the Hybrid Pipeline integrates BM25 scores with BERT’s re-ranking mechanism and the top 10 documents are then further ranked and passed to the BERT re-ranking module. In this stage, BERT embeddings are generated for both the query and the documents. The model computes the cosine similarity between the query and the contents, resulting in a similarity score. Based on this score, the documents are re-sorted, ensuring the most relevant documents are placed at the top. For example, if the 8th document scores the highest similarity, it will be moved to the top of the list. In fact, the architecture of the combined model can be handled the relation of the similar queries.

Finally, once re-ranking is completed, the model generates and returns the final top 10 re-ranked documents, ensuring the most relevant results are presented based on the computed similarity scores and thereby, providing users with the most relevant content based on their query through a robust, two-stage retrieval process where it ensures that important and contextually relevant content is prioritized, significantly improving the accuracy of search results.

4.3.2 Summarized Text Generation from top 10 contents

In this model, the larger version of BART, known as BART-Large-CNN, is utilized to generate text summaries from the top 10 content pieces. The model is initialized with its default parameter setup (406,290,432), which includes 12

encoder and decoder layers.

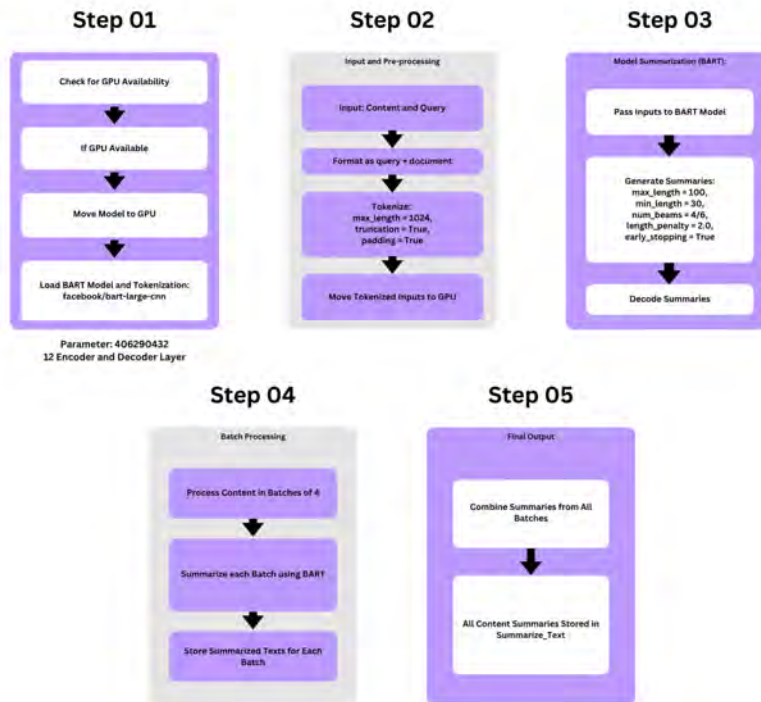


Figure 4.5: Text Generation from Top 10 Content Model

The process begins by loading the query and content as input, which is subsequently merged and formatted as query + document. The model is configured with a maximum input length of 1024, along with truncation and padding enabled to ensure appropriate input handling. These tokenized inputs are then transferred to the GPU for further processing, as the model was already shifted and configured to the GPU to enhance performance. Next, the summarization phase begins by passing the tokenized input to the BART model. To avoid excessively long summaries, a length penalty of 2.0 is applied. The number of beams is increased to 6 from the default value of 4, to improve sentence structure generation and quality. At this stage, BART generates a summary of the article content based on the query without introducing any new information. Consequently, the model primarily focuses on decoding rather than encoding. For batch processing, content is processed in batches of 4 to strike a balance between speed and memory consumption. Although an initial batch size of 6 was tested, it led to higher memory usage despite faster output generation, resulting in a more efficient choice of 4.

Lastly, in the final step, the summaries from all batches are combined and are appended into a list named `summarize_text`, providing a comprehensive overview of the top 10 content based on the query and thereby, ensuring that all content summaries are stored and readily available for subsequent tasks.

4.3.3 Classification Model

This classification model is designed to predict the `initial_label` and category of a user's query, which were originally termed `double_category` and `final_category`, respectively in the main dataset. To achieve this, the model employs the GloVe (Global Vectors for Word Representation) embedding technique, which generates 300-dimensional word embeddings to facilitate the tokenization and vectorization of the top 10 articles identified in the preceding module.

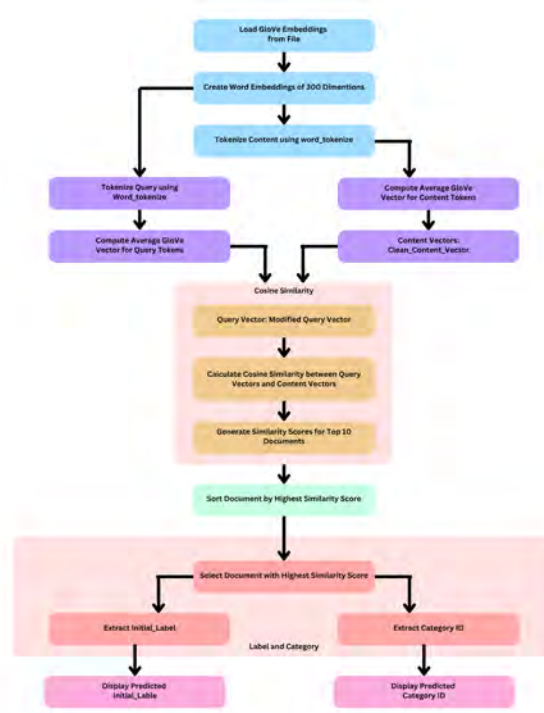


Figure 4.6: Classification Model

The procedure begins by loading the GloVe embeddings from a pre-trained file, which is then used to create word embeddings of 300 dimensions. These embeddings help in tokenizing both the user query and the article contents using the `word_tokenize` function. Both the content and the query undergo tokenization using the GloVe embeddings. The core functionality of this model involves evaluating the vector similarity between the query and the article contents. Once relevant content is identified, its corresponding category is determined. This is achieved through the tokenization of the content, mirroring the tokenization applied to the query. For each, an average GloVe vector is computed, leading to the creation of the `Clean_Content_Vector` and the `Clean_Query_Vector`, with the query vector further modified for enhanced accuracy. The cosine similarity between these vectors is calculated, leveraging the extensive 300-dimensional space to measure their proximity. This calculation yields similarity scores for all ten article contents, which are then sorted, placing the highest similarity score at the forefront and the lowest at the bottom.

Upon identifying the content with the highest similarity score, its corresponding `initial_label` and Category ID are extracted. The `initial_label` is displayed to the user as the predicted `initial_label`, whilst the Category ID is retained for backend functionalities, remaining hidden from the user interface, thereby ensuring seamless system functionality and efficient data processing.

4.3.4 Bart Training

In this modeling process, the BART-base model is utilized, and the tokenizer is first loaded. Subsequently, the pretrained BART model is initialized. By default, the model architecture includes 12 layers each for encoding and decoding. However, in this case, the number of encoding and decoding layers has been reduced to 6, a decision made to simplify the model's structure. This reduction in complexity allows the model to train faster and consume fewer computational resources.

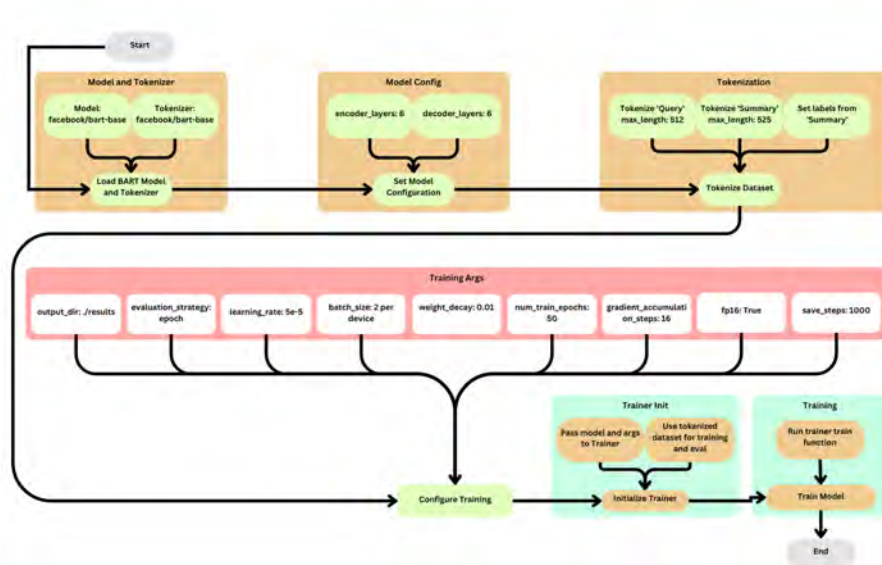


Figure 4.7: Bart Training

Next, the input query is tokenized, with a maximum length set to 512 tokens to ensure that any query exceeding this length is truncated. Similarly, the summary is tokenized with a configured maximum length of 525 tokens, ensuring that the generated summary is not shorter than this threshold. A tokenized dataset is then constructed, incorporating the model configuration, tokenized query, and tokenized summary. This dataset is subsequently passed on to the training configuration stage, where parameters controlling how the model will be trained are set.

The training configuration specifies several important aspects:

- **Output Directory:** Defined as `./results`, where the model’s outputs will be saved.
- **Evaluation Strategy:** Set to "epoch," meaning that evaluation will occur at the end of each epoch.
- **Learning Rate:** A learning rate of 5×10^{-5} is employed, which has proven to yield better performance compared to other experimented values (such as 1×10^{-5} to 4×10^{-5}), where the training and validation loss were significantly higher.
- **Batch Size:** Set to 2 per device for both training and evaluation.
- **Number of Epochs:** The model is trained over 50 epochs.
- **Weight Decay:** A weight decay of 0.01 is used to prevent overfitting.
- **Gradient Accumulation Steps:** As the batch size is small, the gradient accumulation steps are set to 16 to ensure sufficient gradient updates.
- **Mixed Precision Training:** Enabled (fp16), as the model is run on a GPU, leading to faster training.
- **Save Steps:** Set to 1000, ensuring that the model checkpoints are saved after every 1000 steps.

Once the trainer is initialized, the configured model and associated arguments, along with the tokenized dataset, are passed to it for training. Finally, the model is trained, and the output is generated and saved.

Advantages

1. The reduced complexity speeds up the training process.
2. Fewer resources are required due to the simplified model.
3. With fewer layers, the likelihood of overfitting is minimized.
4. The validation and training loss are lower compared to when 12 layers were used.

Disadvantages

- The reduced complexity might limit the model’s ability to capture intricate relationships and context between words, potentially affecting performance on complex queries, leading to less precise answers for more complex queries.
- However, in this case, it has been observed that with 6 layers, the model provides more precise responses.

Balancing computational resources and execution time, the model is considered optimized and a well-balanced approach for many tasks, with 6 layers.

Chapter 5

Real-Time Implementation

Relying on only a dataset of 13k data points is not an optimum option as information on cyber security, as well as every domain, keeps updating regularly. So, updating the dataset based on real-time data collection can be efficient and relevant for the proposed system. By doing this, responses can be unique and will be more accurate; we could also provide the newest article link to the user. This chapter outlines an overview of how real-time data implementation updates the dataset using the everyday captured information.

5.1 Scraped Dataset and Pre-processing

At first, a scraped dataset will be created using the scraping code for that specific website. Here, it will scrape the data that is not already in the dataset, as it will check the date of the first row of the dataset. It will scrape the data until the date matches. After acquiring the dataset, it will be loaded. Initially, it has a total of four columns, including an unnamed column that acts as a serial number. Columns like Date, Content, and Category, similar to Double_Category of the main dataset, are present. Now, some empty columns need to be added, such as Priority_Category and Final_Category. The unnamed column and Category column must be renamed as SL no and double_category, respectively, to merge the scraped dataset with the main dataset. In fact, the Final_category label needs to be set, and to do so, k-means clustering is used.

5.2 Integration with Clustering Models

As the scraped dataset is unsupervised (has no label), autoencoder K-means clustering is used to classify the label of Final_category. This clustering is within four different clusters based on Title and Content. The Title and Content columns are joined into a new column called Text_column, which then passes to the SentenceTransformer('all-MiniLM-L6-v2') model to convert into sentence embeddings. This is then converted into a tensor format to be used in the autoencoder.

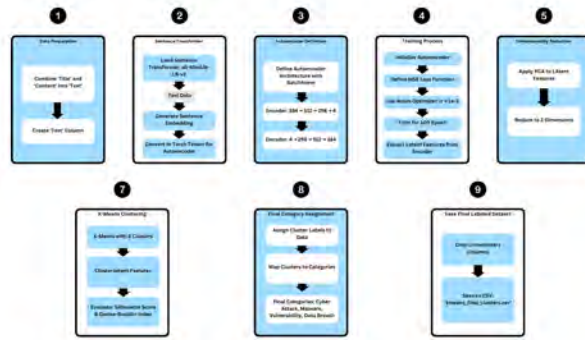


Figure 5.1: K-means clustered autoencoder

The autoencoder is configured with an input dimension derived from the embeddings and a latent dimension of 4, using batch normalization. The model is trained for 100 epochs using MSELoss as the loss function. The Adam optimizer is employed with a learning rate of 1×10^{-3} . After training, latent features are extracted from the encoder. PCA is then applied to reduce the dimensionality of latent features, followed by K-means clustering to categorize the data into four clusters: [1, 2, 3, 4], which correspond to Cyber Attack, Malware, Vulnerability, and Data Breach, respectively.

The final cluster labels are mapped for the Final_category column as Klabeled_scrapped_dataset by dropping the Text and Cluster columns, creating a processed dataset ready for integration with the main dataset.

5.3 Merging with the Main Dataset

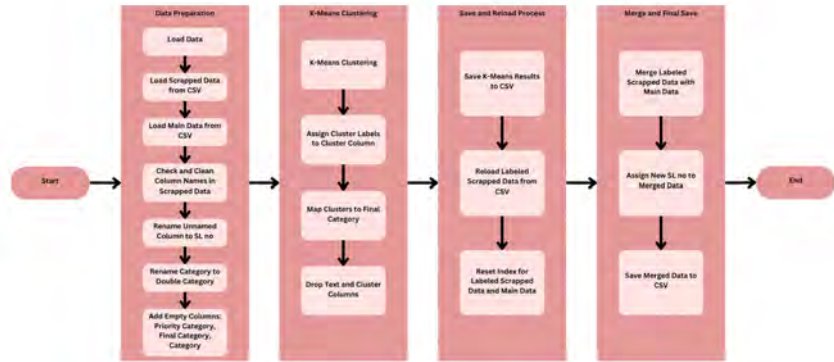


Figure 5.2: Real-time dataset Implementation

Once updated and labeled, `Klabeled_scrapped_dataset` is then merged with the main dataset. New data points are added to the top, and the SL No column is re-indexed to maintain order automatically. Now, this updated version of the main dataset will overwrite the actual main dataset and be saved as the most current version. This merging process is computationally efficient and automatic and focuses on combining datasets without recalculating embeddings or clustering for existing data.

5.4 Improving Response Quality

This implementation provides significant benefits to the proposed system, as the dataset will be updated daily, enhancing the quality of the system's responses over time. As the dataset expands, the Distil-BART model will continuously learn from these new data, ensuring that the model responds more accurately and relevantly and that the articles and summaries will be the newest. For instance, a query today might yield a more detailed and precise response with a summary and link when asked again after two months due to real-time implementation.

Additionally, at the time of updating the main dataset, the user will not face any difficulty in getting a response, as data is not fetched from the main dataset; rather, a copy of the dataset is used to fetch the information. This ensures that user query responses remain stable even as new data is integrated

into the main dataset. Thus, the adaptability of the data integration process provides a solid scope for ongoing improvement and future expansion.

The current setup uses data from a single website, but the system is designed for scalability. Future enhancements include adding data sources from multiple websites. With this setup, adding a new website would only require adjusting the web scraping algorithm for that particular site. The rest of the pipeline remains unchanged, making it adaptable to different data sources. Adding more datasets will improve the depth and scope of responses provided to users. As more websites are added, the system will gain access to a wider range of content, resulting in a more robust dataset. This will improve the quality and relevance of the information available to users while keeping up with the changing landscape of cybersecurity threats.

Chapter 6

Result Analysis and Comparison

6.1 Execution Time & Precision to fetch Top 30 Content

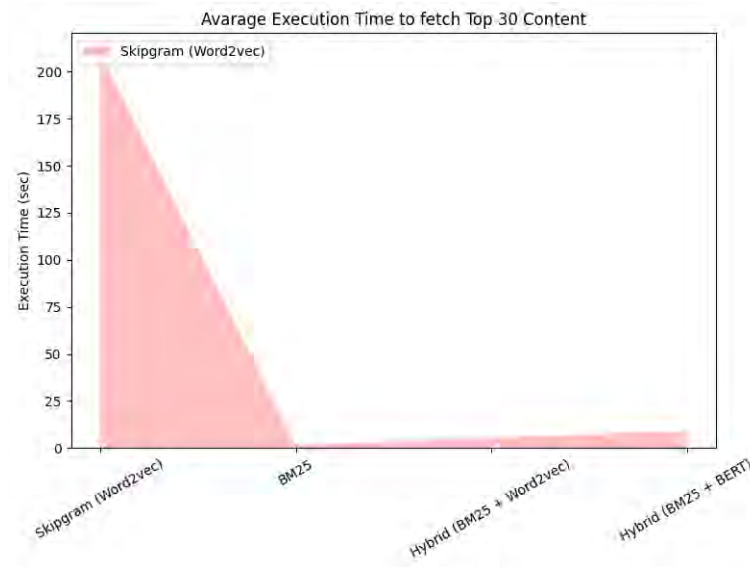


Figure 6.1: Average Execution Time to fetch Top 30 Content

The graph illustrates the marked variations among the different models in terms of how long it takes for each model to extract the most similar content. However Skip Gram takes quite a longer time of 210 seconds which is somewhat relatively extreme since scoring methods like BM25 and hybrid models BM25 with Word2vec and BM25 with BERT have relatively stable low execution times of 1.5 seconds, 5 seconds and 8 seconds respectively. Here, both hybrid models are efficiently faster because after BM25 extracts 30 data points, the Word2Vec and BERT models only embed these 30 documents. Hence, the BM25 and its hybrids handle most basic queries much more efficiently than these dimensional approaches, Skipgram's significant computing burden

indicates a difficulty while also offering an opportunity to enhance its comprehensive semantic representation for retrieval tasks. Investigating performance trade-offs and utilizing innovations like parallelisation or model pruning may enhance the feasibility of Skipgram for large-scale applications, facilitating a balance between semantic depth and efficiency.

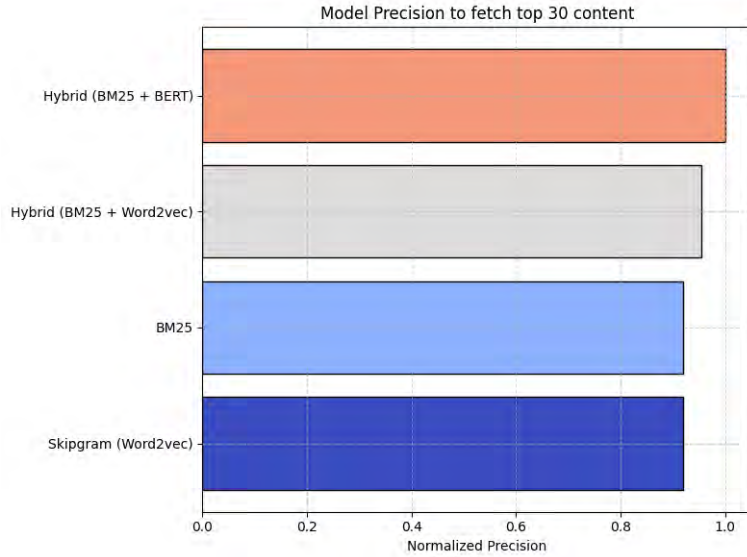


Figure 6.2: Model Precision to fetch top 30 content

It is evident from the above graph that the Hybrid model (BM25 + BERT) obtains the greatest normalized precision of 0.86 among models that retrieve the top 30 content. The Hybrid model (BM25 + Word2vec) comes in second with a precision score of 0.82. While BM25 also demonstrates strong precision of 0.79, it falls slightly behind its hybrid counterparts, and Skipgram (Word2vec), though lower in precision of 0.79, still performs reasonably. This result is optimistic as it highlights the potential of hybrid models, particularly BM25 combined with BERT or Word2vec, to significantly enhance precision. Despite Skipgram’s lower precision and high execution time, its semantic richness suggests potential for improvement.

Here, the precision was calculated with a ground truth dataset of 500 data points. 500 queries are passed to the Hybrid model (BM25 + BERT) and the fetched article links were checked manually to make sure if the given article link content is truly matched with the user query. If all the content of top 10 is matched then labeled as True (1) and if even only one article link content does not match out of 10 then labeled as False (0).

The analysis of the above resultant graphs and table reveals that each model presents its distinct balance between execution time and precision. As for Skipgram (Word2Vec) model, this is one of the models with a relatively large execution time of 210 seconds but with a precision score of 0.79, although it is declared unfit for our proposed system. On the contrary, while BM25 can achieve a precision score of 0.79 in fetching the documents, it only requires 1.5 seconds in performing the fetched processes which in most cases are speed sensitive as the whole dataset is tokenized as corpus only one time at the

Model name	Average Execution Time (sec)	Precision	Overall
Skipgram (Word2vec)	210	0.79	Bad
BM25	1.5	0.79	Average
Hybrid (BM25 + word2vec)	5	0.82	Good
Hybrid(BM25 + BERT)	8	0.86	Good

Table 6.1: Overall performance of models

start. The BM25 + Word2vec and BM25 + BERT hybrid models present more improvements in two aspects rather than one at a time, namely, time and quality. In terms of the structural precision of 0.82, the model achieved this in 5 seconds as compared to BM25 which took 8 seconds with a higher precision of 0.86. Overall such a traditional model as BM25 + BERT strikes an excellent balance between performance and precision, making it the most optimal choice for applications that prioritize accuracy without sacrificing efficiency whilst does not allow to overestimate BM25 and therefore is an overall best model

6.2 Query Classification Analysis



Figure 6.3: Time Taken in Classification

The above diagram compares the time taken on average by three distinct approaches in classifying the models: TF-IDF, Word2Vec (300d) and GloVe (300d). This indicates that Word2Vec runs quite slow as it takes about 140 seconds to embed the 30 documents, followed by the second slowest classification model's speed which is the GloVe(300d) with a 3.5 seconds outcome while the TF-IDF system is almost instantaneous with just 2 seconds, indicating that whilst Word2Vec is more system computation resources, TF-IDF and GloVe are more efficient in terms of processing time.

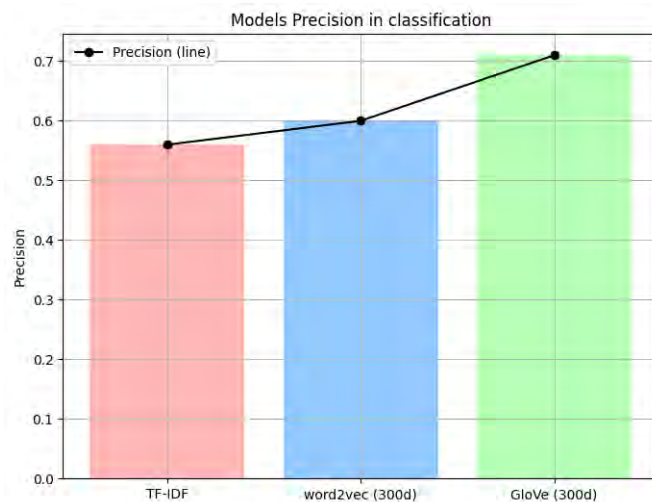


Figure 6.4: Model Precision in Classification

The graph shows that GloVe (300d) model specifies that the specificity for the classification accuracy scores of 0.71 is obtained for the best, followed by the Word2Vec (300d), where TF-IDF achieved lower values of precision scores of 0.6 and 0.56 respectively. In comparison to the conventional TF-IDF model,

this suggests that embedding-based models such as GloVe and Word2Vec are more effective at capturing the semantic links required for more precise classifications. Here, model precision was calculated with a ground truth dataset of 1000 data points. Here, 1000 queries are passed to the GloVe embedded model and the predicted labels are checked manually to make sure if the predicted label is truly matched with the user query. If matched, then labeled as True (1) and if not matched then labeled as False (0).

In reality, 0.71 is a better precision score because there are chances that the fetched top 10 articles might contain significant information similar to the query but the actual label of the article is different. In this scenario, the most fetched contents might be of different labels but carry important information which will cause the capturing of a different classification label by GloVe as it has to classify based on the fetched ones only.

Model name	Precision	Average Execution Time	Overall
TF-IDF	0.56	2 sec	Bad
word2vec (300D)	0.6	140 sec	Bad
GloVe (300D)	0.71	3.5 sec	Good

Table 6.2: Classification

6.3 Text Generation from the Top 10 Contents

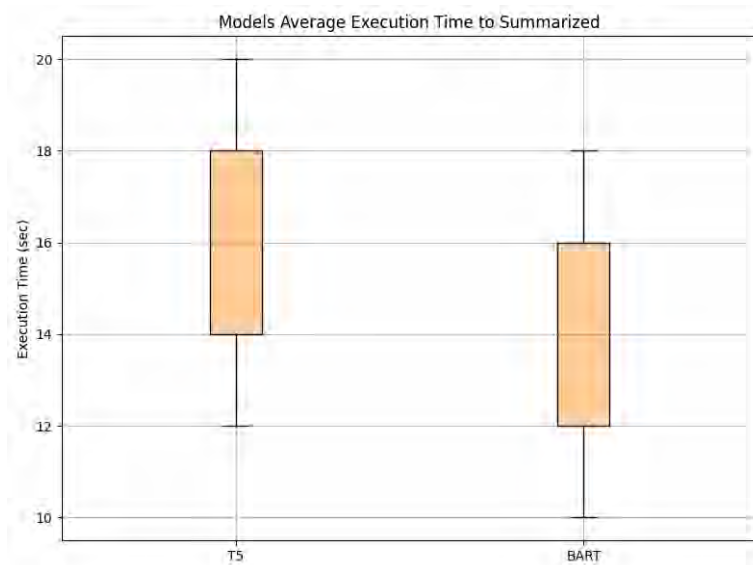


Figure 6.5: Models Average Execution Time to Summarized

- **T5 Model:** The T5 model has an average execution time ranging between 12 to 22 seconds. It shows an average content similarity with queries between 77% and 88%, indicating a decent alignment of summaries with user inputs.
- **BART Model:** The BART model performs slightly faster, with an average execution time between 10 to 18 seconds. It demonstrates a higher similarity with queries, ranging from 81% to 91%, which signifies a better precision in aligning summaries with user queries compared to T5.

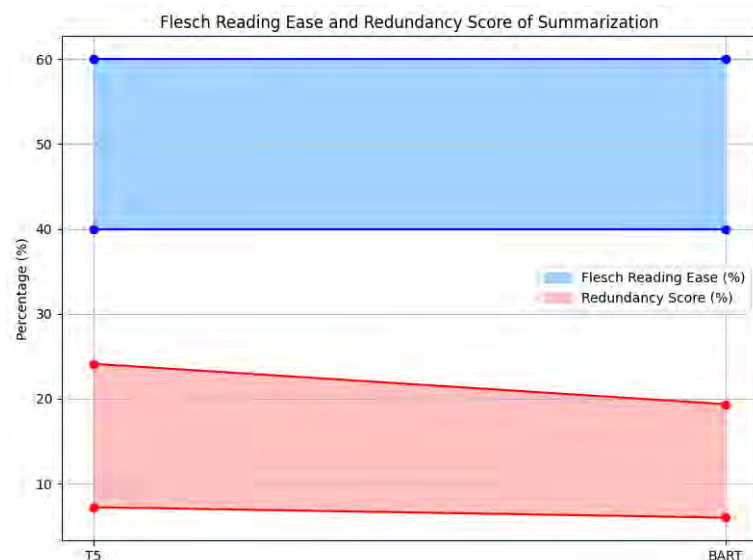


Figure 6.6: Flesch Reading Ease and Redundancy Score of Summarization

The average Flesch Reading Ease score of T5 falls between 40 and 60, making the readability moderately difficult. BART also has an average Flesch Reading Ease score between 40 and 60, indicating a similar readability level.

Model Name	Average Execution Time	avg Summarized Content Similarity with Query (%) / Precision	avg Flesch Reading Ease (%)	avg Redundancy Score (%)	Overall
T5	12-22 sec	77 - 88	40 - 60	7.23 - 24.11	Good
BART	10-18 sec	81 - 91	40 - 60	6 - 19.36	Better

Table 6.3: Flesch Reading Ease and Redundancy Score of Summarization for both models

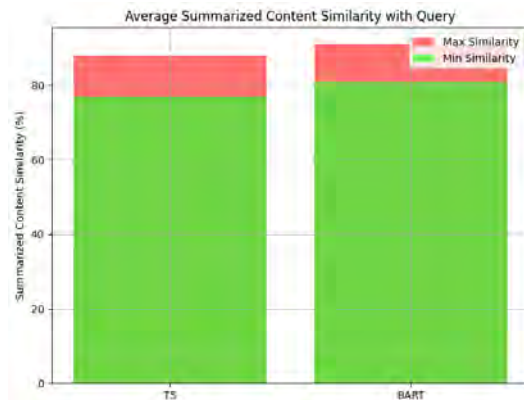


Figure 6.7: Average Summarized Content Similarity with Query

T5 model’s average redundancy score ranges from 7.23% to 24.11%, suggesting some potential for repetitive information in the generated summaries. Overall, the performance of T5 is considered good.

BART’s average redundancy score is lower, ranging from 6% to 19.36%, suggesting less repetition in the generated summaries. Given these factors, BART is rated as better in overall performance.

While both models are effective for summarization, BART stands out due to its balance of speed, higher similarity with queries, and lower redundancy, making it the preferable choice for this system even though the model is not trained with significant summarized texts.

6.4 DISTILBART Train Result

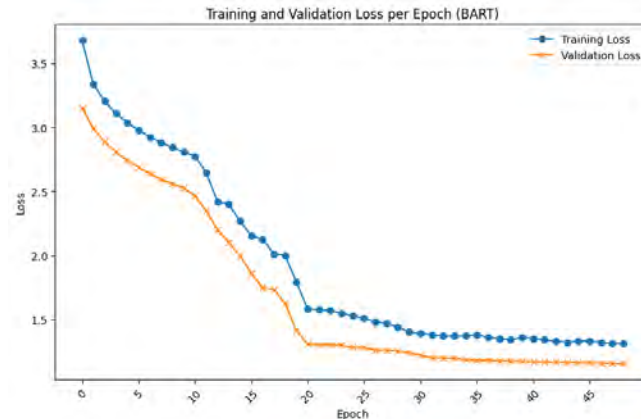


Figure 6.8: Training and Validation Loss per Epoch (BART)

The training of the DistilBART model was evaluated using different metrics such as learning rate, average training loss, average validation loss, and the number of epochs. The model was trained with a learning rate of $5e(-5)$. After trying a number of learning rates like $1e(-5)$, $2e(-5)$, $3e(-5)$, $4e(-5)$, this learning rate was chosen to strike a balance between training speed and the stability of the model's learning process. The query or question based on the summary has to be done manually so that we can train the model precisely. Here, we are manually creating a question or query based on a summary as sometimes questions for a specific summary can become irrelevant as a result we finalized 7k data points for training out of 12k data points Throughout the training process, the average training loss recorded was approximately 1.8201. The average validation loss was observed to be around 1.4772. The validation loss being lower than the training loss indicates that the model performs well on data it hasn't seen before, without overfitting. The model was trained for 50 epochs. This was determined to be a suitable number of epochs for achieving stable training and validation losses after checking 15 and 25 epochs.

6.5 K-mean Clustering

The comparison of different unsupervised models for clustering, including K-means with an autoencoder, agglomerative clustering, a supervised classification approach, and Self-Organizing Maps (SOM), is summarized based on Davies-Bouldin Index, Silhouette Score, Calinski-Harabasz Index, and execution time metrics. Here's the analysis in text format:

K-Means Clustering (Autoencoder) shows the best performance among the models, with a high Silhouette Score of 0.7936, indicating well-defined clusters. Its Davies-Bouldin Index of 0.4812 suggests low intra-cluster variance, and a Calinski-Harabasz Index of over 100 indicates good separation between clusters. However, its execution time is relatively higher, ranging from 120 to 180 seconds. Overall, this method is rated as Good.

Learning Rate	avg Training loss	avg Validation loss	Epoch
5e ⁻⁵	1.820147551	1.477192061	50

Table 6.4: Training result

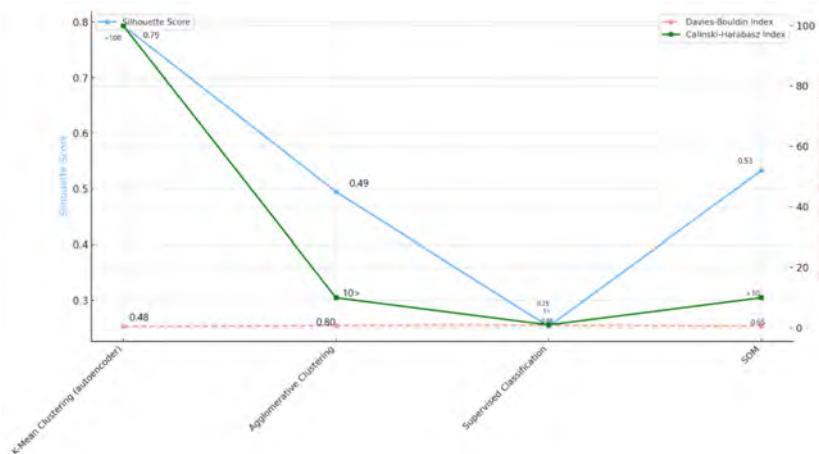


Figure 6.9: Comparison of clustering model

Agglomerative Clustering performs poorly in comparison, with a Silhouette Score of 0.4944, indicating less distinct clustering. The Davies-Bouldin Index is higher at 0.8031, suggesting worse intra-cluster cohesion. The Calinski-Harabasz Index is below 10, reflecting poor cluster separation, though the execution time is shorter, ranging between 20 and 40 seconds. Overall, this method is rated as Bad.

Supervised Classification Approach shows the weakest performance in clustering, with a Silhouette Score of only 0.2528 and the highest Davies-Bouldin Index of 0.8835, indicating poorly defined clusters and high intra-cluster variance. The Calinski-Harabasz Index is below 1, suggesting inadequate separation. It has an execution time of 40 to 80 seconds. Thus, it is rated as Bad. However, it can perform well if enough classified or labeled dataset can be provided for this model for training.

Self-Organizing Maps (SOM) achieves a moderate performance with a Silhouette Score of 0.5325, indicating average clustering quality. Its Davies-Bouldin Index is 0.6578, suggesting moderate intra-cluster cohesion. The Calinski-Harabasz Index is above 10, showing some degree of separation between clusters, with an execution time ranging from 22 to 40 seconds. It is rated as Average.

These performance metrics were calculated from the 1.3k real time implemented dataset where how k-means clusters accurately on that 1.3k dataset were shown below:

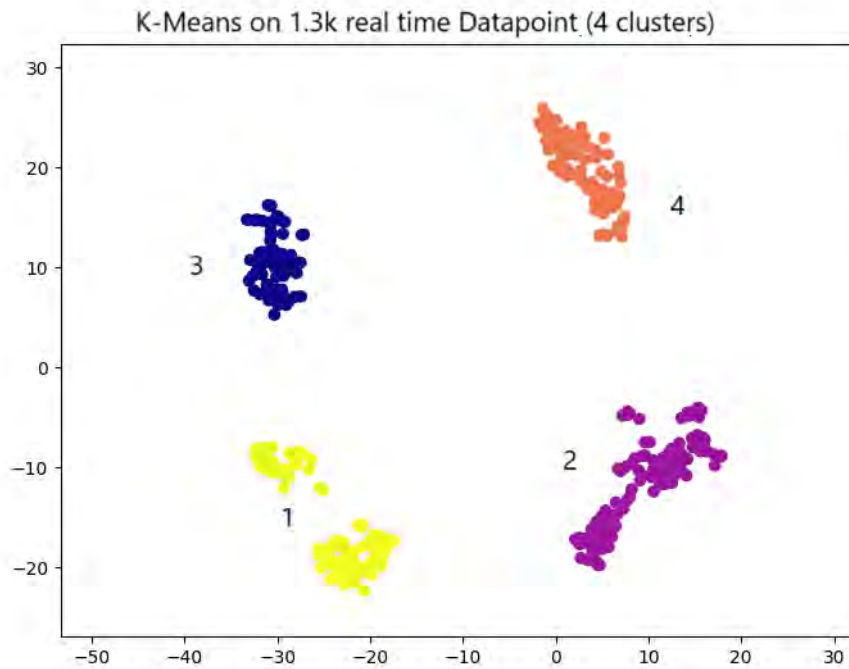


Figure 6.10: K-Means on 1.3k real time Datapoint (4 clusters)

K means clusters the data points so neatly that each of the 4 labels are in separate vector space. These four clusters represent the four categories: 1,2,3,4. 1 corresponds to Cyber Attack, 2 for Malware, 3 for Vulnerability, and 4 for Data Breach.

Therefore, K-means Clustering with an autoencoder provides the best balance between clustering quality and performance, though it requires more time to execute compared to other methods.

Model Name	Silhouette Score	Davies-Bouldin Index	Calinski-Harabasz Index	Execution Time (Sec)	Overall
K-Mean Clustering (autoencoder)	0.7936	0.4812	>100	120-180	Good
Agglomerative Clustering	0.4944	0.8031	<10	20-40	Bad
Supervised Classification Approach	0.2528	0.8835	<1	40-80	Bad
SOM	0.5325	0.6578	>10	22-40 sec	Average

Table 6.5: Comparison of clustering model

6.6 Response Result



Figure 6.11: Average Response Time Comparison

The time to generate response by the trained Distil-Bart is quite less than the pre-trained NEO-GPT. NEO-GPT is more complex and can handle complex queries even though it is not trained whereas Distil-BART has less encode and decode layer which makes it a less complex model. However we failed to train the NEO-GPT because of computational resources. If there is a possibility to train the NEO then it can outperform the BART model. Although Distil-

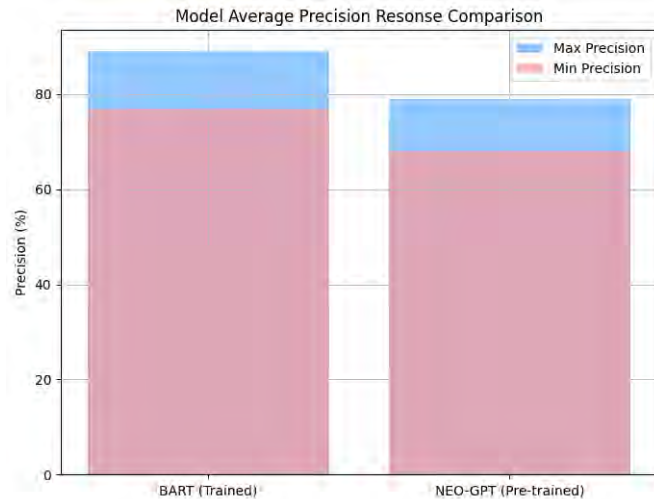


Figure 6.12: Model Average Precision Resonse Comparison

BART is less complex, it can give precise results than the NEO-GPT. Its precision range is 77-89. As it was trained on both query and summarization of the dataset of 7k data points out of 12k data points. The response from the trained BART is quite accurate and precise based on the query. Here, model precision was calculated with a ground truth dataset of 1k data points. Here, 1k queries are passed to the models and the responses are checked manually

to make sure if the predicted response is truly accurate with the user query. If accurate, then labeled as True (1) and if not accurate then labeled as False (0). On the other hand most of the time fine-tuned NEO gives less accurate response and if its sample parameter is true then it can give precise response of range 68-79. But out of all samples some can be accurate and others can be irrelevant which is why its precision score is less than the trained Distil-BART. However, Distil-BART loses some of its precision due to the query of different domains. But, as the real time implementation is implemented that is why in future the domain will be expanded and the model will be able to give accurate results on different domains.

Query	Context	Response
What is wannacry?	<p>A new variant of the ransomware strain that infected 300,000 machines in 150 countries over the weekend has been linked to a state-sponsored hacking group in North Korea, according to an analysis published by Symantec's threat intelligence team on Wednesday. The malware is being distributed via spam emails and malicious download links designed to infect Windows PCs with ransomware and double extortion. It is believed that the cyber criminals behind the campaign are exploiting a Windows SMB vulnerability that was patched by Microsoft as part of its February patch Tuesday updates for Windows XP and Windows Vista to spread the infection into more than 160,000 computers worldwide. This week, we have reported about a new version of ransomware strainware called EternalBlue that has already affected more than 200,000 computers across 120 countries, including Australia, Germany, Italy, Turkey, Italy, Turkey and Vietnam. Since the start of the month, the worm has been designed to encrypt files on a computer and demand victims to pay a ransom, usually 200-500 in bitcoins, to regain access to the system. EternalBlue is a sophisticated piece of malware that uses a zero-day vulnerability in Microsoft Windows operating system that allows remote attackers to severely escalate arbitrary code on the infected computer without requiring any user interaction. Once installed, the malware creates a full switch and injects itself into every process running on the victim's computer until the attacker pays a ransom amount to decrypt the files. But researchers have also discovered other variants of the same malware that are capable of stealing passwords and other sensitive information from compromised computers. The latest variant of wannacry has been found to infect more than 45,000 computers in 73 countries, including United States, Russia, Germany, Yemen, Italy, Philippines, South Korea, Spain, Canada, Italy, Poland, Thailand, Japan, Europe, Indonesia, Mexico, Malaysia, France and the Netherlands. So far, there is no evidence that the attacks are still ongoing or that the operators behind the operation are continuing to use the same techniques to spread their malware. Last week, security researchers at Group-IB's antivirus firm SentinelOne said they were able to identify two new samples of wannacry trojan that are being used to infect unsuspecting computers worldwide using a technique dubbed wormable. A French security researcher who goes by the alias quark0x3 has discovered a new type of malware named wannacry that spreads like a worm by leveraging a ms17-010 exploit cve-2017-0145 that has been exploited by cybercriminals. In parallel, Microsoft has just released an emergency security patch update for all its unsupported version: windows xp, vista, windows 8 server 2003 and 2008 editions. If your organization, for some reason, is still running on windows xp or vista, you are advised to immediately apply the patches as soon as possible to mitigate the risk associated with these infections. Stay tuned to the hacker news for more details on the latest developments. Check out the blog post published Wednesday by athenaquest, chief security officer.</p>	<p>WannaCry is a widespread ransomware attack that occurred in May 2017, affecting over 230,000 computers in more than 150 countries. It specifically targeted systems running the Windows operating system by exploiting a vulnerability in Microsoft's Server Message Block (SMB) protocol, known as the EternalBlue exploit, which was leaked by a hacker group called Shadow Brokers. This vulnerability allowed WannaCry to spread rapidly across networks without user intervention. Once infected, the malware encrypted files on the victim's computer, making them inaccessible. The attackers then demanded a ransom, typically in Bitcoin, to restore access to the encrypted data. If the ransom was not paid within a specified time, the price would increase, and eventually, the files would be permanently locked. Many organizations, including hospitals, government institutions, and businesses, were severely impacted by the attack, leading to significant disruptions. One of the most notable effects of WannaCry was on the UK's National Health Service (NHS), where many hospitals and clinics were forced to cancel appointments and operations due to the ransomware attack crippling their systems. Other industries, such as telecommunications, transportation, and logistics, were also affected globally. Microsoft had issued a security patch for the vulnerability two months prior to the attack, but many organizations had not applied it, leaving their systems exposed. WannaCry also had a "kill switch" discovered by a security researcher, which halted its spread when a specific domain name was registered. Despite its widespread impact, the attackers behind WannaCry did not earn significant amounts of money from ransom payments, as many victims either refused to pay or restored their systems using backups. The attack highlighted the importance of keeping systems updated with security patches, using robust backup systems, and implementing strong cybersecurity measures to protect against similar threats.</p>

Figure 6.13: GPT Table

Here the model is not trained on the query of wannacry still it can generate response on its own using the summarized texts and the query. Query similarity score with the proposed model response is 96%. In fact, the similarity score between the GPT response and the proposed response is 49% which shows that the proposed system can generate precise responses based on summarized texts and query.

Precision dataset: <https://tinyurl.com/38ehf7v5>

6.7 Limitations

During our research, we encountered significant challenges related to computational power. We initially aimed to leverage models like GPT (2.7 billion and 1.3 billion parameters), RAG, BART Large, and BART Base. However, due to the limitations of our computational resources, we could not train these larger models effectively. Instead, we used DistilBART, which is a smaller model with fewer parameters. Although it performed well, using a larger model could have potentially improved the performance by capturing more

nuances and producing richer responses. The current system relies on a set of predefined rules for capturing sequences. This approach, while useful, has its limits in terms of flexibility and adaptability. If a query falls outside the scope of the existing rules but is actually related in context, the system might not recognize it as such. Additionally, writing and maintaining rules manually can be cumbersome. And rules must be carefully ordered and evaluated to ensure accurate sequence maintenance.

For the query classification part, there can be instances where the content of a datapoint is very similar to the query, but the `Category_ID` of the content might differ from the category intended by the query. This is a limitation due to the dependency on similarity-based classification.

The BART model occasionally produces summaries that are too concise, potentially omitting key points from the original content if the information related to queries are not in the dataset. This can be problematic when users require more detailed information to fully understand their specific topic.

With the real-time implementation, the dataset size grows as new data is scraped everyday. As a result, more powerful computational resources may be required to efficiently process and merge the expanding datasets. This can increase both the time and costs involved in maintaining the system.

Ideally, training the Distil-BART model directly on the content of the articles would produce more accurate results. However, due to computational limitations, we had to rely on training the model using summaries instead because to train the Distil-BART queries are needed and to generate this query based on summary manually is very time consuming work also. While this approach allowed us to manage resource constraints, a content-based training could have led to better understanding and more refined responses. Additionally, as the training queries were manually written, only around 7,000 queries were made. A larger set of training queries would likely enhance the model's capability to handle a broader range of user inputs.

Chapter 7

Conclusion

This thesis presents Risk Radar, a query-based system designed to enhance user understanding of cybersecurity threats through advanced NLP techniques. By integrating tools like BERT for semantic similarity, BM25 for efficient retrieval, and DistilBART for summarization and context-based response generation, the system effectively addresses the complexities of user queries related to cyber incidents. Our rule-based approach for maintaining query sequence and context continuity offers a lightweight yet effective solution, providing accurate responses even in multi-turn interactions. The real-time data scraping mechanism ensures that the system stays updated with the latest cybersecurity information, delivering timely insights to users. Although computational limitations influenced some design choices, such as the use of distilled models over larger, more powerful alternatives, Risk Radar strikes a balance between performance and resource efficiency. Overall, this work demonstrates the potential of a well-designed, query-based information retrieval system in delivering precise, context-aware assistance for complex domains like cybersecurity. Future improvements could include expanding the dataset, refining sequence rules, and further training models to enhance system accuracy and applicability across different fields. Risk Radar thus sets a foundation for user-centric, real-time knowledge dissemination in cybersecurity, with the potential for broader applications in other technical fields.

7.1 Further Research Areas

Developing more sophisticated rules for capturing sequences, as well as expanding the list of multi-word expressions (MWE) would enhance the interpretation of user queries. This would help the system better understand and respond to more complex queries. In fact, adding more rules for sequence capturing can be useful to handle more complex and difficult queries. Secondly, incorporating data from multiple websites would allow us to create a larger and more diverse dataset. This would not only improve the scope of the data but also provide richer content for training and response generation. The real-time nature of our scraping setup means that adding new sources would enhance the variety and relevance of the dataset which would allow for

more effective training of the BERT model using advanced NLP techniques. This would help improve the model's ability to capture the semantic similarity between user queries and the dataset, leading to more accurate results. Additionally, A tracking mechanism can also be implemented to monitor the volume of data extracted each day. This can be expanded to track data contributions from each website as more sources are added. To accommodate multiple sources, a new column can be added to the main dataset to indicate the origin of each datapoint, which will help with identifying the source and data analysis. A promising area of research is to create a dataset that includes pairs of queries, summaries, and full content. This would enable the BART model to learn how to generate queries based on given summaries and content. Such a dataset could significantly improve the model's capability to formulate relevant and specific responses to user inputs.

Bibliography

- [1] R. Shah, S. Lahoti, and K. Lavanya, “An intelligent chat-bot using natural language processing,” *International Journal of Engineering Research*, vol. 6, no. 5, pp. 281–286, 2017.
- [2] N. T. W. Khin and N. N. Yee, “Query classification based information retrieval system,” in *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, IEEE, 2018, pp. 151–156.
- [3] —, “Query classification based information retrieval system,” in *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, vol. 3, IEEE, Oct. 2018, pp. 151–156.
- [4] C. Zhong, T. Lin, P. Liu, J. Yen, and K. Chen, “A cyber security data triage operation retrieval system,” *Computers & Security*, vol. 76, pp. 12–31, 2018.
- [5] F. T. C. (FTC). (2019). “Equifax data breach settlement.” Accessed: 2024-10-17, [Online]. Available: <https://www.ftc.gov/enforcement/refunds/equifax-data-breach-settlement>.
- [6] M. Aleedy, H. Shaiba, and M. Bezbradica, “Generating and analyzing chatbot responses using natural language processing,” *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 9, 2019.
- [7] Cybersecurity and I. S. A. (CISA). (2020). “Emotet malware advisory (aa20-280a).” Accessed: 2024-10-17, [Online]. Available: <https://www.cisa.gov/news-events/cybersecurity-advisories/aa20-280a>.
- [8] A. Esteva, A. Kale, R. Paulus, *et al.*, “Covid-19 information retrieval with deep-learning based semantic search, question answering, and abstractive summarization,” *npj Digital Medicine*, vol. 4, p. 68, 2021. DOI: 10.1038/s41746-021-00437-0. [Online]. Available: <https://doi.org/10.1038/s41746-021-00437-0>.
- [9] A. S. Alqahtani, P. Saravanan, M. Maheswari, and S. Alshmrany, “An automatic query expansion based on hybrid cmo-coot algorithm for optimized information retrieval,” *The Journal of Supercomputing*, pp. 1–19, 2022.
- [10] G. Caldarini, S. Jaf, and K. McGarry, “A literature survey of recent advances in chatbots,” *Information*, vol. 13, no. 1, p. 41, 2022.
- [11] Dynatrace. (2022). “What is log4shell? understanding the vulnerability in log4j.” Accessed: 2024-10-17, [Online]. Available: <https://www.dynatrace.com/news/blog/what-is-log4shell/>.

- [12] M. A. Leiva, A. J. García, P. Shakarian, and G. I. Simari, “Argumentation-based query answering under uncertainty with application to cybersecurity,” *Big Data and Cognitive Computing*, vol. 6, no. 3, p. 91, 2022.
- [13] J. Lin, “A proposed conceptual framework for a representational approach to information retrieval,” in *ACM SIGIR Forum*, ACM New York, NY, USA, vol. 55, 2022, pp. 1–29.
- [14] K. Liu, F. Wang, Z. Ding, S. Liang, Z. Yu, and Y. Zhou, “Recent progress of using knowledge graph for cybersecurity,” *Electronics*, vol. 11, no. 15, p. 2287, 2022.
- [15] N. Sun, J. Zhang, S. Gao, L. Y. Zhang, S. Camtepe, and Y. Xiang, “Cyber information retrieval through pragmatics understanding and visualization,” *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 2, pp. 1186–1199, 2022.
- [16] Q. Ai, T. Bai, Z. Cao, Y. Chang, J. Chen, Z. Chen, and X. Zhu, “Information retrieval meets large language models: A strategic report from chinese ir community,” *AI Open*, vol. 4, pp. 80–90, 2023.
- [17] M. Alawida, S. Mejri, A. Mehmood, B. Chikhaoui, and O. Isaac Abiodun, “A comprehensive study of chatgpt: Advancements, limitations, and ethical considerations in natural language processing and cybersecurity,” *Information*, vol. 14, no. 8, p. 462, 2023.
- [18] Cloudflare. (2023). “What is wannacry ransomware?” Accessed: 2024-10-17, [Online]. Available: <https://www.cloudflare.com/learning/security/ransomware/wannacry-ransomware/>.
- [19] Fortinet, *2023 cybersecurity skills gap report*, <https://www.fortinet.com/content/dam/fortinet/assets/reports/2023-cybersecurity-skills-gap-report.pdf>, Accessed: 2024-10-17, 2023.
- [20] M. A. Hadi, M. N. Abdulredha, and E. Hasan, “Introduction to chatgpt: A new revolution of artificial intelligence with machine learning algorithms and cybersecurity,” *Sci. Arch*, vol. 4, no. 04, p. 276, 2023.
- [21] K. A. Hambarde and H. Proenca, “Information retrieval: Recent advances and beyond,” *arXiv preprint arXiv:2301.08801*, 2023.
- [22] R. Kumar and S. Sharma, “Hybrid optimization and ontology-based semantic model for efficient text-based information retrieval,” *The Journal of Supercomputing*, vol. 79, no. 2, pp. 2251–2280, 2023.
- [23] —, “Hybrid optimization and ontology-based semantic model for efficient text-based information retrieval,” *The Journal of Supercomputing*, vol. 79, no. 2, pp. 2251–2280, 2023.
- [24] H. Li, Q. Ai, J. Chen, Q. Dong, Y. Wu, Y. Liu, and Q. Tian, “Sailer: Structure-aware pre-trained language model for legal case retrieval,” in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 1035–1044.
- [25] Z.-X. Li, Y.-J. Li, Y.-W. Liu, C. Liu, and N.-X. Zhou, “K-ctiaa: Automatic analysis of cyber threat intelligence based on a knowledge graph,” *Symmetry*, vol. 15, no. 2, p. 337, 2023.

- [26] A. Qammar, H. Wang, J. Ding, A. Naouri, M. Daneshmand, and H. Ning, “Chatbots to chatgpt in a cybersecurity space: Evolution, vulnerabilities, attacks, challenges, and future recommendations,” *arXiv preprint arXiv:2306.09255*, 2023.
- [27] C. Sánchez-Zas, V. A. Villagrà, M. Vega-Barbas, X. Larriva-Novo, J. I. Moreno, and J. Berrocal, “Ontology-based approach to real-time risk management and cyber-situational awareness,” *Future Generation Computer Systems*, vol. 141, pp. 462–472, 2023.
- [28] Y. T. H. Vuong, Q. M. Bui, H. T. Nguyen, *et al.*, “SM-BERT-CR: a deep learning approach for case law retrieval with supporting model,” *Artificial Intelligence and Law*, vol. 31, pp. 601–628, 2023. DOI: 10.1007/s10506-022-09319-6. [Online]. Available: <https://doi.org/10.1007/s10506-022-09319-6>.
- [29] Y. Zhu, H. Yuan, S. Wang, J. Liu, W. Liu, C. Deng, Z. Dou, and J.-R. Wen, “Large language models for information retrieval: A survey,” *arXiv preprint arXiv:2308.07107*, 2023.
- [30] —, “Large language models for information retrieval: A survey,” *arXiv preprint arXiv:2308.07107*, 2023.
- [31] ActZero, *Cybersecurity knowledge gap: No longer an excuse*, <https://actzero.ai/resources/blog/cybersecurity-knowledge-gap-no-longer-an-excuse>, Accessed: 2024-10-17, 2024.
- [32] —, *Cybersecurity knowledge gap: No longer an excuse*, <https://actzero.ai/resources/blog/cybersecurity-knowledge-gap-no-longer-an-excuse>, Accessed: 2024-10-17, 2024.
- [33] Center for Strategic and International Studies (CSIS), *Cybersecurity workforce gap*, <https://www.csis.org/analysis/cybersecurity-workforce-gap/>, Accessed: 2024-10-17, 2024.
- [34] Fortinet, *2024 cybersecurity skills gap report*, <https://www.fortinet.com/content/dam/fortinet/assets/reports/2024-cybersecurity-skills-gap-report.pdf>, Accessed: 2024-10-17, 2024.
- [35] X. Li, J. Jin, Y. Zhou, Y. Zhang, P. Zhang, Y. Zhu, and Z. Dou, “From matching to generation: A survey on generative information retrieval,” *arXiv preprint arXiv:2404.14851*, 2024.
- [36] McKinsey & Company, *Cybersecurity trends: Looking over the horizon*, <https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/cybersecurity/cybersecurity-trends-looking-over-the-horizon>, Accessed: 2024-10-17, 2024.
- [37] S. Muhammad, M. Khan, and S. S. Khan, “A hybrid query-based extractive text summarization based on k-means and latent dirichlet allocation techniques,” *Journal of Artificial Intelligence*, vol. 6, 2024, ISSN: 2579-0021.

- [38] G. V. R. Ram, K. Ashinee, and M. A. Kumar, “End-to-end space-efficient pipeline for natural language query based spacecraft health data analytics using large language model (llm),” in *2024 5th International Conference on Innovative Trends in Information Technology (ICITIIT)*, IEEE, 2024, pp. 1–6.
- [39] M. St John, *Cybersecurity stats: Facts and figures you should know*, <https://www.forbes.com/advisor/education/it-and-tech/cybersecurity-statistics/>, Accessed: 2024-10-17, Aug. 2024.
- [40] The Hacker News. (2024). “The hacker news.” Accessed: 2024-10-17, [Online]. Available: <https://thehackernews.com/>.